IBM Application Development and Problem Determination Tools for z/OS and OS/390

- Overview of the z/OS problem determination and deployment tools
- Coverage of V5 Debug Tool, Fault Analyzer, and File Manager
- Using V2 Application Monitor and V1 Workload Simulator

Eric Chabert
Sylvie Delahaye
Sergei Litvak
Gary Racine

ibm.com/redbooks

Redbooks

International Technical Support Organization

IBM Application Development and Problem Determination Tools for z/OS and OS/390

July 2005

SG24-6492-00

Note: Before using this information and the product it supports, read the information in “Notices” on page xv.

First Edition (July 2005)

This edition applies to IBM Application Monitor for z/OS, Version 2, Release 1, IBM Debug Tool for z/OS, Version 5, Release 1, IBM Fault Analyzer for z/OS, Version 5, Release 1, IBM File Manager for z/OS, Version 5, Release 1, and IBM Workload Simulator for z/OS and OS/390, Version 1, Release 1.

Contents

Notices xv
Trademarks xvi

Preface xvii
The team that wrote this redbook. xvii
Become a published author xix
Comments welcome. xix

Part 1. Executive summary and overview 1

Chapter 1. Executive summary 3
1.1 Introduction 4
1.2 Application Monitor for z/OS 4
1.3 Debug Tool and Debug Tool Utilities & Advanced Functions 5
1.4 Fault Analyzer for z/OS 5
1.5 File Manager for z/OS 6
1.6 Workload Simulator for z/OS and OS/390 7

Chapter 2. Overview of the problem determination and deployment tools 9
2.1 Application Monitor for z/OS 10
2.1.1 Use of Application Monitor 10
2.1.2 What Application Monitor can do 11
2.1.3 The major functions of Application Monitor. 12
2.2 Debug Tool for z/OS and Debug Tool Utilities and Advanced Functions 13
2.2.1 What Debug Tool can do 14
2.2.2 What Debug Tool provides 15
2.2.3 Debug Tool Utilities and Advanced Functions 16
2.3 Fault Analyzer for z/OS 16
2.3.1 Locate application failures quickly to reduce development costs 17
2.3.2 Maximizing development assets 17
2.3.3 Analyze application failures to improve developer productivity. 18
2.3.4 Fault Analyzer functions to optimize efficiencies. 18
2.4 File Manager for z/OS 20
2.4.1 Enhance development productivity through structured data support 20
2.4.2 Maximize flexibility with the File Manager base component 20
2.4.3 Manage data more efficiently with the File Manager DB2 component 22
2.4.4 Develop faster, less costly applications with the File Manager IMS component. . 22
2.4.5 Flexible file processing to meet your business goals 22
2.4.6 For more information. 23
2.5 Workload Simulator for z/OS and OS/390 23
2.5.1 What Workload Simulator can do 23
2.5.2 What Workload Simulator provides. 24
2.5.3 The latest enhancements 25

Part 2. Introduction to Problem Determination and Deployment Tools 27

Chapter 3. Introduction to Application Monitor for z/OS 29
3.1 How Application Monitor samples an application 30
3.2 How to get help 30

3.3 The functions of Application Monitor . . . 30
3.3.1 Online analysis . . . 30
3.3.2 Program module reports . . . 31
3.3.3 Delay analysis reports . . . 31
3.3.4 Data set analysis reports . . . 31
3.3.5 DB2 analysis . . . 31
3.3.6 Monitoring CICS . . . 32
3.3.7 Monitoring IMS . . . 32
3.3.8 Monitoring jobs . . . 32
3.3.9 Deferred monitoring request . . . 33
3.3.10 History data set . . . 33

Chapter 4. Introduction to Debug Tool and Debug Tool Utilities and Advanced Functions 35
4.1 Debug Tool . . . 36
4.1.1 Debug Tool interfaces . . . 36
4.1.2 Compiler options . . . 43
4.1.3 Link-edit options . . . 43
4.1.4 Runtime TEST option . . . 43
4.1.5 Special files . . . 48
4.1.6 Global preferences file enhancement . . . 49
4.1.7 Frequently used commands . . . 50
4.1.8 Other useful commands . . . 53
4.1.9 Finishing a Debug Tool session . . . 54
4.1.10 Built-in functions . . . 54
4.1.11 Dynamic Debug Facility . . . 55
4.2 Debug Tool Utilities and Advanced Functions . . . 55
4.2.1 Debug Tool Utilities . . . 55
4.2.2 Advanced functions . . . 62

Chapter 5. Introduction to Fault Analyzer for z/OS . . . 63
5.1 Validating your software level . . . 64
5.1.1 PTF information . . . 64
5.2 Mechanics of Fault Analyzer . . . 67
5.2.1 Fault history file . . . 67
5.2.2 Supported application environments . . . 68
5.2.3 Summary of real-time analysis . . . 69
5.3 Preparing your program for Fault Analyzer . . . 72
5.3.1 Compiler options . . . 72
5.3.2 TEST option considerations . . . 73

Chapter 6. Introduction to File Manager for z/OS . . . 75
6.1 Overview . . . 76
6.1.1 File Manager for working with MVS data sets . . . 76
6.1.2 File Manager for DB2 data . . . 77
6.1.3 File Manager for IMS data . . . 77
6.2 Major functions . . . 77
6.2.1 Support for DFSORT . . . 78
6.2.2 Templates . . . 78
6.2.3 REXX functions . . . 78
6.2.4 Batch processing . . . 79

Chapter 7. Introduction to Workload Simulator . . . 81
7.1 Overview . . . 82

7.1.1 Use of WSim . . . 83
7.1.2 Resources WSim can simulate . . . 83
7.1.3 Testing with WSim . . . 83
7.2 System configuration . . . 84
7.2.1 Physical configurations . . . 85
7.2.2 Logical configurations . . . 85
7.3 Script preparation . . . 86
7.3.1 Network definition statements . . . 86
7.3.2 Message generation decks . . . 86
7.3.3 Methods for creating message decks . . . 87
7.3.4 Testing scripts . . . 90
7.4 WSim output . . . 90
7.5 Operating WSim . . . 91
7.6 WSim Test Manager . . . 91
7.7 Latest enhancements . . . 92

Part 3. Application Monitor for z/OS . . . 95

Chapter 8. Application Monitor: Settings and customization . . . 97
8.1 Application Monitor Primary Options panel . . . 98
8.2 Application Monitor Set Your Defaults panel . . . 98
8.2.1 Specify Default Sampling Parameters . . . 99
8.2.2 Specify History File Name and Options . . . 101
8.2.3 Specify History Allocation Options . . . 102
8.2.4 Specify COBOL and PL/I Listing Data Sets . . . 103

Chapter 9. Application Monitor: Selecting a job . . . 105
9.1 Active job selection . . . 106
9.2 Select an Active Job to Analyze . . . 106
9.2.1 How to read this panel . . . 107
9.2.2 How to select your job from this panel . . . 108
9.3 Analyze a Specified Job . . . 110
9.4 Managing and analyzing a named set of jobs . . . 113
9.4.1 Managing named sets . . . 113
9.4.2 Analyze a named set of jobs . . . 117
9.5 Deferred monitoring request . . . 119
9.5.1 How to define a deferred monitoring request . . . 119
9.5.2 How to administer a deferred monitoring request . . . 121

Chapter 10. Application Monitor: Analyzing a job . . . 125
10.1 Special features of history data sets . . . 126
10.1.1 Analyze Historical Data . . . 126
10.1.2 Select a History Data Set . . . 127
10.1.3 Specify the Time Intervals to Analyze . . . 128
10.2 Program module reports . . . 130
10.2.1 CPU and Loadmod Analysis . . . 130
10.2.2 Statement Analysis . . . 134
10.2.3 Detail Statement Display . . . 135
10.2.4 Instruction Analysis . . . 136
10.2.5 Detail Instruction Display . . . 137
10.2.6 Statement Analysis Member selection . . . 137
10.3 Delay Analysis Report . . . 139
10.4 Data Set analysis . . . 141
10.4.1 Data Set Analysis Report . . . 142

10.4.2 VSAM Data Set Analysis Detail Report . . . 143

Chapter 11. Application Monitor: Analyzing a job with subsystems . . . 149
11.1 CICS analysis . . . 150
11.1.1 CICS customization . . . 150
11.1.2 CICS displays . . . 151
11.1.3 CICS Transactions display . . . 152
11.1.4 CICS Calls in a Transaction display . . . 153
11.1.5 CICS Transaction Executions display . . . 155
11.1.6 CICS Calls in a Unit of Work display . . . 156
11.2 DB2 analysis . . . 157
11.2.1 DB2 customization . . . 157
11.2.2 DB2 displays . . . 159
11.2.3 SQL Analysis display . . . 161
11.2.4 SQL Statement Summary display . . . 162
11.2.5 SQL Statement Full Text . . . 163
11.2.6 SQL Detailed Explain Analysis . . . 164
11.2.7 SQL Explain Detail display . . . 167
11.2.8 DB2 Table Information display . . . 170
11.2.9 SQL End Completion Statistics display . . . 171
11.2.10 SQL Statement Completion display . . . 172
11.3 IMS analysis . . . 173
11.3.1 IMS customization . . . 173
11.3.2 IMS messages processed by region display . . . 173
11.3.3 IMS Message Analysis display . . . 176
11.3.4 IMS Message Call Analysis display . . . 177
11.3.5 IMS segment search criteria analysis display . . . 179

Chapter 12. Application Monitor and batch reports . . . 181
12.1 Overview . . . 182
12.2 JCL - FBIBATCH . . . 182
12.3 Printed report layouts . . . 186
12.3.1 The title page . . . 186
12.3.2 Jobs, Steps, and Time Ranges page . . . 186
12.3.3 Historical Reports available . . . 188
12.3.4 Detailed Analysis reports . . . 188

Part 4. Debug Tool and Debug Tool Utilities and Advanced Functions . . . 191

Chapter 13. Debugging programs using Debug Tool . . . 193
13.1 Debugging C/C++ programs using Debug Tool . . . 194
13.1.1 Preparing the program . . . 194
13.1.2 Using mainframe interface debugger (3270 interface debugger) . . . 198
13.1.3 Using IBM Distributed Debugger . . . 204
13.2 Debugging PL/I programs using Debug Tool . . . 217
13.2.1 Preparing a PL/I program for debugging . . . 217
13.2.2 Debugging a PL/I application with the mainframe interface (MFI) . . . 220
13.2.3 Debugging a PL/I application with the remote interface . . . 229
13.3 Debugging COBOL programs using Debug Tool . . . 240
13.3.1 Preparing an OS/VS COBOL application for debugging using JCL . . . 240
13.3.2 Preparing a COBOL for z/OS and OS/390 application using DTU&AF . . . 246
13.3.3 Using MainFrame Interface debugger . . . 259
13.3.4 Using IBM Distributed Debugger . . . 265

13.4 Debugging non-LE programs using Debug Tool . . . 268
13.4.1 Debugging exclusively non-LE programs . . . 268
13.4.2 Debugging MVS batch or TSO non-LE initial programs . . . 268
13.4.3 Debugging CICS non-LE initial programs . . . 269
13.5 Debugging Assembler programs using Debug Tool . . . 269
13.5.1 Preparing the program . . . 269
13.5.2 Preparing programs and modifying setup files with Debug Tool Utilities . . . 270
13.5.3 Assembling your program using Debug Tool Utilities . . . 271
13.5.4 Modifying and using a setup file . . . 286
13.5.5 Using MainFrame Interface debugger. . . . 291
13.5.6 Using IBM Distributed Debugger. . . . 292
13.5.7 Debug Tool commands supported in remote debug mode . . . 295

Chapter 14. Using Debug Tool under UNIX System Services . . . 297
14.1 Common considerations for all languages . . . 298
14.1.1 Specifying the TEST runtime option . . . 298
14.1.2 Special considerations for running DLL applications from the shell . . . 299
14.2 Using C/C++ and Debug Tool under USS . . . 299
14.2.1 Compiling a C/C++ application . . . 300
14.2.2 Debugging a USS application . . . 300

Chapter 15. Debug Tool and subsystems . . . 305
15.1 Debug Tool and CICS . . . 306
15.1.1 How to add support for debugging under CICS . . . 306
15.1.2 How to prepare a CICS program. . . . 314
15.1.3 How to start Debug Tool under CICS . . . 335
15.1.4 How to debug CICS programs . . . 343
15.2 Debug Tool and DB2. . . . 346
15.2.1 Compiling for DB2 and Debug Tool . . . 346
15.2.2 Invoking a DB2 application using the TEST runtime option . . . 346
15.2.3 Using DTSU to invoke your DB2 application. . . . 346
15.2.4 DB2 stored procedures considerations. . . . 348
15.3 Debug Tool and IMS . . . 349
15.3.1 Traditional methods of debugging under IMS . . . 349
15.3.2 Debugging with IMS V8 and above. . . . 349
15.3.3 Precedence of the information on the LE Runtime Options panel . . . 362

Chapter 16. Debug Tool Utilities and Advanced Functions. . . . 365
16.1 Coverage Utility. . . . 366
16.1.1 DTCU customization . . . 366
16.1.2 DTCU requirements . . . 368
16.1.3 Running DTCU . . . 368
16.1.4 Using DCTU . . . 369
16.1.5 Modifying your DTCU defaults . . . 370
16.1.6 Compiling the program to be measured . . . 371
16.1.7 Setting up your control file . . . 372
16.1.8 Create the setup JCL . . . 373
16.1.9 Create JCL to start a monitor session. . . . 374
16.1.10 Create JCL for a report . . . 375
16.1.11 Jobs generated so far . . . 376
16.1.12 Perform the setup step . . . 377
16.1.13 Perform the link step . . . 377
16.1.14 Start the monitor session . . . 377
16.1.15 Start your application . . . 377

16.1.16 Display statistics online . . . 377
16.1.17 Stop the monitor . . . 378
16.1.18 Create the summary report . . . 379
16.1.19 Create the summary and annotated listing report . . . 381
16.1.20 Use of the monitor panel . . . 382
16.2 COBOL and CICS Command Level Conversion Aid . . . 383
16.2.1 Requirements for the conversion tools . . . 383
16.2.2 Setting up for convert and compile . . . 383
16.2.3 Start the conversion process . . . 385
16.2.4 Convert old 68/74 COBOL to 85 COBOL . . . 388
16.2.5 Summary . . . 392

Part 5. Fault Analyzer for z/OS . . . 393

Chapter 17. Fault Analyzer settings and customization . . . 395
17.1 Invocation exits . . . 396
17.2 Fault Analyzer ISPF interface . . . 396
17.2.1 Invoking the interface . . . 398
17.2.2 Using views . . . 400
17.2.3 Setting up views . . . 401
17.2.4 Changing the history file or the view displayed . . . 406
17.2.5 Fault entry list column configuration . . . 409
17.2.6 Matching and selecting fault . . . 412
17.2.7 Applying an action against a particular fault . . . 416
17.2.8 History file properties . . . 417
17.2.9 Resetting history file access information . . . 417
17.2.10 Refreshing fault entry information . . . 418
17.2.11 Actions command . . . 418
17.2.12 Action bar pull-down menus . . . 418

Chapter 18. Fault Analyzer interactive reanalysis . . . 423
18.1 Interactive reanalysis options . . . 424
18.2 Initiating interactive reanalysis . . . 427
18.3 General information about the interactive report . . . 428
18.4 Exit from the interactive report . . . 428
18.5 Synopsis . . . 428
18.6 Event summary . . . 429
18.6.1 Detailed event information . . . 430
18.7 System-wide information . . . 432
18.8 Abend job information . . . 447
18.9 Options in effect . . . 448
18.10 Displaying associated storage areas . . . 448
18.11 Expanding messages and abend codes . . . 451
18.12 Displaying source code . . . 454
18.13 Displaying storage locations . . . 455
18.13.1 Creating and managing user notes . . . 455
18.14 Mapping storage areas using DSECT information . . . 459
18.14.1 IDIDSECT concatenation . . . 460
18.14.2 Indexing . . . 460
18.14.3 DSECT indexing utility . . . 460
18.15 Displaying chained data areas . . . 461
18.16 Disassembling object code . . . 462
18.17 Converting STORE CLOCK values . . . 463
18.18 User-specific report formatting . . . 463

18.19 Prompting for compiler listing or side file . . . 465
18.19.1 Controlling prompting . . . 466
18.20 Data sets used for interactive reanalysis . . . 467

Chapter 19. Fault Analyzer real-time analysis report . . . 469
19.1 Successful analysis processing . . . 470
19.2 Fault history file selection . . . 470
19.3 Controlling the real-time analysis with options . . . 470
19.4 Pointing to listings with JCL DD statements . . . 471
19.5 Real-time analysis report . . . 471
19.6 SYSLOG summary . . . 472
19.7 Using the program SNAP interface (IDISNAP) . . . 472
19.8 Dump registration processing . . . 472

Chapter 20. Fault Analyzer batch reanalysis report . . . 475
20.1 Batch reanalysis options . . . 476
20.2 Initiating batch reanalysis . . . 478
20.3 Data sets used for batch reanalysis . . . 478
20.4 Creating your own batch reanalysis job . . . 479

Chapter 21. Fault Analyzer and subsystems . . . 481
21.1 Fault Analyzer and CICS . . . 482
21.1.1 Configuring Language Environment for CICS to invoke Fault Analyzer . . . 483
21.1.2 Defining required program to CICS . . . 483
21.1.3 Controlling CICS transaction abend analysis . . . 484
21.1.4 Using a CFA from a CICS terminal . . . 485
21.1.5 Using CFA from an MVS console . . . 486
21.1.6 Ensuring transaction abend analysis is not suppressed by DUMP(NO) . . . 486
21.1.7 Preventing LE from causing the CICS trace to wrap . . . 486
21.1.8 Specifying data sets through the IDIOPTS DDname . . . 487
21.1.9 Language Environment abend considerations . . . 488
21.1.10 Installing the MVS post-dump exit IDIXTSEL . . . 488
21.1.11 Storage requirements . . . 489
21.2 Fault Analyzer and DB2 . . . 502
21.2.1 Binding DB2 . . . 502
21.2.2 DB2 and Language Environment . . . 502
21.2.3 DB2 stored procedures . . . 503
21.3 Fault Analyzer and IMS . . . 505
21.3.1 IMS and Language Environment . . . 505
21.4 Performing WebSphere or Java dump analysis . . . 505
21.4.1 Setting options for WebSphere or Java analysis . . . 506
21.4.2 Type of dumps supported . . . 506
21.4.3 Selecting a WebSphere or Java dump data set . . . 506

Chapter 22. Fault Analyzer additional information . . . 509
22.1 Requirements at a glance . . . 510
22.1.1 Hardware requirements . . . 510
22.1.2 Software requirements . . . 510
22.1.3 Additional requirements . . . 510
22.2 Target system requirements . . . 511
22.2.1 Library names after you finish installing . . . 512
22.3 Healthy installation and customization checklist . . . 512

Part 6. File Manager for z/OS 515

Chapter 23. Using File Manager 517
23.1 What File Manager can be used for 518
23.1.1 Performance guidelines 518
23.1.2 Conventions used 519
23.1.3 Simple copy 519
23.1.4 Selective copying 521
23.1.5 How to perform a global find and replace in a PDS 522
23.1.6 How to create one VSAM file using another as a model 528
23.1.7 How to initialize a VSAM file with low-value records 531
23.1.8 How to populate this newly created file 533
23.1.9 How to split a single file into constituent record types 538
23.1.10 Replace a string in a specific location in a file 541
23.1.11 Copy selected variably blocked records to another file 541
23.1.12 Search for a string in all members of a PDS 546
23.1.13 Multiple find 548
23.1.14 QSAM allocation using a model 550
23.1.15 Work with VTOC 551
23.1.16 Get information about a load module 554
23.1.17 Dynamic template 555
23.1.18 Compare data sets using templates 558

Chapter 24. File Manager/DB2 561
24.1 Overview of FM/DB2 562
24.2 How to edit a table 562
24.3 Utility function to create a table 567
24.4 Copy data to a table 572

Chapter 25. File Manager/IMS 575
25.1 Navigating an IMS database 576
25.2 Template processing 586
25.2.1 It really does remember the copybook 586
25.2.2 Using the SELECT command in edit templates 587
25.2.3 How to process COPY REPLACING statements 591
25.2.4 How to build a template for multi-record file layouts 593
25.3 Hints and tips 593
25.3.1 System programmer notes 594
25.3.2 Look out for your PF keys 595
25.3.3 What to do when a copybook fails to compile 595
25.3.4 Record structure defined in source application program 597
25.3.5 Watch out for that bad disposition 597
25.4 Copybook and include 597

Chapter 26. Test data and File Manager 599
26.1 Creating MVS data using File Manager and copybooks 600
26.1.1 Data creation on line 600
26.1.2 Data creation in a batch mode 603
26.1.3 Field attributes for template-based data creation 604
26.2 Copying MVS data using File Manager 606
26.2.1 Data copy on line 606
26.2.2 Data copy in a batch mode 610
26.3 Creating DB2 test data using File Manager 611
26.3.1 DB2 data creation on line 611

26.3.2 DB2 data creation in batch mode . . . 615
26.4 Copying DB2 data using File Manager . . . 615
26.4.1 DB2 data copy on line . . . 615
26.4.2 DB2 data copy in a batch mode . . . 618
26.5 Importing and exporting DB2 data using File Manager . . . 619
26.5.1 DB2 data export on line . . . 619
26.5.2 DB2 data export in a batch mode . . . 624
26.5.3 DB2 data import on line . . . 624
26.5.4 Data import to DB2 in batch mode . . . 627
26.6 Extract and load IMS data using File Manager . . . 627
26.6.1 IMS data extract . . . 628
26.6.2 Load data extracted from IMS by File Manager . . . 635
26.7 File Export . . . 638
26.7.1 Copybooks: File Manager versus File Export . . . 641
26.7.2 A few samples . . . 643
26.7.3 Relationship specification: Filter . . . 656
26.7.4 Relationship specification: External reference . . . 660
26.7.5 File Export versus File Manager in batch . . . 662

Part 7. Workload Simulator for z/OS and OS/390 . . . 665

Chapter 27. Workload Simulator and Workload Simulator Test Manager settings and customization . . . 667
27.1 Installing WSim . . . 668
27.2 Installing WTM . . . 669
27.2.1 Creating a VTAM application major node . . . 669
27.2.2 Providing access to WTM . . . 670
27.2.3 Setting up a new WTM user . . . 671
27.2.4 Latest enhancements . . . 672

Chapter 28. Workload Simulator Test Manager . . . 675
28.1 Concepts . . . 676
28.2 WTM operating modes . . . 676
28.3 Creating a project . . . 677
28.4 Test cases . . . 680
28.5 Scheduling and running a simulation . . . 698
28.6 Simulation reports . . . 703
28.7 Organizing test cases . . . 708
28.7.1 Creating modular scripts . . . 709
28.7.2 Creating a test group . . . 740
28.8 User data tables . . . 749
28.9 WSim in batch mode . . . 765
28.10 Regression test . . . 770
28.11 Display Monitor Facility . . . 776
28.12 Load test . . . 778
28.13 Using SNA trace to create test case . . . 788

Chapter 29. Basics of scripting for Workload Simulator . . . 791
29.1 WTM-generated scripts review . . . 792
29.2 Network definition statements . . . 797
29.3 Coding STL programs . . . 800
29.4 WSim operator commands . . . 804

Part 8. Scenarios . . . 805

Chapter 30. Introduction to the scenarios . . . 807
30.1 Overview . . . 808
30.1.1 Overview of the programs . . . 808
30.1.2 Software prerequisites . . . 810
30.2 Install the application software . . . 810
30.2.1 Install the demo files . . . 810
30.2.2 Customization . . . 811
30.2.3 Set up the applications . . . 812
30.3 About the system configuration . . . 815
30.3.1 About the software prerequisites . . . 815
30.3.2 About the CICS configuration . . . 815
30.3.3 About the DB2 configuration . . . 815
30.3.4 Validate the installation . . . 816

Chapter 31. Guided tour of the Trader application . . . 817
31.1 Getting started . . . 818
31.2 Trader application in CICS . . . 818
31.2.1 Log on to the application . . . 818
31.2.2 Running the Trader application . . . 819
31.2.3 Obtain real-time quotes . . . 820
31.2.4 Buy shares . . . 820
31.2.5 Sell shares . . . 821
31.3 Running the Trader application in batch . . . 822
31.3.1 Run the batch job . . . 822
31.3.2 Check the result . . . 823

Chapter 32. Scenario 1: Invalid VSAM data generating an abend . . . 825
32.1 Set up the components . . . 826
32.1.1 CICS components . . . 826
32.1.2 Program products . . . 826
32.2 Tracking an abend in the application . . . 827
32.2.1 Viewing the abend in Fault Analyzer . . . 828
32.2.2 Using Debug Tool to analyze the problem . . . 834
32.2.3 Using File Manager to correct a problem with data . . . 838
32.2.4 Running the application after the fix . . . 840
32.3 Summary of Scenario 1 . . . 841

Chapter 33. Scenario 2: Using Debug Tool . . . 843
33.1 Set up the components . . . 844
33.1.1 Batch components . . . 844
33.1.2 Program products . . . 844
33.2 Walkthrough of the batch Trader application . . . 844
33.2.1 The Trader batch job . . . 845
33.2.2 Listing shares . . . 846
33.2.3 Buying and selling shares . . . 847
33.2.4 Using Debug Tool in batch mode to try to find the error . . . 849
33.2.5 Using Debug Tool in DTSU to pinpoint the solution . . . 852
33.2.6 Executing the batch application after the fix . . . 868
33.2.7 Summary of scenario 2 . . . 868

Chapter 34. Is the error in DB2 data or program logic? . . . 869
34.1 Set up the components . . . 870

34.1.1 CICS and DB2 components . . . 870
34.1.2 Program products . . . 870
34.2 Tracking a problem in the application . . . 871
34.2.1 Recreating the error . . . 871
34.2.2 Viewing the data in File Manager/DB2 . . . 873
34.2.3 Using Debug Tool to identify the logic problem. . . . 876
34.2.4 Using File Manager/DB2 to correct the data . . . 882
34.3 Summary of Scenario 3. . . . 884

Chapter 35. Using Fault Analyzer and File Manager . . . 887
35.1 Set up the components . . . 888
35.1.1 CICS components . . . 888
35.1.2 Program products . . . 888
35.2 Walkthrough of the CICS Trader application. . . . 889
35.2.1 Log on to the application . . . 889
35.2.2 Obtaining quotes . . . 891
35.2.3 Buying shares . . . 891
35.2.4 Selling shares . . . 892
35.3 Tracking an abend in the application . . . 893
35.3.1 Viewing the abend in Fault Analyzer. . . . 893
35.3.2 Initiating interactive reanalysis for the abend . . . 895
35.3.3 Using File Manager to correct a problem with data. . . . 898
35.3.4 Running the application after the fix . . . 902
35.4 Summary of Scenario . . . 903

Chapter 36. Application Monitor and CICSplex configuration . . . 905
36.1 Application Monitor and Transaction Routing . . . 906
36.1.1 System configuration . . . 906
36.1.2 Application Monitor set-up . . . 906
36.1.3 Application Monitors reports . . . 909
36.2 Application Monitor and Function Shipping. . . . 920
36.2.1 System configuration . . . 920
36.2.2 Application Monitor set-up . . . 920
36.2.3 Application Monitor reports . . . 920
36.3 Application Monitor and Dynamic Program Link . . . 921
36.3.1 System configuration . . . 921
36.3.2 Application Monitor set-up . . . 922
36.3.3 Application Monitor reports . . . 922
36.4 Conclusions. . . . 925

Chapter 37. Application Monitor and Parallel Sysplex . . . 927
37.1 Batch job in a Parallel Sysplex environment . . . 928
37.1.1 System configuration . . . 928
37.1.2 Application Monitor set-up . . . 928
37.1.3 Application Monitor reports . . . 929
37.1.4 Conclusion . . . 933
37.2 CICS and DB2 data sharing group . . . 933
37.2.1 System configuration . . . 933
37.2.2 Application Monitor set-up . . . 935
37.2.3 Application Monitor reports . . . 936
37.2.4 Conclusion . . . 940

Part 9. Using WSED with Debug Tool 941

Chapter 38. Remote development: Debugger 943
38.1 Overview of the Debug perspective 944
38.2 Controlling the flow 948
38.2.1 Controls in the Debug View 948
38.2.2 Entry breakpoint 950
38.2.3 Address breakpoint 951
38.2.4 Watch breakpoint 951
38.2.5 Load breakpoint 952
38.2.6 Line breakpoint 952
38.2.7 Disable breakpoints 953
38.3 Dealing with variables 953
38.4 Debug Console 957
38.5 Exception handling 959

Chapter 39. Using PL/I or COBOL DB2 stored procedures with WSED 963
39.1 Overview of DB2 stored procedures 964
39.1.1 DB2 stored procedure types 965
39.2 Creating stored procedure using WSED 966
39.2.1 Allocating the z/OS data sets for stored procedure. 966
39.2.2 Setting up the z/OS build properties 967
39.2.3 Connections required to z/OS when creating the stored procedures 968
39.2.4 Tips on creating stored procedures. 969
39.2.5 Create a COBOL stored procedure. 969
39.2.6 Adding program logic and SQL statements to the source code 974
39.3 Building the 390 stored procedure 978
39.3.1 Checking COBOL syntax before doing the building 979
39.3.2 Process for building a DB2 stored procedure 980
39.3.3 Checking for compiling and building errors. 982
39.4 Running a stored procedure 983
39.5 Removing a stored procedure 984
39.6 Changing a stored procedure 985
39.7 Debugging a stored procedure 985
39.8 Current stored procedures SQL assist wizard limitations 989

Part 10. Appendixes 993

Appendix A. Additional material 995
Locating the Web material 995
Using the Web material 995
System requirements for downloading the Web material 996
How to use the Web material 996

Related publications 997
IBM Redbooks 997
Other publications 997
Online resources 998
How to get IBM Redbooks 999
Help from IBM 999

Index 1001

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:
This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

@server®
@server®
Redbooks (logo) ™
Redbooks (logo)™
ibm.com®
eServer™
z/OS®
zSeries®
AIX®
BookManager®
CICS/ESA®
CICS®
CICSPlex®
DB2 Universal Database™
DB2®
DFSORT™
ESCON®
IBM®
IMS™
Language Environment®
MQSeries®
MVS™
MVS/ESA™
NetView®
OS/390®
Parallel Sysplex®
Rational®
Redbooks™
RACF®
RETAIN®
S/390®
VisualAge®
VTAM®
WebSphere®

The following terms are trademarks of other companies:

EJB, Java, JVM, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Windows NT, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel, Pentium, Intel logo, Intel Inside logo, and Intel Centrino logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, and service names may be trademarks or service marks of others.

Preface

This IBM® Redbook introduces the IBM Problem Determination and Deployment Tools for z/OS® and OS/390®.

The tools covered are:

- IBM Application Monitor for z/OS, Version 2, Release 1
- IBM Debug Tool for z/OS, Version 5, Release 1
- IBM Fault Analyzer for z/OS, Version 5, Release 1
- IBM File Manager for z/OS, Version 5, Release 1
- IBM Workload Simulator for z/OS and OS/390, Version 1, Release 1

This comprehensive suite of powerful yet easy-to-use tools helps developers to efficiently build, test, and service applications while minimizing errors and increasing cost-effectiveness. The tools allow users to quickly identify and resolve problems that occur in batch, IMS, DB2, CICS, and UNIX System Services applications.

For each tool, this book provides step-by-step instructions for installation and customization, a detailed discussion of the features and capabilities, and guidance for using the tool in your environment.

Practical scenarios demonstrate how to apply the tools to monitor, manage, debug, and analyze applications in a real-world environment.

The team that wrote this redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization, Raleigh Center.

Eric Chabert is an IT Specialist in Advanced Technical Support (ATS) in Montpellier, France. He has 16 years of experience with IBM, predominantly as a CICS® and CICSPlex® System Manager specialist, but now with responsibilities for all PD tools for z/OS.

Sylvie Delahaye is a software pre-sales IT specialist on the WebSphere/390 team in Paris, France. During 17 years at IBM, her diverse experience has included assignments in the Translation Center, CICS and MQSeries® marketing, Techline, and finally the software group for S/390®. Sylvie has extensive practical experience with the tools and languages of S/390.

Sergei Litvak is an IT Specialist with Securities Industry Services, IBM Canada, in Toronto. He has over 20 years of experience in the computer technology field. Sergei joined IBM in 1999. His areas of expertise include PL/I, COBOL, IMS, and DB2. This is his second redbook.

Gary Racine is an Advisory Technical Services Professional in the IBM Global Services, Strategic Outsourcing Group in Canada. He provides ongoing support and software upgrades to multi-client mainframe environments, including multi-IBM and third-party software products. Gary has strong skills in numerous software designs and architectures.

Figure 0-1 From left - Eric Chabert, Gary Racine, and Sergei Litvak. Not pictured - Sylvie Delahaye

Thanks to the following people for their contributions to this project:

James Conover, System Programmer, IBM, Buffalo

Jim McIntosh, IBM, Tools Client Implementation Advocate

Ken Hume, IBM, Tools Client Implementation Advocate

Ira Sheftman, Fault Analyzer Specialist, IBM, Silicon Valley Laboratory

Jenny Mao, File Manager Specialist, IBM, Silicon Valley Laboratory

Jean Nardi, Debug Tool Specialist, IBM, Silicon Valley Laboratory

Francisco Anaya, Debug Tool Specialist, IBM, Silicon Valley Laboratory

Terry Harris, Workload Simulator Specialist, IBM, Silicon Valley Laboratory

Kevin Cummings, Tools Client Implementation Advocates Team Leader, IBM Canada

Michael Engledew, File Manager and Fault Analyser Product Manager, IBM, Silicon Valley Laboratory

William Deason, AD Tools Solution Test Manager, IBM, Silicon Valley Laboratory

John Leake, Application Monitor Project Manager, IBM, Silicon Valley Laboratory

Lisa Ready, IBM, Silicon Valley Laboratory

Ken Edwards, IBM, Silicon Valley Laboratory

Marty Shelton, IBM, Silicon Valley Laboratory

David Tran, IBM, Silicon Valley Laboratory

Kevin Poole, IBM, Silicon Valley Laboratory

Alison Chandler, Content Editorial Services, IBM ITSO Poughkeepsie

Joe DeCarlo, Manager for Special Projects, IBM ITSO Raleigh

Become a published author

Join us for a two- to six-week residency program! Help write an IBM Redbook dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You'll team with IBM technical professionals, Business Partners and/or customers.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you'll develop a network of contacts in IBM development labs, and increase your productivity and marketability.

Find out more about the residency program, browse the residency index, and apply online at:

`ibm.com/redbooks/residencies.html`

Comments welcome

Your comments are important to us!

We want our Redbooks™ to be as helpful as possible. Send us your comments about this or other Redbooks in one of the following ways:

- Use the online **Contact us** review redbook form found at:

  `ibm.com/redbooks`

- Send your comments in an email to:

  `redbook@us.ibm.com`

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. HZ8, Building 662
  P.O. Box 12195
  Research Triangle Park, NC 27709-2195

Part 1

Executive summary and overview

Part one presents an executive summary followed by an overview of IBM z/OS Problem Determination and Deployment Tools, including Application Monitor for z/OS, Debug Tool for z/OS and Debug Tool Utilities and Advanced Functions, Fault Analyzer for z/OS, File Manager for z/OS, and Workload Simulator for z/OS and OS/390.

1

Executive summary

This chapter provides a high level description of IBM z/OS Problem Determination and Deployment Tools, which include:

- Application Monitor for z/OS
- Debug Tool for z/OS and Debug Tool Utilities and Advanced Functions
- Fault Analyzer for z/OS
- File Manager for z/OS
- Workload Simulator for z/OS and OS/390

Hereafter these products are referred to as:

- Application Monitor (AM)
- Debug Tool (DT) and Debug Tool Utilities and Advanced Functions (DTU&AF)
- Fault Analyzer (FA)
- File Manager (FM)
- Workload Simulator (WSim)

Your IT staff faces increasing pressure to deliver function-rich applications quickly to meet your company's business objectives. Whether designed to perform routine or critical tasks, the underlying code that drives today's applications can be highly complex. Programmers have to minimize errors while working quickly to meet demand - even adapting code on the fly as your business needs change. To effectively build and service applications, developers need robust, easy-to-use tools to compile, test, and debug their applications. IBM z/OS Problem Determination and Deployment Tools will help them to face these challenges.

1.1 Introduction

The e-business paradigm demands more and more complex applications, driving a requirement for more sophisticated tooling. Software applications have become more complex, and so application development has become more and more expensive. IBM has a comprehensive suite of powerful tools that you can use for improving the health of your application portfolio and, at the same time, address the growing need for cost-effective application development and testing.

IBM recognizes that zSeries®-related software is one of the cornerstones of enterprise computing. IBM is committed to providing our customers with the most reliable, scalable, and cost-effective solutions; and IBM is working hard to evolve products like AM, DT and DTU&AF, FA, FM, and WSim.

This IBM Redbook serves as a comprehensive reference for a set of tools that can help your organization to better manage z/OS-based applications.

1.2 Application Monitor for z/OS

Application Monitor for z/OS is a tool for analyzing application performance. Application Monitor evaluates application prototypes in the design phase, reviews the impact of increased data volume or changes in business requirements on performance, and generates historical data and reports to analyze performance trends and evaluate program changes for cost effectiveness. To Application Monitor, a job is an activity or subsystem that executes in its own MVS™ address space, and can be:

- A batch job with possible access to DB2 and IMS databases
- A CICS region with possible access to DB2 and IMS databases
- An IMS dependent region with possible access to DB2 or IMS databases

For a batch job, Application Monitor enables you to display detailed performance data. If you make the program compilation listing available to Application Monitor, you can drill down to the statement level in the program being executed to display detailed information about statement execution. Application Monitor can also monitor jobs that execute in a sysplex environment where you may not know the system where the job will be scheduled for execution.

For CICS, you can monitor a CICS region as a whole or display detailed information about a transaction that is running in the region or that can run on multiple regions. For the selected transaction, you can drill down to the statement level in the program being executed if you make the program compilation listing available to Application Monitor. You can also display detailed information about the files or databases being accessed.

For IMS applications, you can use Application Monitor to monitor the activity in IMS dependent regions, including batch, Batch Message Processing (BMP), Message Processing Program (MPP), and Fast Path (FP) dependent regions. You can monitor an IMS dependent region as a whole and select a particular message or batch program to analyze in detail. If you make the program compilation listing available to Application Monitor, you can drill down to the statement level in the program or programs being executed. You can also display detailed information about the files or databases the target program or message accesses.

If your application accesses a DB2 database, you can capture detailed SQL EXPLAIN analysis data, information about the table that is accessed, the SQL statements that are

executed, and SQL statement completion data. You can also display the text of the SQL statement that you want to analyze.

Application Monitor is easy to learn and to use. The user interface to Application Monitor is a standard ISPF dialog.

1.3 Debug Tool and Debug Tool Utilities & Advanced Functions

Debug Tool is IBM's interactive source-level debugging tool for compiled applications. It is a program testing and analysis aid that helps you examine, monitor, and control the execution of application programs written in C, C++, COBOL, or PL/I on a z/OS or OS/390 system. By using the disassembly view, Debug Tool is also able to provide support for programs compiled with the NOTEST compiler option, or applications that include other languages.

Debug Tool supports debugging of application programs that run in the following environments:

- CICS
- IMS
- DB2
- WebSphere
- TSO
- JES/batch
- UNIX System Services

You can use Debug Tool to debug your programs in batch mode, interactively in full-screen mode, or in remote debug mode.

Debug Tool Utilities and Advanced Functions assists the developer in the source-level debugging for compiled applications. Debug Tool now ships with Debug Tool Utilities and Advanced Functions, reducing ordering and installation complexities. Not only can Debug Tool Utilities and Advanced Functions help you debug Language Environment® (LE) enabled applications, integrated support has been expanded to include non-LE Assembler. IBM's latest versions of the Problem Determination tools continue the trend of protecting your tooling investments and maximizing IT productivity with affordable tooling.

1.4 Fault Analyzer for z/OS

Fault Analyzer assists the developer in quickly analyzing and fixing application and system failures.

It helps to determine the cause of failure and also provides assistance in how to resolve the problem.

Fault Analyzer supports different modes of operation:

- Real-time analysis at time of abend
- Batch reanalysis of faults in history file
- Interactive reanalysis of faults in history file under ISPF

Fault Analyzer also:

- Facilitates expansion of messages and abend codes applicable to the analysis result.
- Lets an application add its own application-specific messages and abend codes to supplement those supplied by IBM.

- Provides management of application abends. The fault history file helps track and manage application abends, fault reports, and associated dumps through an interactive display.
- Eliminates the need to recompile programs or change JCL to invoke Fault Analyzer.
- Applies the collective knowledge of many experts to each abend.
- Offers interactive reanalysis to provide programmers with the ability to increase the level of detail the analysis engine can provide after the abend has occurred.

1.5 File Manager for z/OS

File Manager provides comprehensive, user-friendly tools for working with OS/390 data sets, DB2 data, or IMS data. These tools include the familiar browse, edit, copy, and print utilities found in ISPF, enhanced to meet the needs of application developers.

File Manager utilizes COBOL or PL/I copybooks stored in partitioned data sets and external library management systems to define data set records more efficiently, or dynamically define your own record structure in a template.

File Manager allows users to:

- Browse, edit, copy, and compare data using copybooks or dynamic templates
- Find, edit, print, and display data quickly and easily to reduce development time
- Edit record and field selection criteria on the fly
- Leverage multiple selection criteria to make data more accessible
- Work with files containing multiple record structures
- Work with data in IAM data sets as well as data in QSAM and VSAM data sets
- Sort data during an edit or browse session
- Perform sophisticated data set comparisons, using the new Data Set Compare utility
- Use multiple search strings in the Find/Change utility
- Edit entire files, regardless of size
- Copy data between fields of different data type and length
- Enhance File Manager with the user's own custom REXX procedures (such as TALLY for tallying field values)
- Use the attributes of existing QSAM data sets to define attributes in new data sets
- Automate tasks in batch jobs, REXX procedures, or CLISTs

File Manager also contains DB2 and IMS components, sometimes called File Manager for DB2 Data and File Manager for IMS Data, hereafter referred as FM/DB2 and FM/IMS.

FM/DB2 is a powerful set of utility functions for editing, browsing, printing, copying, and maintaining DB2 data. It also provides utilities for:

- Listing DB2 objects
- Managing DB2 privileges
- Generating JCL to run DB2 standalone utilities
- Exporting and importing DB2 tables to or from QSAM or VSAM data sets
- Creating data to populate DB2 tables
- Prototyping SQL SELECT statements

FM/DB2 is an ISPF application program. It provides panels that you can use to select options and to specify parameters, commands, and program function keys to simplify requests for common functions, and full-screen format for information display and editing.

FM/IMS is an ISPF application for manipulating data stored in IMS databases. Using FM/IMS, you can:

- Display data from one or more IMS segment occurrences, and see their relationship to other segment types within the database

- Edit and update data in IMS segment occurrences
- Insert segment occurrences into an IMS database
- Delete segment occurrences from an IMS database
- Extract a subset of IMS data to a flat file
- Load data into IMS databases
- Print selected data or entire databases

For many tasks, you can use the FM/IMS elements called templates and views to define a logical view of a database, based upon field definitions from COBOL or PL/I copybooks. When you associate a view with a database you can:

- Format data according to segment layouts defined in COBOL or PL/1 copybooks
- Select the segment types and fields that you want to display, edit, or extract
- Change the order in which fields are displayed and adjust field headings
- Create criteria to identify and select the data that you want to use

FM/IMS provides you with a number of flexible ways to connect to your IMS databases:

- In BMP mode you can connect to an online multi-user database and manipulate the data.
- In DLI mode you can work with data offline as a single user or you can share the data with others.

In addition, FM/IMS provides two functions that you can use in batch jobs. FM/IMS Edit Batch (IEB) runs a REXX procedure that can insert, update, retrieve, delete, or print segments and create views. FM/IMS Batch Print (IPR) can print the entire database in one of several available display formats, or a selected sub-set of the database, based on a view.

1.6 Workload Simulator for z/OS and OS/390

Workload Simulator (WSim) is a terminal and network simulation tool. Workload Simulator enables users to perform the following functions:

- Function testing to ensure a new program or subsystem performs according to specifications. Test cases are prepared to verify that the code will execute the job it is designed to run.
- Regression testing to ensure a modified program or subsystem to ensure that the change has had no effect on the modified portion.
- Performance testing to ensure the system can handle the projected throughput with acceptable response times.

Used as a basic tool in a comprehensive test plan, WSim increases the effectiveness of system testing by providing a structured and systematic approach to all phases of system testing.

WSim enables users to test and evaluate teleprocessing systems without needing to have terminals and terminal operators present. WSim can be used to simulate the actions of a number of different applications and terminals. These simulated resources communicate with the real teleprocessing system as if they were physically present. No modifications to the system under test are required.

Workload Simulator includes Workload Simulator Test Manager, hereafter referred as WTM. WTM is a utility that guides the user through the test process. WTM can help users with the development and management of test cases, the automation of test runs, and the analysis of results. For even more flexibility, WTM offers three modes of operation that control the amount of interaction the user may observe with Workload Simulator.

The latest enhancements to the WSim, implemented in September 2004, include:

- WSim Adapters for Rational® TestManager, which allow existing WTM schedules to be launched from the Rational TestManager running on a remote workstation
- 3270 screen passwords masking in test scripts, logs, and reports

2

Overview of the problem determination and deployment tools

This chapter describes the basic functions of the tools that are part of IBM z/OS Problem Determination and Deployment Tools. This suite of products includes:

- Application Monitor for z/OS
- Debug Tool for z/OS and Debug Tool Utilities and Advanced Functions
- Fault Analyzer for z/OS
- File Manager for z/OS
- Workload Simulator for z/OS and OS/390

In this redbook, these products are referred to as:

- Application Monitor (AM)
- Debug Tool (DT) and Debug Tool Utilities and Advanced Functions (DTU&AF)
- Fault Analyzer (FA)
- File Manager (FM)
- Workload Simulator (WSim)

It is said that today's business customers and prospects are only a click away from your competitor's Web site. Users no longer tolerate poor response time for transactions. In addition, today's companies must be agile and move quickly to take advantage of the ever-changing landscape of new business technologies. Good response time and good turnaround time are key requirements for companies that expect to be leaders in this new paradigm. E-business demands more and more complex applications, driving a requirement for more sophisticated tooling.

As applications have become more complex, application development has become more and more expensive. The IBM z/OS Problem Determination and Deployment Tools comprise a comprehensive suite of powerful tools for improving the health of a company’s application portfolio and, at the same time, addressing the growing need for cost-effective application development and testing.

Using these tools, users can optimize their application development activities and quickly identify and resolve problems that occur in batch, IMS, DB2, CICS, and UNIX System Services (USS) applications. There are many features within this suite of tools that can help users perform day-to-day tasks such as:

- Recognizing, locating, and fixing application problems
- Working with z/OS data sets, DB2 data, or IMS data
- Conducting reliable tests on function and performance

2.1 Application Monitor for z/OS

Application Monitor is a non-intrusive performance measurement and analysis system that helps to resolve z/OS and OS/390 job performance issues, whether those jobs are applications, subsystems, or tasks. It provides performance statistics on any job you want to monitor. These statistics can be the current system data (online analysis) or data collected over a certain time period (historical analysis). By using Application Monitor, you can immediately focus your activities on tuning specific areas of an application, thereby improving productivity and meeting the challenges demanded by e-business applications.

2.1.1 Use of Application Monitor

The next sections discuss when an application programmer can use Application Monitor.

During application development

The most straightforward use of the profiler is in situations where your code is not meeting its intended performance targets. A profiler can be used to determine which routines are used the most, and what would benefit from examination for performance tuning.

The next obvious use is for determining the location of a “supposedly” infinite loop. A profiler such as Application Monitor can be used to actively monitor the application while it is running and “drill down” through the list of modules to the assembler instruction or the COBOL/PLI source line.

There is yet another use for a profiler during development. While programming an application, a programmer often uses code developed by another developer. This can leave a programmer dangerously lacking in information about the performance of the code. If two implementations are possible, a profiler can be used to determine which one will perform best.

During application test

Most application development projects have performance targets. A regimen of performance test scenarios is especially common. The profiler tool can be used to provide more information; for example, suppose that an application runs 25% slower. The tool could be used to determine which part of the application has degraded or if it was an overall degradation. Comparative reports could also be used to monitor changes in the application’s performance.

During application service

If a customer reports a problem with performance, a profiler is the perfect tool to help the service professional narrow down the cause and pinpoint the solution.

During application maintenance

It is a fact of modern development that the person who maintains a given application is not necessarily the person who wrote it. It is also true that the performance characteristics of an application can change during its service lifetime.

A profiler such as Application Monitor can help stop the performance loss that often occurs during maintenance. An application programmer can use this tool to determine where the hot spots or heavy loops are in the application. The information on where the performance bottleneck is located can then be taken into consideration when putting in a fix. For example, the developer could find out where the heavy loop is, and then do whatever is possible to avoid adding code to that loop when fixing the problem.

Application Monitor could also be used after a tentative fix is made that seems to degrade performance, to help find the cause of that degradation.

Other uses

These are some other uses for Application Monitor:

- To provide information to a software provider when you run into a performance problem with their code.
- To help system programmers determine what parts of a given application are causing a system performance problem.

2.1.2 What Application Monitor can do

Application Monitor is a performance measurement and analysis tool that:

- Optimizes the performance of existing application resources
- Provides maximum flexibility, with support for CICS, COBOL, DB2, IMS, and PL/I technologies
- Eliminates excessive I/O activity and CPU time to increase response times
- Improves response time of online transactions and batch turnaround times
- Identifies code bottlenecks during initial testing and isolates performance problems in existing applications

Application Monitor collects samples from the monitored address space and analyzes the system or resource application in the following categories:

- Total address space utilization of all modules in the address space
- CSECT utilization within each load module
- Instruction or statement utilization within each CSECT
- COBOL statements utilization within each COBOL module (or PL/I for a PL/I module)

The CICS support in Application Monitor allows you to:

- Monitor a region as a whole or display detailed information about an individual transaction
- For an individual transaction, drill down to the statement level and display detailed information about:
  - CICS Transactions
  - CICS, DB2 and DLI Calls in a Transaction
  - CICS Transaction Executions
  - CICS Calls in a Unit of Work

If the DB2 option is activated, Application Monitor analyzes the following DB2 categories for any application (batch or online) using DB2:

- SQL Analysis
- SQL Statement Summary

- SQL Statement View
- SQL Statement Full Text
- SQL Detailed Explain Analysis
- SQL Explain Detail
- SQL End Completion Statistics
- SQL Statement Completion
- DB2 Table Information

Application Monitor monitors IMS applications and collects data that indicates where IMS tasks are spending time. IBM Application Monitor only monitors IMS dependent regions. An IMS dependent region is defined as one where a job step executes an IMS program. These regions include batch, Batch Message Processing (BMP), Message Processing Program (MPP), and Fast Path (FP) regions. If a batch job calls IMS by attaching either DL/I or a BMP, Application Monitor monitors such a job as an MVS batch job and not as an IMS application. Application Monitor provides the following display panels for IMS message analysis:

- IMS Messages Processed by Region
- IMS Message Analysis
- IMS Message Call Analysis
- IMS Segment Search Criteria Analysis Subsequent sections describe these displays in detail.

2.1.3 The major functions of Application Monitor

Application Monitor for z/OS provides analysis information throughout the cycle of application design, development, and maintenance. It evaluates application prototypes in the design phase, reviews the impact of increased data volume or changes in business requirements on performance, and generates historical data and reports to analyze performance trends and evaluate program changes for cost effectiveness.

A high-level overview of function provided follows.

Online analysis

Provides dynamic displays of job performance. You can check job execution online without having to first record the data and then route it to display or print.

Deferred monitoring

When you do not know exactly when some jobs will begin executing, you can set up deferred requests. In the deferred request, you specify a start/end time frame for Application Monitor to watch for the job to become active. Deferred requests allow you to monitor jobs that run at night or at other times when you are not available to initiate a monitoring request online.

Historical analysis

Lets you view data collected during a previous analysis session. The data, collected over a period of time, resembles the information available in online analysis.The only difference is that the screens have a HISTORY banner across the top.

Delay analysis

Provides information about potential or existing bottlenecks that may impact job performance.

Data set analysis

Displays I/O allocations for the DD cards of a monitored jobname, VSAM data set information, and information on the local shared resource pools for the jobname and stepname being monitored.

DB2 analysis

Provides all the information that you need for DB2 performance analysis and tuning. In addition to CPU utilization by SQL statement, you see the EXPLAIN data and object analysis for tables and indexes.

Monitoring CICS

Application Monitor understands information associated with CICS Transaction Server regions. You can monitor a CICS region as a whole or display detailed information about a transaction that is running in the region.

Monitoring IMS

You can use Application Monitor to monitor the activity in IMS dependent regions, including batch, batch Message Processing (BMP), Message Processing Program (MPP), and Fast Path (FP) dependent regions. You can monitor an IMS region as a whole or select a particular message or batch program to analyze in detail.

Monitoring jobs

With Application Monitor you can monitor a collection of jobs for analysis together as a group. This is referred to as using a *named set*. For CICS you can define a named set for each possible TOR/AOR/FOR combination of regions and systems if the transactions you want to analyze can run in a multi-region environment. You can also define a named set for each MVS system in a sysplex where a batch application can execute. This way, you can have Application Monitor monitor the application without you having to specify the actual system in which the application job will be scheduled for execution.

Reports

Printed reports are useful for comparing job performance from a daily, weekly, or monthly perspective. All of the analysis data is available in printed reports.

2.2 Debug Tool for z/OS and Debug Tool Utilities and Advanced Functions

Debug Tool helps the developer to test programs and examine, monitor, and control the execution of programs written in assembler, C/C++, COBOL, or PL/I on a z/OS or OS/390 system.

Applications can also include other languages. For such applications Debug Tool provides a disassembly view that lets the developer debug, at the machine code level, the corresponding portions of applications. Of course, in the disassembly view, debugging capabilities are limited.

Debug Tool works in the following environments:

- DB2
- CICS
- TSO
- JES batch
- UNIX System Services in remote debug mode, or full-screen mode through a VTAM® terminal
- IMS

To use Debug Tool, an application must be compiled with the TEST compile-time option, link-edited, and then run with the TEST run-time option.

Debug Tool Utilities and Advanced Functions enhances Debug Tool, and the combined strength of these products can help the user debug and examine programs.

2.2.1 What Debug Tool can do

An overview of how Debug Tool can assist the user is presented in the following sections.

Source-level debugging

Monitor source code for C/C++, or a full compiler listing for COBOL and PL/I. Compiler listings are an expanded source listing that, in addition to program source, contain copy book source for COBOL or include file source for PL/I.

Mixed-language applications debugging

Debug Tool supports debugging of mixed-language applications within the same session, and recognizes when the current programming language changes (see dynamic patching below). Language modules not supported by Debug Tool, such as assembler, are tolerated, but no debugging support is provided for these modules.

Dynamic patching

For each supported programming language, there is a set of interpreted commands that you can use to specify actions to be taken. These commands are subsets of the programming languages, so they are easy to learn, and allow you to make adjustments to your application while you debug it.

The user can use the commands to alter the value of variables and structures, and to control the flow of an application. For example, a programmer can declare a new variable and use the variable to patch a program as it executes.

Dynamic breakpoint setting

The user can set breakpoints in an application program, monitor variables for changes, and watch for specified exceptions and conditions during program execution. For example, the user can cause an application to break when a specific variable or location in storage is changed. The user can set, change, and remove breakpoints when going through the application. There is no need to know where to break before the start.

Single-step debugging

To focus on a problem area, the user can step line by line through the execution of an application. For example, when an application stops for a breakpoint, the user can examine each line that follows. Single-step debugging, along with the ability to set dynamic breakpoints, allows the user to monitor, interrupt, and continue through the flow of the program to identify errors easily.

Program frequency counting

Debug Tool counts how many times a statement or verb has been processed in an application program. This allows the user to verify the coverage of code paths.

Program information gathering

Debug Tool can display program and environment information enabling the user to display, monitor, and alter program variables or storage in real time. The user can also check how the application was compiled, and look at its structure.

Session logging

Each debug session can be recorded in a log file for reviewing, editing, or replaying. This allows the user to replay the actions taken in a session to pinpoint errors in an application.

General testing

The user can also use Debug Tool as a test tool. By using the session logging feature as code is being debugged, the user can save the results of the session for use as input to a future Debug Tool session. As changes are made to the code, the saved log file can be used as input to Debug Tool in order to verify that no unexpected behavior occurs as a result of these changes. Session logging allows the user to create suites of regression test cases that can be used to minimize the number of new bugs introduced during the normal application development process.

2.2.2 What Debug Tool provides

Debug Tool can be started in the following ways:

- The application can start Debug Tool when it terminates normally, abends, or generates errors or conditions above a chosen severity, if the pertinent suboptions are specified on the TEST run-time option.
- The application can start Debug Tool directly using a library service call such as CEETEST, PLITEST, or the ctest() function.
- For CICS applications, Debug Tool can be used in one of the following ways:
  - Single-terminal mode: A single 3270 session for Debug Tool and the application; when the application is running Debug Tool is hidden, and vice versa.
  - Dual-terminal mode: One 3270 session is used to display the application, and the other is used for Debug Tool.

Debug Tool enables the user to perform interactive or batch debugging:

- Using the full-screen interface, the user can interactively debug an application as it runs (including batch applications).
- The user can also debug batch applications with Debug Tool in batch mode, using a predefined command file.

The full-screen interface of the Debug Tool is divided into three windows:

Monitor window Displays status of items chosen to be monitored, such as variables, registers, programs, the execution environment, and Debug Tool settings. For example, this window is used to watch how the content of variables changes during program execution.

Source window Displays the program source, with the current statement highlighted. The prefix area at the left of this window is used to enter commands to set, display, and remove breakpoints.

Log window Records and displays user interactions with Debug Tool and, optionally, shows program output. This window contains the same information as the log file.

Debug Tool provides a tool called Debug Tool Setup Utility (DTSU) to help create and manage its setup files.

Debug Tool has an extensive set of commands to debug programs.

2.2.3 Debug Tool Utilities and Advanced Functions

Debug Tool Utilities and Advanced Functions adds tools to help the user perform the following tasks:

- Prepare high-level language programs for debugging by helping convert, compile, and link.
- Prepare assembler programs for debugging by helping assemble, create debug information, and link.
- Conduct analysis on test cases to determine how thoroughly test cases test programs (also called code coverage).
- Start and run a program in foreground or batch by storing and using setup information. Setup information can be the run-time parameters, libraries, and names of input and output data sets.
- For IMS Version 8, browse and edit the Language Environment run-time parameters table.
- Create a batch job for a private IMS message region with customized load libraries and region attributes.
- Convert old COBOL source code and copybooks to new versions of COBOL by using COBOL and CICS Command Level Conversion Aid (CCCA).

The combination of DTSU and these tools is called Debug Tool Utilities.

Debug Tool Utilities and Advanced Functions enhances the set of Debug Tool commands by adding more than 15 additional commands, such as **`allocate`**, **`free`**, **`query automonitor`**, and others.

2.3 Fault Analyzer for z/OS

In today's fast-paced environment, application developers have to work quickly, with as few errors as possible, to keep up with increasing business demands.

The slightest application or system failure can jeopardize goals crucial to business success. To meet these challenges, users need greater visibility into application events to help analyze and resolve problems before they can pose a serious threat to business objectives.

IBM Fault Analyzer, Version 5.1 is a robust problem determination tool that helps ascertain why applications fail.

By gathering information about an application and its environment at the time of failure, Fault Analyzer software can help users repair failures quickly, develop and test new and existing applications more productively, and reduce costs along the way. It also analyzes IBM CICS system failures, and can help analyze IBM WebSphere Application Server for z/OS, Version 4 and Version 5 system failures.

2.3.1 Locate application failures quickly to reduce development costs

When an application abend occurs, Fault Analyzer captures and analyzes real-time information about the application and its environment, then generates an analysis report detailing the cause of the failure.

The report describes the failure in terms of the application code, so developers no longer lose time reviewing cumbersome, low-level system error messages. Fault Analyzer allows the user to choose a report format to locate the information more easily.

Each application abend is recorded by Fault Analyzer in a fault-history file by job name, failure code, and other details, along with the analysis report and storage pages referenced during the analysis. This information can later be retrieved to reanalyze the failure, helping to speed future application development.

You can logically group fault-history files, and restrict each user's access to particular groups. You can also transfer fault records between history files for analysis on remote systems.

Designed for ease of use, Fault Analyzer includes soft-copy versions of selected manuals from the IBM OS/390 Online Library. It extracts message and failure code descriptions from these manuals and inserts them into the analysis report where applicable, freeing your developers from researching the explanation of message and failure details. And to maximize flexibility, Fault Analyzer allows you to specify criteria to exclude particular jobs from problem determination.

You can also provide your own descriptions for messages (including those issued by your own applications). Or you can write your own user exits in a variety of languages, including assembler, C, COBOL, PL/I or Restructured Extended Executor (REXX). For example, you can write user exits to:

- Override Fault Analyzer options
- Access compiler listings that are compressed or only available through a proprietary access method
- Provide custom explanations for particular message IDs to be included in the analysis report when these messages occur
- Notify specific people about an application or system failure

2.3.2 Maximizing development assets

Fault Analyzer starts only after an application abend occurs. Having said that, you won't waste valuable processing resources during normal application execution.

You can choose a different storage-display concept, based on the total size of eligible storage involved.

This feature can be used to prevent the generation of potentially extremely large reports in situations where, for example, a COBOL program has a very large working storage area. You can also opt to delete fault-history file entries based on the percentage of space used.

Fault Analyzer does not require the applications to be recompiled. You can choose to produce a side file that identifies the failure source statement. In the absence of a side file, Fault Analyzer uses the compiler listing. If you choose to keep Fault Analyzer side files instead of compiler listings, you can produce a readable listing file from a side file. Alternatively, you can request prompting for only the point-of-failure side file during interactive reanalysis.

2.3.3 Analyze application failures to improve developer productivity

Fault Analyzer provides three modes to help you better track and analyze application and system failure information. They are defined in this section.

Real-time analysis

Fault Analyzer software includes exit programs for CICS, IBM Language Environment, and OS/390 systems, which it adds to the normal failure processing for these environments.

When an application failure occurs in any of these environments, the Fault Analyzer exit starts real-time analysis. After failure processing, you can view the analysis report in the job output or through the Fault Analyzer Interactive System Productivity Facility (ISPF) interface.

Batch reanalysis

The Fault Analyzer batch reanalysis mode generates a new analysis report. This report is based on the dump and information gathered in real time, but with potentially different options specified, or with compiler listings or side files made available. You can submit a Fault Analyzer batch mode job using either the Fault Analyzer full-screen ISPF interface or your own job control language (JCL).

Interactive reanalysis

This Fault Analyzer mode lets you view working storage and control blocks at the time the dump was written. The interactive reanalysis runs under ISPF and enables you to navigate on screen through a formatted, structured view of a fully detailed reanalysis.

The ISPF interface has many point-and-click fields for easy navigation through the interactive reports. For example, when viewing an analysis report on screen in interactive mode, you can cursor-select a storage address to view the contents of storage at the time of the dump.

You can also associate addresses with notes that might assist with solving a problem. The notes are saved in the fault entry and are automatically displayed whenever the storage for the associated address is displayed.

During interactive reanalysis, the FIND command can be used to locate data anywhere in storage that was saved during real-time analysis. This storage display is not limited to the currently displayed page.

An interactive problem control system (IPCS) equivalent RUNCHAIN command is available from within the interactive report for ease of displaying chained-data areas.

2.3.4 Fault Analyzer functions to optimize efficiencies

Among the functions of Fault Analyzer that can optimize efficiencies are the following:

- Integration of WebSphere and Java™ information into the standard interactive reanalysis report in the system-wide information section
- Enhancements to CICS system dump-analysis performance and usability:
  - New domains added
  - LE heap storage analysis added
  - New fast-path commands for easier navigation between CICS domains
  - CFA transaction improvements, including a feature that permits SDUMP screening to improve performance when capturing SVC dumps, and improved control over the operation of Fault Analyzer in a CICS region

- Improved CICS transaction abend analysis
  - Recovery manager files and temporary storage added
  - Ability to select trace entries interactively added
  - Other CICS trace entries now available
- Enhanced linkage to IBM File Manager
  - The ability to browse for VSAM files involved in an abend from the interactive report with File Manager for z/OS at the file record active at the time of the abend
  - Use of the current template for VSAM or QSAM data sets
- Support for Formatting User Exit
  - A new user exit type that permits users to add a separate section to the report containing information that might be specific to their environments. Enables REXX exits of this type to be executed on demand in the interactive report.
- Improved user interface
  - An option added to the interactive report services pull-down menu to list all user notes that exist in the current fault entry
  - A new ISPF interface command, STCK, to convert a binary STORE CLOCK value to human-readable date and time
  - A new ISPF interface command, DISASM, to perform disassembly of object code at a given address in storage
  - The ability to establish a view-specific Fault Entry List display column configuration and initial fault entry selection
  - The ability to specify a user job card for batch reanalysis jobs, without needing to edit the generated JCL, through a new option on the Batch Reanalysis Options display
- Improved fault report function
  - Improved abend-specific diagnostics for many common abends
  - More complete information about where modules were loaded from
  - Improved event summary to provide better information about the location of an event
  - Enhanced formatting of parameters for the last IMS call
- Improved product administration
  - Support for renamed COBOL SYSDEBUG files added through the IBM Debug Tool for z/OS EQAUEDAT exit
  - Support for substitution symbols in IDIVIEWS data set names either Time Sharing Option (TSO) user ID or TSO profile prefix
- User-specific reporting or statistical analysis of history file contents by providing mapping information of the $$INDEX member
  - Enhancements to the IDILANGP utility to read COBOL SYSDEBUG files
- Improved serviceability
  - Changed location for message IDI0047S, allowing SLIP MSGID capture of Fault Analyzer internal abends with an SVC dump
  - New function to detect Fault Analyzer real-time execution exceeding a dynamically calculated maximum execution time, and if so, issuing message IDI0092S (which allows SLIP MSGID capture to take an SVC dump) and cancelling the analysis
  - Fault Analyzer service level now visible in the LE traceback

2.4 File Manager for z/OS

IBM File Manager for z/OS software provides comprehensive, user-friendly tools to help you work more efficiently with IBM z/OS data sets, IBM DB2 data or IBM IMS data. Building on the standard browse, edit, copy, and print utilities of IBM Interactive System Productivity Facility (ISPF), File Manager includes enhancements to meet the needs of application developers working with structured data files.

The basic features of the File Manager editor and the ISPF/PDF editor are virtually identical, allowing you to take advantage of extra features—without having to relearn basic skills.

File Manager consists of three components. These components are:

- **Base component** for managing z/OS data sets, including queued sequential access method (QSAM) data sets, virtual storage access method (VSAM) data sets and partitioned data sets (PDS)
- **DB2 component** for managing DB2 data
- **IMS component** for managing IMS data

2.4.1 Enhance development productivity through structured data support

Typically, records in application data sets have a well-defined structure consisting of many fields. Each field can contain a discrete item of data, such as a person's name, a product code, or a currency amount. To work efficiently with these data sets, you need utilities that recognize the record structure, so you can selectively process individual fields. To maximize flexibility, each of the File Manager components allows you to identify your record structure in ways suited to the data type.

2.4.2 Maximize flexibility with the File Manager base component

The File Manager base component helps speed the application development process by identifying the structure of your records and displaying each field according to its data type. This component supports QSAM, VSAM, and indexed access method (IAM) data sets, including support for double-byte character set (DBCS) data in these data sets. You can edit entire files (regardless of size) and sort data during an edit or browse session. You can browse and edit VSAM alternate index components, as well as browse VSAM clusters via a path.

Using templates, you can select the records you want based on field values, find and change data in individual fields, display fields with headings showing name, data type and length, then print or copy only the fields you need. You can quickly and easily work with files containing multiple record structures and copy data between fields of different data types and lengths. In the base component, your templates can be generated from a copybook (or from multiple copybooks) that contains either COBOL data description entries or PL/I declare statements, or you can define your own fields in a dynamic template. You can display or print a copybook or base template. You can then add record-selection criteria and other formatting information and save the templates for reuse with different data sets that have the same record structure or for reuse with different File Manager utilities.

For example, while browsing through data sets, you can create, refine, and save a template that displays records meeting certain criteria, such as records with particular field values. Later, you can reuse that template in the File Manager Copy utility to extract the records that meet those criteria and copy them to another data set. The base component processes most record identification and selection criteria internally, while providing fast access to Restructured Extended Executor (REXX) for complex criteria statements. The Copy utility can

also be used to copy a series of PDS members according to a user-specified member name range, or to copy PDS members conditionally, based on some condition being true in the content of the member. The Copy utility allows you to rename PDS members according to a user-specified mask.

The Copy and Print utilities use the IBM DFSORT™ utility for supported data sets to help you gain significant performance improvements. Support is also available for DFSORT statements. The Copy utility also offers various PDS-member copying capabilities—including the ability to copy members based on finding a particular condition to be true from records in the member. The Find/Change utility offers flexible options to provide increased efficiency for batch and online modes. With the Data Set Compare utility, you can use a number of different synchronization types. You can also enhance File Manager with your own custom REXX procedures (such as tallying field values) and automate tasks in batch jobs, REXX procedures, or CLISTs. A high-performance subset of REXX and File Manager REXX functions, called FASTREXX, is also available. The FASTREXX subset is broad enough to handle most normal processing tasks.

The File Manager base component also includes:

- **Tape utilities**
  - Browse physical records on a tape
  - Copy tape files, optionally changing the block size and record format
  - Copy a logical volume from an exported stacked volume to a physical tape volume
  - Update tape records on screen, while copying the records from one tape to another
  - Compare two tapes byte by byte
  - Bypass tape blocks that include data-check errors
  - Position a tape at a particular file, or record within a file
  - Summarize the contents of a tape
- **Catalog services**
  - List or print catalog information
  - Work with catalog entries using user-friendly panels
  - Allocation of non-VSAM data sets
- **OAM utilities**
  - List, browse, print, update, and erase object access method (OAM) objects
  - Copy, back up, and restore OAM objects
- **Disk utilities**
  - List data sets on a disk while editing data on a specific track
  - Identify the disk extents of a data set
  - Search for data within a disk extent
- **Load module utility**
  - Name of the compiler
  - Name of utility that created load module
  - Multiple date formats

2.4.3 Manage data more efficiently with the File Manager DB2 component

The File Manager DB2 component delivers the commands and behavior of the ISPF/PDF editor and extends the capabilities to DB2 data. You can view detailed information about DB2 objects, show the relationship between them, and select the objects you want to work with from virtually any connected location or subsystem.

With the DB2 Object List utility, you can selectively list DB2 objects (for example, by owner), then invoke commands against those objects. The File Manager DB2 component also enables you to view and change DB2 privileges through a full-screen interface, so you can eliminate the need to code SQL statements to list and change DB2 privileges.

The interactive SQL SELECT Prototyping utility in the File Manager DB2 component helps you build a valid SELECT statement on screen. You can interactively test, analyze, and execute SQL statements. When you run the SQL statement, you can browse or edit the formatted result table and then save the statement for use in your application code.

The DB2 component features a sort command that enables you to easily rearrange data in an edit or browse session, to export selected rows and columns to QSAM or VSAM data sets, and to generate templates based upon your export selections. You can also generate job control language (JCL) for the most commonly used DB2 utilities to save time.

Edit/Browse utilities allow specification of the number of rows to edit or browse.

The File Manager V5 DB2 component also provides support of DB2 V8 new long names and DB2 V7 online utilities, and it supports synonyms in DB2 functions such as edit and browse.

2.4.4 Develop faster, less costly applications with the File Manager IMS component

Using record structures defined in COBOL or PL/I copybooks, the File Manager IMS component lets you edit IMS segments displayed as individual fields, formatted according to their data types. You can find and change data across an entire database hierarchy or in specified segments and fields. You can run a REXX procedure that will edit segments in an IMS database. The support for tabular display of data helps improve the display of segments within the hierarchy and offers significantly improved response times. Navigation commands allow you to specify segment operands, simplifying navigation between different segment types. You can also initialize and insert the first segment into an empty database. The File Manager IMS component helps minimize resources held during user screen interactions to maximize concurrent access to data.

You can use flexible criteria to select IMS database segments containing specific field values, and extract the segments into a sequential data set for later use in reports. Or you can use the IMS component to load them into a different database. For example, you might create a small test database using data extracted from a large production database. Through the IMS component, you can access databases using Data Manipulation Language 1(DL/I) or use batch message processing (BMP) to access databases that are concurrently being used in IMS Transaction Manager, DBCTL, or CICS environments.

2.4.5 Flexible file processing to meet your business goals

File Manager is designed to help increase productivity and lower costs as you build and test new and existing applications. You can use it to quickly create, edit, print, and reformat data files and to build new applications easily.

With File Manager, you'll gain the flexibility you need to build, leverage, and extend your assets to meet your business goals.

2.4.6 For more information

To learn more about IBM File Manager for z/OS, visit:

http://www-306.ibm.com/software/awdtools/filemanager/

2.5 Workload Simulator for z/OS and OS/390

Workload Simulator (WSim) is a terminal and network simulation tool that can help you test your networks and determine if your infrastructure is capable of handling expected additional workload. It provides the ability for the simulation of terminals and associated messages, including provision for you to alter message loads during a run.

WSim has evolved from the IBM TeleProcessing Network Simulator (TPNS) product. The addition of an interactive ISPF-based Test Manager (WTM) can significantly assist users in the development and management of test cases, automation of test runs, and analysis of results. At the same time, support for obsolete and little-used functions has been removed.

The IBM WebSphere Studio Workload Simulator (WSWS) tool can be used to provide similar functionality for browser-based infrastructures.

2.5.1 What Workload Simulator can do

Workload Simulator enables users to perform the following functions:

- Functional testing to ensure that a new program or subsystem is performing in accordance with specification
- Regression testing to ensure that a modified program or subsystem is still performing in accordance with original specifications where no changes were required
- Performance, stress, and capacity testing to ensure that a system can handle the projected load with acceptable response times

WSim can simulate three types of resources:

- System Network Architecture (SNA) logical units running as Virtual Telecommunications Access Method (VTAM) applications
- Common Programming Interface for Communications transaction programs
- Transmission Control Protocol/Internet Protocol (TCP/IP) clients using Telnet 3270, 3270E, 5250, Network Virtual Terminal, File Transfer Protocol, or simple TCP protocols attached to TCP/IP network via the IBM TCP/IP for Multiple Virtual Storage product

Both the network to be simulated and the system to be used to run WSim must be configured before testing.

A physical configuration is a configuration of the real system used to run WSim. There are two basic physical configurations:

- VTAM and CPI-C application configuration
- TCP/IP application configuration

A logical configuration is a configuration of the network containing the resources to be simulated by WSim and the real system to be tested. There are three basic logical configurations:

- VTAM application configuration
- CPI-C application configuration
- TCP/IP application configuration

When the system is configured, the network to be simulated should be defined, which is done by creating a script. Scripts contain two parts:

- Network definition statements describing resources to be simulated by WSim
- Message generation decks defining messages to be sent by the simulated resources to the system under test

WSim provides several methods to create message generation decks:

- Write statements directly
- Write programs in Structured Translator Language (STL)
- Use one of the script generating utilities provided with WSim

Depending on the user's familiarity with WSim and characteristics of the system under test, one, several, or all of the methods might be used when preparing real tests.

WSim provides several online and printed reports to analyze test results. Some reports are provided by default; others must be specifically requested. The reports fall into the following four general categories:

- Operator reports that indicate what is happening during operation
- The complete message log
- Reports based on the message log and generated by several WSim utilities
- Online response time statistics

2.5.2 What Workload Simulator provides

Workload Simulator offers the facilities for flexible generation of scripts, runtime support, and post-test analysis. It also provides the Test Manager, a convenient usability enhancement to help the users to go through the test process.

Flexible generation of scripts

WSim offers several options for creating scripts to use in simulations:

- Structured Translator Language (STL), a high-level REXX-like language
- Interactive Data Capture (IDC)
- Network Performance Monitor/LU 2 Reformatter
- Script generator

Runtime support

WSim offers various utilities for users to use in testing their networks:

- An ISPF interface that helps you set up and start the application and its utilities
- A Display Monitor Facility that shows simulated 3270 screen images and the data streams sent and received

- Run time reports
- ITPECHO, a VTAM application that echoes received data

Post-test analysis

WSim provides the following utilities to analyze test results:

- Loglist for formatting simulation run logs
- Log Compare to display records from two simulation runs with differences listed
- Response time to print a report with response time retrieved from logs

WSim Test Manager

The WSim Test Manager (WTM) is a usability feature that provides guidance through the test process. WTM offers selectable modes of operation, test management services, automatic script generation, and task automation. WTM guides the user through the test process and helps to develop and manage test cases, automate test runs, and analyze results. Use of WTM is highly recommended at least until a certain level of familiarity with WSim and scripts is achieved.

2.5.3 The latest enhancements

The latest enhancements to the WSim, implemented in September 2004, include:

- WSim Adapters for Rational TestManager, which allow existing WTM schedules to be launched from the Rational TestManager running on a remote workstation
- 3270 screen password masking in test scripts, logs, and reports

Part 2

Introduction to Problem Determination and Deployment Tools

In this part we provide a detailed introduction to each of the Problem Determination and Deployment Tools.

3

Introduction to Application Monitor for z/OS

This chapter describes the main functions provided by Application Monitor for z/OS, hereafter referred to as Application Monitor or simply AM. These functions can be used safely in both test and production environments. Application Monitor uses a low-impact sampling technology so it runs with very low processor overhead. It contains an online, instantaneous monitoring and measurement function for real-time bottleneck analysis, as well as a project oriented performance management functionality. It provides you with responsive and speedy information needed to make your applications deliver the best performance possible.

In this chapter, the following functions are discussed:

- On line analysis
- Program module reports
- Delay analysis reports
- Data set analysis reports
- DB2 analysis
- Monitoring CICS
- Monitoring IMS
- Monitoring jobs
- Deferred monitoring request
- History data set

Note: This redbook describes Application Monitor Version 2 Release 1 plus the following PTFs:

- UQ83326
- UQ85391
- UQ86637
- UQ87221
- UQ88081
- UQ89665
- UQ90570
- UQ92317
- UQ92540
- UQ92901
- UQ94341

3.1 How Application Monitor samples an application

Application Monitor is a sampling profiler. That is, it periodically samples the monitored program. This means that the profile it gives *is not* an “exact” view of the program (that is, found by following the code exactly and seeing all that is executed) but an “average.” To get the most out of a sampling profiler, a user should ensure that they are basing their analysis on a significant number of samples.

If an application programmer needs to follow an application step by step, Debug Tool for z/OS should be used instead.

3.2 How to get help

Panel-level help is available for all Application Monitor input and output displays.

You can display the help associated with a particular panel or display by pressing PF1 while you are on that panel or output display. All input entry fields and output display fields are described in detail on the associated panel-level help, as are all available actions.

3.3 The functions of Application Monitor

This section describes the main functions provided by Application Monitor.

3.3.1 Online analysis

Online (or real time) analysis monitors an active job in real time as the job executes, using ISPF as the interface. An active job is in memory, is waiting to be dispatched, or is currently executing.

When you monitor job performance in real time, the analysis data you see on Application Monitor displays is refreshed each time you press Enter. As an alternative, you can enter a command to request that displays be refreshed at a specified interval.

The analysis data is also written to a history data set so you can view the data later at your own pace.

To select or identify the job you want to analyze do one of the following:

- Display all the active jobs in the systems where Application Monitor is running and then select one to monitor.
- Type in the details (such as job name and the system where it is running) for the job you want to analyze.
- Select a set of jobs (for example, if you need to track a CICS transaction that can execute on multiple CICS regions).

Once you have selected your job to analyze, Application Monitor starts collecting samples and presents you with reports to view.

3.3.2 Program module reports

Application Monitor provides the following displays about the program modules for the target job step:

- CPU and Loadmod Analysis
- CSECT Utilization Analysis
- Statement Analysis (for COBOL or PL/I programs)
- Detail Statement Display v Instruction Analysis
- Detail Instruction Display

These displays are described in detail in Part 3, “Application Monitor for z/OS” on page 95.

3.3.3 Delay analysis reports

Delay analysis reports show where any waits in the application occur. Application Monitor displays and identifies the reason the programs associated with the target job are not executing (are in a wait state). The display has a line item for each wait reason. In some cases the delays are in system modules that may be hard to connect to a specific portion of an application.

These displays are described in detail in Part 3, “Application Monitor for z/OS” on page 95.

3.3.4 Data set analysis reports

The Data Set Analysis Report provides information about the data sets that the target job step accesses. Application Monitor shows data set attributes, access method, and any EXCPs that were encountered.

For VSAM data sets, two other reports are available:

- Detailed information about the target VSAM data set. For both data and index portions of the VSAM cluster, Application Monitor gives details such as free space, splits, control interval size, I/O.
- Detailed information about the VSAM Local Shared Resource pool, such as look aside activity, number and type of the buffers.

3.3.5 DB2 analysis

The SQL Analysis display is the first display that is available for DB2 analysis. It identifies the application plan and Database Request Modules (DBRMs) and provides information about the SQL statements that are executed in the target job.

The SQL Statement displays show the full text (or a summary of the text) of the target SQL statement. If a DECLARE CURSOR statement is associated with the target SQL statement, the first line of the text of the DECLARE CURSOR statement is also shown.

The SQL Detailed Explain Analysis display provides information from the plan table that is created to hold EXPLAIN output.

The DB2 Table Information display provides information about the DB2 table that the target SQL statement is accessing.

The SQL End Completion Statistics display shows a line item for each execution of each statement.

The SQL Statement Completion display shows the completion data about SQL statement execution that is returned to the program in the SQL Communication Area (SQLCA).There are two versions of this display:

- If the target application calls program DSNTIAR to interpret the SQL return code issued after the target SQL statement has executed, DSNTxxxx messages are given along with the contents of the SQLCA data items (SQLCODE, SQLSTATE, SQLERRD, and so on).
- If the target application does not call program DSNTIAR to interpret the SQL return code issued after the target SQL statement has executed, DSNTxxxs messages are not displayed.

3.3.6 Monitoring CICS

The CICS Transactions display lists all the transactions that are or have been active in the target CICS region during the displayed monitoring interval.

The CICS Calls in a Transaction display shows the elapsed wall-clock time that all the instances of a CICS or SQL or DLI call from the same location in the program spent in application code before the call was issued and spent in CICS or DB2 or IMS code to execute the call's function.

The CICS Calls in a Unit of Work display provides information about all the calls processed in the target transaction instance (unit of work).

The CICS Transaction Executions display provides information about each instance of the target transaction that is active during the displayed monitoring interval.

3.3.7 Monitoring IMS

The IMS Messages Processed by Region display shows all messages executing in the selected region will be displayed.

The IMS Message Analysis display has two parts:

- The fixed area summarizes all instances of the target message (or Jobname if you are analyzing an IMS batch job, meaning a non-message-processing job).
- The scrollable area of the display summarizes, as one entry, all calls with the same operands from the same location in the application. The data on the scrollable area of the panel is ordered by Program.

The IMS Message Call Analysis display shows all calls made from the target program during the displayed monitoring interval.

3.3.8 Monitoring jobs

One useful application of the named set feature is to track a CICS transaction that can execute on multiple CICS regions. Application Monitor requires that you specify the system in which a CICS transaction will execute if you want to analyze its performance. Often, a CICS transaction starts in a Terminal Owning Region (TOR) and may be dispatched to other regions (for example, AORs and FORs) during its execution. Since you may not know in advance the regions that will actually process the transaction, you can create a named set with an entry for each region in which the transaction could execute. You can launch the named set and Application Monitor will monitor the regions named and collect performance data about the target transaction when it becomes active in those regions.

All the jobs identified in the named set must be active when you launch it to initiate real-time monitoring. If any job is not active, Application Monitor displays a message and does not process your monitoring request.

Typically, the requirement for all jobs in a named set to be active is not an issue if the named set identifies a collection of CICS regions, because these regions are usually always active.

However, if you try to monitor a named set that identifies a collection of user applications that are submitted for execution as batch jobs, all jobs may not be active when you launch the named set for analysis. In this case, you should set up a deferred request to monitor the named set. With a deferred request, all jobs in the named set do not have to be active at the same time. Application Monitor begins watching for each job in the named set to become active to monitor the job.

3.3.9 Deferred monitoring request

You cannot always predict exactly when a job will begin executing so you can analyze it in real time. Therefore, you set up deferred requests for such jobs. In a deferred request, Application Monitor can be set up to “sleep” until the job that is to be monitored begins execution; Application Monitor will “wake up” and will then write performance records out to disk to be processed to detect where the bottlenecks occur in the specified jobs. In this manner, the analyst does not have to be present to physically watch over the monitored jobs while still obtaining the information needed to solve even the most complex performance problems.

You can set up a deferred request for a specific job or for a named set. In the request, you specify:

- A Start/Stop time frame for Application Monitor to watch for the job or jobs to become active.
- The maximum number of minutes (Max Duration) to monitor the job after it becomes active.
- The maximum number of samples (Max Samples) to be taken during the monitoring session.

If the job (or a job in a named set) becomes active within the specified Start/Stop time frame, Application Monitor monitors the job for Max Duration, until Max Samples have been taken, or until the job completes, whichever occurs first.

Application Monitor stores the data that it captures during the monitoring session in a history data set. You can access this data set later to review the analysis data. Once you have set up one or more deferred requests, you can edit or delete them.

If a single job (or a job in a named set) does not become active during the Start/Stop time interval, Application Monitor does not monitor that job. Therefore, be sure to estimate the Start/Stop time period carefully and specify a period that is large enough to accommodate a reasonable deviation from your estimate of when a job (or jobs in a named set) will start and stop executing.

3.3.10 History data set

Application Monitor can create a history data set for:

- Each job that you monitor in real time
- Each named set that you monitor in real time
- Each completed deferred request that you scheduled for a particular job or named set

History data sets are named according to your settings (see Chapter 8, “Application Monitor: Settings and customization” on page 97). You can analyze the contents of a history data set in one of two ways:

- Online analysis

  When you analyze the contents of the history data set online, you can specify the time range that you want to analyze as well as the length of the sampling interval over which you want to view the data.

- Batch analysis

  When you print batch reports, the printed data reflects the analysis data that is collected over the entire monitoring interval for the target job step or transaction. You cannot specify or change the length of the sampling interval over which you want to view the data.

4

Introduction to Debug Tool and Debug Tool Utilities and Advanced Functions

Debug Tool lets application programmers trace through an application program to determine if and where any errors exist and to identify areas of potential problems.

Available interfaces, program compile, link-edit and runtime options requirements, and frequently used Debug Tool commands are described in this chapter.

4.1 Debug Tool

Debug Tool helps the developer to test programs and examine, monitor, and control the execution of programs written in assembler, C/C++, COBOL, or PL/I on a z/OS or OS/390 system.

Applications can also include other languages. For such applications, Debug Tool provides a disassembly view that lets the user debug, at the machine code level, the corresponding portions of applications. Of course, in the disassembly view, debugging capabilities are limited.

Debug Tool can be used to debug programs in a batch mode, interactively in a full-screen mode, or in a remote debug mode. The latter mode cannot be used with some compilers, for example, PL/I for MVS and VM.

You must use the correct compiler options to be able to use Debug Tool.

4.1.1 Debug Tool interfaces

The terms batch mode, remote debug mode, and full-screen mode identify the types of debugging interfaces provided by Debug Tool.

Batch mode

To run Debug Tool in batch mode, a command file is prepared in advance, with a predefined series of Debug Tool commands which will be performed on a running batch application.

No terminal input or user interactive intervention is possible in a batch debugging session.

The results are saved in a log data set and can be reviewed and analyzed when a batch debugging session is finished.

There are several ways to define the TEST runtime option with the commands file specified, including:

- In the PARM parameter of the JCL EXEC statement
- Using PLIXOPT string in the PL/I source application
- Using CEEUOPTCSECT linked with an application

Remote debug mode

In remote debug mode, the host application starts Debug Tool, which uses a TCP/IP connection to communicate with a remote debugger on your Windows® workstation. Not all compilers are compatible with this mode.

Debug Tool, in conjunction with a remote debugger, provides users with the ability to debug host programs, including batch programs, through a graphical user interface (GUI) on the workstation.

The following remote debuggers are available:

- Compiled Language Debugger component of WebSphere Studio Enterprise Developer

  This remote debugger is the recommended choice since it offers more functionality than the IBM Distributed Debugger.

- IBM Distributed Debugger

  This remote debugger is available through several products, for example:

  - OS/390 C/C++ Productivity Tools
  - VisualAge® for Windows family of products

Both remote debuggers run on Windows NT® 4.0, Windows 2000, and Windows XP.

WebSphere Studio Enterprise Developer Debugger

This debugger is available as part of the WebSphere SDK. On entry to the SDK, there is a small box on the left-hand side of the screen that looks like a divided screen. This box is titled Open a Perspective. To debug, pick the small bug icon and the screen will reformat as shown in Figure 4-1.

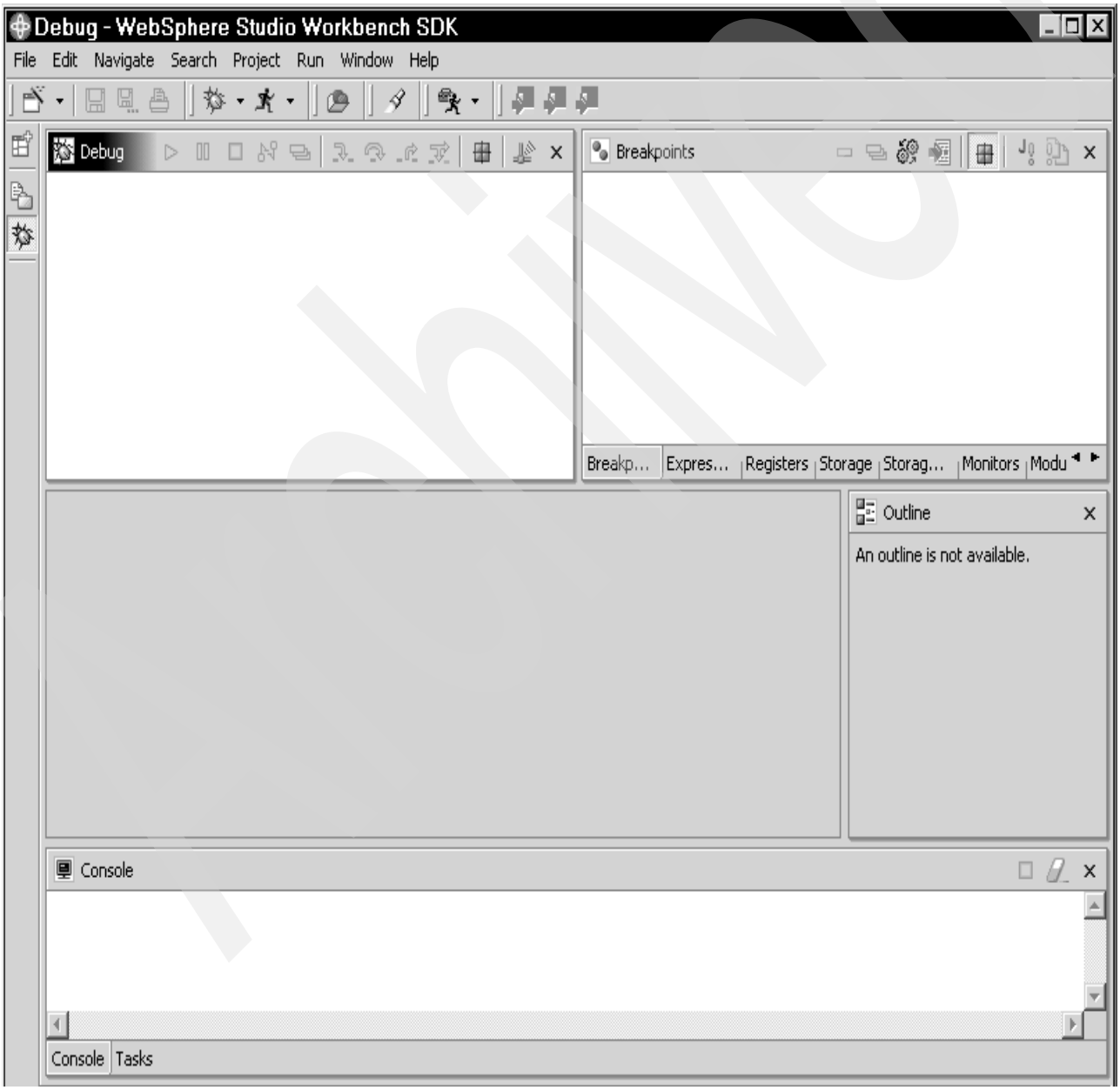


Figure 4-1 The WebSphere Studio Workbench Debug Perspective

At the top of the upper left-hand corner window, which is entitled “Debug,” there is a series of icons as shown in Figure 4-2.

Figure 4-2 The debugger control icons

The final icon is the Listen button. If you click this button, the debugger will begin to listen on channel 8001. The next step to debugging is going back to your TSO, Batch, IMS, or CICS session and starting the job you wish to run with a suitable TEST runtime option. The test runtime option will need to specify the TCP/IP address of the machine that is running the WSED debugger. In this case, the program invocation is as shown in Figure 4-3.

```
  REDBK1.SPFLOG1.LIST has been deleted.
 READY
call dev.load(knightm) 'TEST(,,,TCPIP&9.30.62.149%8001:*) / 8'
```

Figure 4-3 Invoking the program to be debugged on the mainframe

In this example we have invoked the program in TSO, but we could have invoked it in batch just as easily. The TEST runtime option contains enough information that the runtime on the mainframe can contact WSED on our PC.

The WSED screen pops up, along with a dialog box to warn about the state of the program as shown in Figure 4-4.

Figure 4-4 WSED Initial Program Status message

Once we have clicked **OK** and stepped into the program, we have a fully functional point-and-click debugging environment, with many different views such as breakpoints, variables, registers, and so forth, as shown in Figure 4-5.

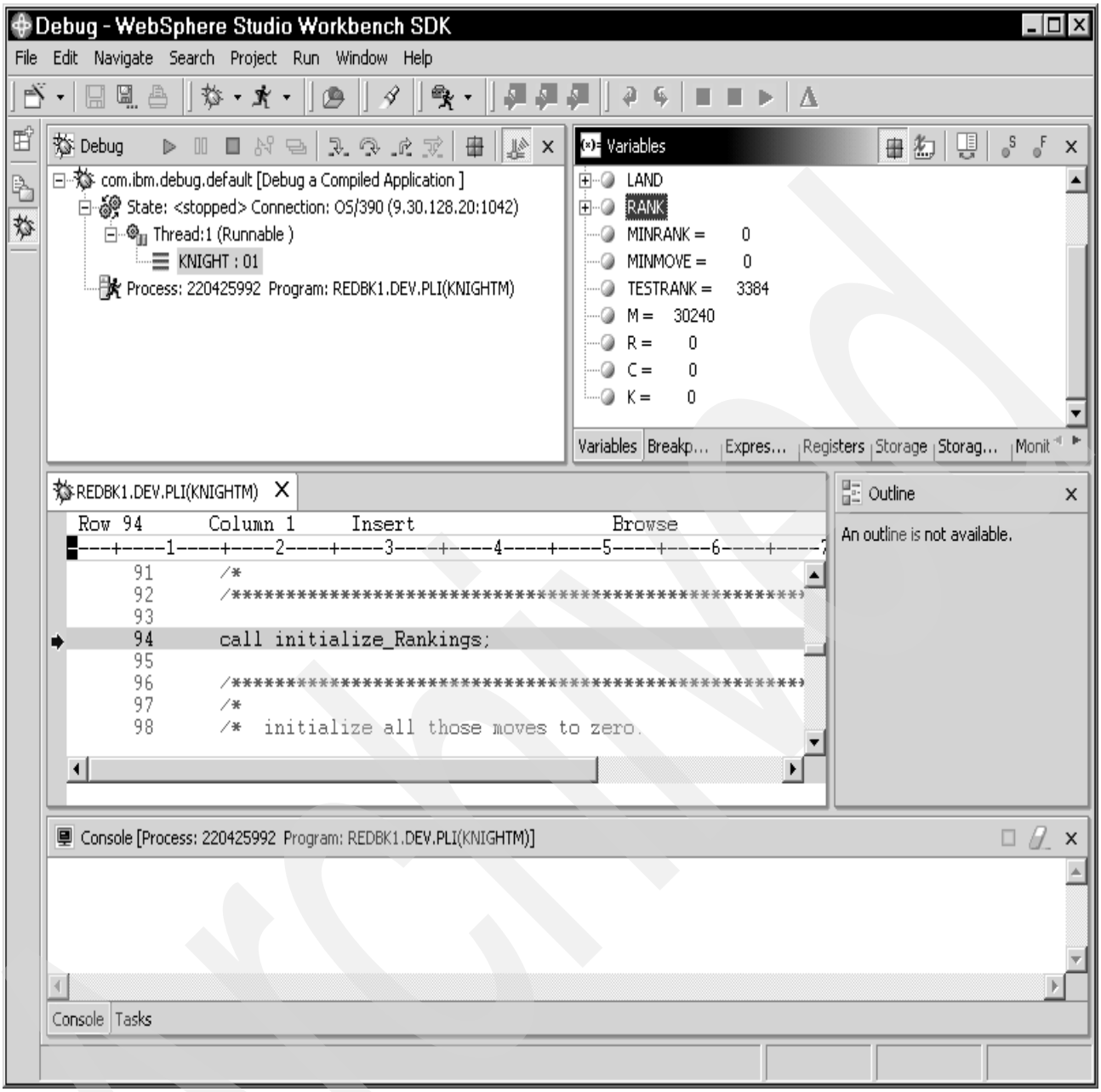


Figure 4-5 The WSED Debug screen

IBM Distributed Debugger

Any version of the IBM Distributed Debugger should work with Debug Tool and all of its supported languages.

Using this interface, your host application running under control of the Debug Tool will start a TCP/IP connection to a Windows NT, 2000, or XP workstation on which the IBM Distributed Debugger program was started and listens for requests on a specified port (default is 8000). This interface is supported by all environments including CICS, IMS, and UNIX System Services (USS).

Note: In some TCP/IP installations, a SYSTCPD DD card may be needed to point to your installation's TCPIPDATA.

Debugging with the IBM Distributed Debugger is a two-step process:

1. Start the remote debugger on your workstation. For the IBM Distributed Debugger on a workstation, this involves a command like:

   ```
   C:\IBMDebug\bin\idebug.exe -qdaemon -quiport=8000
   ```

2. Run your program using the TEST runtime option. The runtime option to be used needs to define your workstation's TCP/IP address. For example, with a COBOL program this might be:

   ```
   /TEST(ALL,*,PROMPT,VADTCPIP&9.30.40.117%8000:*)
   ```

A sample session of the IBM Distributed Debugger is shown in Figure 4-6.

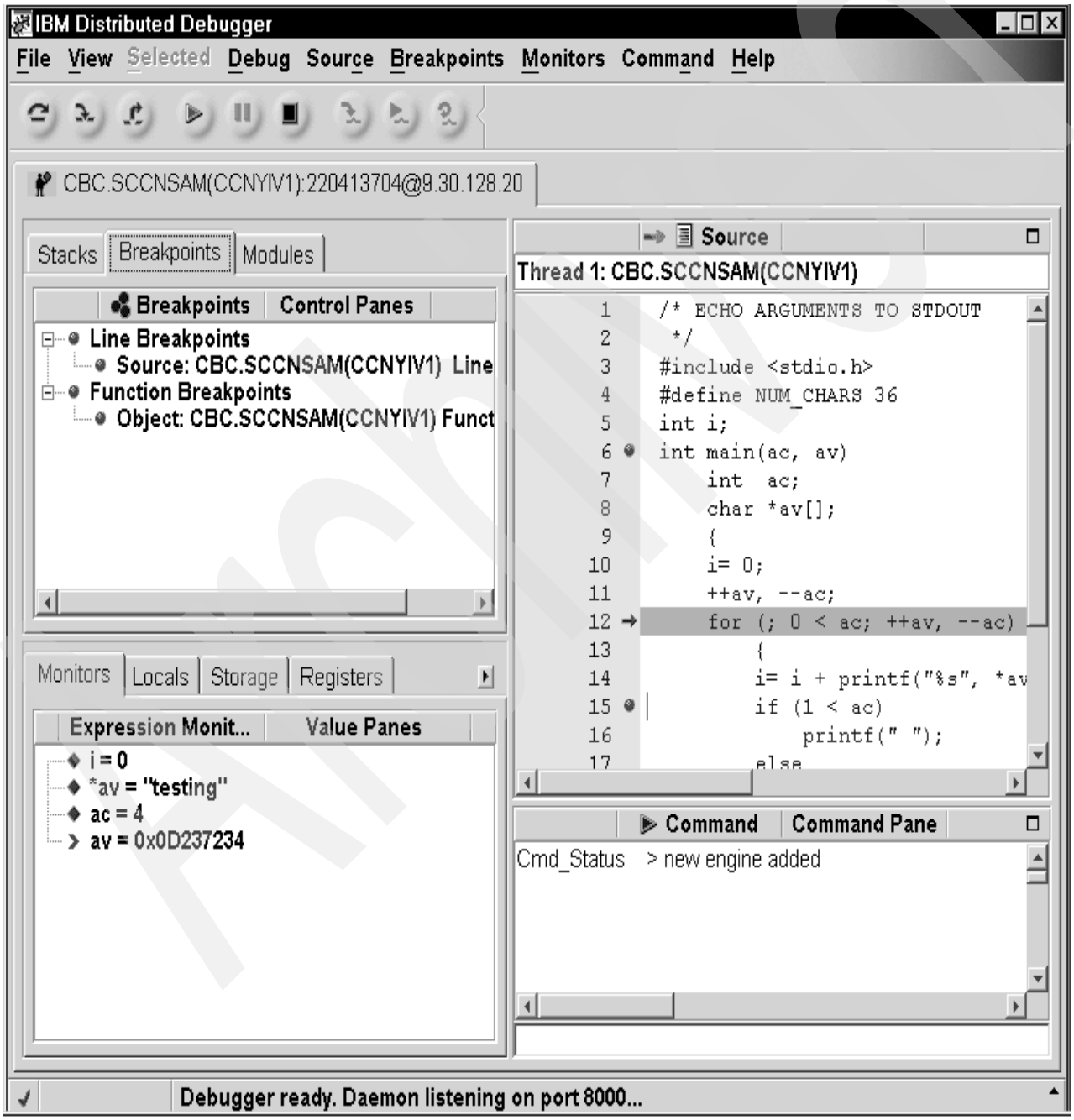


Figure 4-6 A view of the IBM Distributed Debugger screen

Full-screen mode

Debug Tool provides an interactive full-screen interface on a 3270 device, with debugging information displayed in the following three windows:

- Source window: Displays the program source or listing
- Log window: Displays a record of commands and other interactions between Debug Tool and the program
- Monitor window: Indicates changes in the program

Programs written in all languages supported by Debug Tool can be debugged in full-screen mode.

Source window

The Source window displays the source file or listing. It has four parts:

Header area Identifies the window, shows the compile unit name, and shows the current position in the source or listing.

Prefix area Occupies the left-most eight columns of the Source window. Contains statement numbers or line numbers which can be used when referring to the statements in your program. The prefix area can be used to set, display, and remove breakpoints with the prefix commands AT, CLEAR, ENABLE, DISABLE, QUERY, and SHOW.

Source display area Shows the source code (for a C/C++ program), the source listing (for a COBOL or PL/I program), a pseudo assembler listing (for an assembler program), or the disassembly view (for programs without debug information) for the currently qualified program unit. If the current executable statement is in the source display area, it is highlighted.

Suffix area A narrow, variable-width column at the right of the screen that Debug Tool uses to display frequency counts. It is only as wide as the largest count it must display. The suffix area is optional. To show the suffix area, enter SET SUFFIX ON. To hide the suffix area, enter SET SUFFIX OFF. It can also be set on or off with the Source Listing Suffix field in the Profile Settings panel.

Log window

The Log window records and displays user interactions with Debug Tool. All commands that are valid in line mode, and their responses, are automatically appended to the Log window. The following commands are not recorded in the Log window:

- PANEL
- FIND
- CURSOR
- RETRIEVE
- SCROLL
- WINDOW
- IMMEDIATE
- QUERY prefix command
- SHOW prefix command

If SET INTERCEPT ON is in effect for a file, that file's output also appears in the Log window. The user can exclude STEP and GO commands from the log by specifying SET ECHO OFF.

By default, the Log window keeps 1000 lines for display. To change this value, enter SET LOG KEEP *n*, where *n* is the number of lines you want kept for display. The maximum number of lines is determined by the amount of storage available.

Monitor window

The Monitor window is used to continuously display output from the MONITOR LIST, MONITOR QUERY, MONITOR DESCRIBE, and SET AUTOMONITOR commands.

If this window is not open, Debug Tool opens it when the MONITOR or SET AUTOMONITOR commands are entered. Its contents are refreshed whenever Debug Tool receives control and after every Debug Tool command that can affect the display.

When the MONITOR command is issued, it is assigned a reference number between 1 and 99, then added to the monitor list. The user can specify the monitor number; however, it must either replace an existing monitor number or be the next sequential number.

When the user issues the SET AUTOMONITOR ON command (if available), the following line is displayed at the bottom of the list of monitored variables:

```
********** AUTOMONITOR **********
```

Variables that are added to the Monitor window as a result of the SET AUTOMONITOR command are displayed underneath this line.

While the MONITOR command can generate an unlimited amount of output, bounded only by your storage capacity, the Monitor window can display a maximum of only 1000 scrollable lines of output.

If a window is not wide enough to show all the output it contains, either SCROLL RIGHT (to scroll the window to the right) or ZOOM (to make it fill the screen) can be used.

In most cases, the user can update the values of monitored variables by typing new values over the displayed values.

A sample of the Debug Tool screen in the full-screen mode is shown in Figure 4-7.

```
COBOL    LOCATION: PROG1 :> 48.1
Command ===>                                                Scroll ===> PAGE
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 1
****************************** TOP OF MONITOR ******************************
0001  1 TEMP  'TOP  '
****************************** BOTTOM OF MONITOR ***************************



SOURCE: PROG1 ----1----+----2----+----3----+----4----+----5----+ LINE: 46 OF 53
     46            MOVE "TOP" TO STR1 MOVE "BEG" TO STR2 MOVE "UP" TO STR3
     47            ADD 1 TO VARBL1
     48            SUBTRACT 2 FROM VARBL2
     49            ADD 1 TO R
     50            MOVE "BOT" TO STR1 MOVE "END" TO STR2 MOVE "DOW" TO STR
     51          END-PERFORM.
LOG 0----+----1----+----2----+----3----+----4----+----5----+---- LINE: 19 OF 22
0019     LIST ( "Exiting ", %CU ) ;
0020   STEP ;
0021   MONITOR
0022     LIST TEMP ;
PF  1:?         2:STEP      3:QUIT      4:LIST      5:FIND      6:AT/CLEAR
PF  7:UP        8:DOWN      9:GO       10:ZOOM     11:ZOOM LOG  12:RETRIEVE
```

Figure 4-7 Debug Tool full-screen session

4.1.2 Compiler options

Each programming language has a comprehensive set of compiler options. It is very important to use the correct compiler options to debug the program:

C/C++ TEST(ALL) provides maximum debugging capability. There are suboptions to refine debugging capabilities.

COBOL TEST(ALL,SYM) provides maximum debugging capability. There are suboptions to refine debugging capabilities. Some suboptions are used only with a specific version of COBOL. When using Enterprise COBOL for z/OS and OS/390 Version 3 or COBOL for OS/390 and VM Version 2 Release 2 compilers, TEST(NONE,SYM,SEPARATE) compiler option retains most of the Debug Tool's capabilities. The suboption SEPARATE instructs the compiler to store debugging information and symbol tables in a separate file. The suboption NONE specifies that there is no compiled-in hooks, so the Dynamic Debug facility must be activated during a debug session.

PL/I TEST(ALL,SYM) provides maximum debugging capability. Programs compiled with the PL/I for MVS or OS PL/I compilers must specify the SOURCE suboption. The suboptions BLOCK, STMT, PATH, ALL regulate the points in which compiler inserts hooks. The suboption SYM controls the insertion of symbol tables into the object file. These tables are used by Debug Tool to obtain information about program variables. The syntax for the TEST compiler option of the Enterprise PL/I compilers is slightly different. Refer to the documentation that corresponds to the version of the compiler you are using for a description of the TEST compiler option.

Assembler ADATA option must be specified. This option generates a SYSADATA file, which the EQALANGX postprocessor needs to create a debug file (also called the EQALANGX file).

4.1.3 Link-edit options

In most cases, Debug Tool does not require specific link-edit options for application programs.

When using the DTCN transaction to manage debugging profiles in CICS, the main programs to be debugged should be link-edited with the object module EQADCCXT if they are written in PL/I or C/C++. When using the CADP transaction, which is available with CICS Transaction Server for z/OS V2.3 and later, this is not required.

4.1.4 Runtime TEST option

Note: The following information is based on the description of the TEST runtime option provided in Debug Tool V5R1 Reference and Messages, SC18-9304-00.

About runtime TEST option

The TEST runtime option is used to specify the conditions under which Debug Tool will assume control of an application. The basic format of the instruction is as follows:

NOTEST Specifies that Debug Tool is not started at program initialization. However, it can still be started with CEETEST, PLITEST or __ctest(). The suboptions specified with NOTEST are used when Debug Tool is stared (if it is started). Note that if the TEST option is specified somewhere that has a higher precedence than where the NOTEST option is, the values on the NOTEST option will be taken as defaults.

TEST — Indicates that Debug Tool is given control according to the specified sub-options.

test_level

The test_level suboption has three possible values:

ALL — This default value specifies that Debug Tool gains control, even without defined breakpoints, at:

The attention function
Any Language Environment condition of severity 1 or above
Application termination

ERROR — Without a defined AT OCCURRENCE for a particular condition, Debug Tool will only get control at the occurrence of:

The attention function
Any Language Environment condition of severity 1 or above
Application termination

NONE — This specifies that no condition will cause Debug Tool to gain control without a defined AT OCCURRENCE for a particular condition or AT TERMINATION.

commands_file

The commands_file designator is a valid DD name or file name that gives the name of the primary commands file for this program run. If this parameter is empty, requests for commands will go to the user terminal.

If an * is specified instead of a commands file, then no commands file is expected.

prompt_level

The prompt_level suboption is used to specify whether an initial commands list is unconditionally executed during program initialization or to specify particular Debug Tool commands:

PROMPT — This default value specifies that Debug Tool is invoked at Language Environment Initialization.

NOPROMPT — Specifies that the Debug Tool is not invoked at Language Environment Initialization.

* — Equivalent to NOPROMPT.

; — Equivalent to PROMPT.

command_list — A character string that specifies a valid Debug Tool command. Maximum 250 characters. It should be enclosed in single or double quotes whenever it contains embedded blanks, commas, or parenthesis. The use of a preferences file is recommended rather than putting a command list in the third sub-option.

preferences_file

The final suboption is the preferences_file. This suboption controls the interface and location of the debugger as well as the location of a preferences file that becomes the first source of Debug Tool commands after Debug Tool has started. This suboption has a complex format as shown in Figure 4-8.

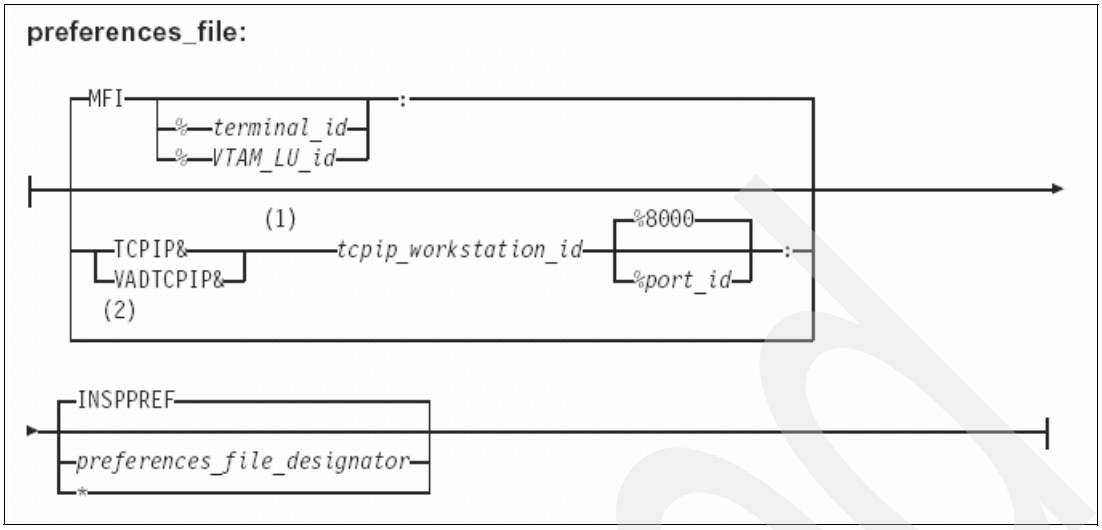


Figure 4-8 Preferences file syntax

Figure 4-8 Notes:

(1) Specifies remote debug mode using WebSphere.Studio Enterprise Developer.

(2) Specifies remote debug mode using the VisualAge Remote Debugger or the IBM Distributed Debugger.

The terms used are as follows:

- MFI (Main Frame interface): Specifies Debug Tool should be started in full-screen mode through a VTAM terminal for your debug sessions.
- terminal_id (CICS only): Specifies up to a four character terminal ID to receive Debug Tool screen output during dual terminal debugging of a CICS application.
- VTAM_LU_id (Full-screen mode using a VTAM terminal only): This is a VTAM logical unit (LU) name for a terminal. This cannot be used to debug CICS applications. It is used for full screen debugging. The application will continue in its original location and the new unused VTAM terminal will be used for the full screen MFI debugger. This can be used for batch or IMS debugging as well as traditional debugging.
- INSPREF (or blank) is the default DD name for the preference file.
- preferences_file_designator: This is a valid DD name, data set name, or file name specifying the name of the preference file. The preference file is a command file that can be used to specify the Debug Tool commands to be executed on entry to your environment.
- *: Instead of a preference file, the asterisk indicates that no preference_file is supplied.
- %port_id: Specifies the TCP/IP port on your workstation to be used by the remote debugger. The default port for WebSphere Enterprise Developer Compiled Language debugger is %8001. The default port for the IBM Distributed Debugger or the VisualAge Remote Debugger is %8000.

Sample runtime options

- NOTEST
  Debug Tool is not started unless there is a call to CEETEST, PLITEST or __ctest().
- NOTEST(ALL,MYCMDS,*,*)
  Debug Tool is not started at initialization. If it is started by a call to CEETEST, PLITEST, or

__ctest(), the suboptions will be used and the instructions in the file allocated to DD name MYCMDS.

- TEST
  Searches through runtime options specified in any level (CEEUOPT, pragma runopts, CEEROPT) and brings up the debugger using the options found on a prior TEST instruction or the default TEST values.
- TEST(ALL,*,*,*)
  Debug Tool is not started initially but any condition in the code (or a call to CEETEST, PLITEST or __ctest) will cause it to be started. No preference file or command file is expected.
- TEST(NONE,,*,*)
  Debug Tool is not started until a call to CEETEST, PLITEST, or __ctest() occurs.
- TEST(ALL,test.scenario,PROMPT,prefer)
  Debug Tool is started after environment initialization but before program initialization. The first commands executed are found in the file referred to by the DDNAME prefer.
- TEST(ALL,,,MFI%F100:*)
  For CICS Dual Terminal and Batch CICS transactions, Debug Tool will be started at CICS terminal F100 after initialization. Alternatively, F100 could be a very short VTAM LU ID that is used to define a terminal for debugging a batch or IMS transaction. No preference file is expected.
- TEST(ALL,,,MFI:foo) or TEST(ALL,,,foo)
  These are equivalent instructions when used in TSO that cause the full screen interface to be used in single screen mode (the same place that the program was called from). In both cases, the preference file is the file referenced by the foo DD.
- TEST(ALL,,,MFI%LU000001:*)
  For environments other than CICS. The MFI full screen debugger will be started on the VTAM LU whose ID is LU000001. The LU must be known to VTAM and not in session when the debugger is started.
- TEST(,,,VADTCPIP&9.30.62.149%8000:*)
  Bring up the IBM Distributed Debugger or the VisualAge Remote Debugger listening on channel 8000 on the terminal at TCP/IP address 9.30.62.149.
- TEST(,,,TCPIP&cello%8001:*)
  Bring up the WebSphere Studio Enterprise Developer listening on port 8001 on the machine named cello. This method does not work in all environments and requires an up to date name server to be available. It was found that using the actual machine address is more consistently effective.

Specifying TEST runtime option

Language Environment has several ways of specifying the runtime options for a program. The following list gives them in ascending order of precedence (that is, things lower on the list can override things higher on the list):

- CEEDOPT: Language Environment options specified at installation
- CEEROPT: Region-wide CICS or IMS default options
- CLER: Transaction under CICS
- CEEUOPT: Also #pragma runopts, or PLIXOPTS
- Command Line, or _CEE_RUNOPTS
- CEEBXITA: Used by IMS Single Point of Control (SPOC) and Debug Tool IMS utilities
- Language Environment Storage Tuning User Exit
- Options defined at installation time that have the non-overridable attribute

For more information on how to specify a Language Environment runtime option see the *Language Environment Programming Guide*.

In some cases, where runtime options cannot be passed to the Language Environment, a CEEUOPT must be generated including a specific TEST runtime option. In Example 4-1 the original sample can be found as member CEEUOPT on your SCEESAMP.

Example 4-1 Defining runtime options using CEEUOPT

```
*/**************************************************************/
*/* LICENSED MATERIALS - PROPERTY OF IBM                       */
*/*                                                            */
*/* 5694-A01                                                   */
*/*                                                            */
*/* (C) COPYRIGHT IBM CORP. 1991, 2001                         */
*/*                                                            */
*/* US GOVERNMENT USERS RESTRICTED RIGHTS - USE,               */
*/* DUPLICATION OR DISCLOSURE RESTRICTED BY GSA ADP            */
*/* SCHEDULE CONTRACT WITH IBM CORP.                           */
*/*                                                            */
*/**************************************************************/
CEEUOPT  CSECT
CEEUOPT  AMODE ANY
CEEUOPT  RMODE ANY
         CEEXOPT TEST(ALL,*,PROMPT,MFI%LU0TCP08:INSPREF)
         END
```

This member must be customized to reflect the actual terminal ID to be used in the debugging session. The name of this member also can be customized to reflect the intention of its use.

When specifying the TEST runtime option on a JCL PARM there are dependences on what is the language of the program's main entry point:

- For C/C++ and PL/I, the PARM should start with the runtime options and have a slash (/) before the program parameters.
- For COBOL, the PARM should start with the program parameters and have a slash before the runtime options.

The user can specify RPTOPTS(ON) in the JCL PARM to have the runtime options report generated. This report lists all runtime options which were in effect when the program was executed.

For CICS, the user can use the Language Environment provided CICS transaction CLER to have runtime options displayed on the terminal, as shown in Figure 4-9.

```
CLER                                                                PAGE 5 OF 6

           Language Environment Region Level Runtime Options.

Current Settings

LAST WHERE SET          OPTIONS
--------------------    ------------------------------------------------------
Installation default    TERMTHDACT(TRACE,CESE,96)
Installation default  NOTEST(ALL,"*","PROMPT","INSPPREF")
Installation default    THREADHEAP(4096,4080,ANYWHERE,KEEP)
Installation default    THREADSTACK(OFF,4096,4080,ANYWHERE,KEEP,4096,4080)
Installation default    TRACE(OFF,4096,DUMP,LE=0)
Installation default    TRAP(ON,SPIE)
Installation default    UPSI(00000000)
Installation default  NOUSRHDLR(,)
Installation default    VCTRSAVE(OFF)
Installation default    VERSION()
Installation default    XPLINK(OFF)
Installation default    XUFLOW(AUTO)



PF:  1=Help   3=Quit   7=Back   8=Forward   10=RPT->CES
```

Figure 4-9 Transaction CLER used to display runtime options

4.1.5 Special files

There are four special files used by Debug Tool in full-screen mode:

- Save file (INSPSAFE)

  This file, if allocated by the user, is used by Debug Tool to save the sizes of panels, colors, PF keys setting, and so forth, between debugging sessions. CICS does not support this file.

- Preference file (INSPPREF)

  This file contains Debug Tool commands used to customize the debugging session. The information about the user's preference file is passed to Debug Tool by specifying it in the TEST runtime option.

- Commands file (INSPCMD)

  This file contains Debug Tool commands that control the debugging session. It can be used to set breakpoints or set up monitoring for variables. The information about this file also should be specified in the TEST runtime option.

- Log file (INSPLOG)

  This file is used by Debug Tool to record the progress of the debugging session. The results of the executed commands are saved as comments, which allows you to use the log file as a commands file in later debugging sessions. Since this file is written to by the Debug Tool, it is recommended to allocate it as a sequential file, which will eliminate any contentions.

There is one additional special type of files used by Debug Tool. A separate debug file SYSDEBUG is produced by the compiler when compilation is performed with the SEPARATE suboption of the TEST compiler option. Currently this option is available only for COBOL for OS/390 and VM and Enterprise COBOL compilers.

4.1.6 Global preferences file enhancement

Debug Tool provides a mechanism where an installation-wide default preferences file can be specified and processed. The purpose is to have:

- A mechanism that is easy to set up at Debug Tool installation and customization time, but that is transparent to Debug Tool users.
- Consistent tailoring of the debug session, such as PF key assignments and window configuration. It provides a set of installation-wide preferences to all users.

For users who want personal customization, the existing preferences file parameter in TEST runtime option provides a way to add additional preferences or override the global settings.

Restrictions

The mechanism works in the:

- Debug Tool supported host subsystems (TSO, Batch, CICS, and IMS)
- MFI debug mode, but not in the remote debug mode (like the current INSPPREF)

Global preferences file location

The global preferences file is a sequential file or a PDS member residing on the host. The name of the file is coded in an Assembler CSECT called EQAOPTS. During Debug Tool installation and customization, you have to code the name of the global preferences file in the EQAOPTS CSECT, assemble it, and build the load module, EQAOPTS. The load module is then placed in a private data set concatenated in the 'load module search path' before hlq.SEQAMOD.

During initialization time, Debug Tool loads in EQAOPTS module and retrieves the global preferences file name.

Using EQAOPTS options file

EQAOPTS uses the EXAXOPT macro to define the global preferences. Two option are available as shown in Example 4-2.

Example 4-2 EQAOPTS

```
EQAOPTS  CSECT       ,
EQAOPTS  AMODE      31
EQAOPTS  RMODE      ANY
                    EQAXOPT  GPFDSN,'DEVELP.TEST.GLBLPREF'
                    EQAXOPT  SVCSCREEN,CONFLICT=NOOVERRIDE
                    EQAXOPT  END
                    END,
```

1. Global preferences file data set name

   GPFDSN: provides the data set name.

   To have a consistent tailoring of the debug session such as PF key assignment, window configuration, or other installation-wide default preferences, the global preferences file enforces that every debug session is initialized with the preferences in the global preferences file.

2. SVC screening filter

   SVCSCREEN(parm1,parm2)

   parm1 Enablement, possible values are ON and OFF

parm2 Conflict resolution, possible values are CONFLICT=OVERRIDE and CONFLICT=NOOVERRIDE

The filter allows you to enable SVC screening and to override the SVC screening already put in place by other vendor product. More information are available in the *Customization Guide*.

Global preferences file content

The Debug Tool commands allowed in the current preference file (INSPPREF) are eligible in the global preferences file.

Function

Debug Tool processes the global preferences file at initialization time, like the existing user preferences file and the commands file. The order of processing is as follows:

1. Global preferences file
2. User preferences file
3. Commands file

If a command is specified multiple times in a file, or in multiple files, the last instance is used.

If a user adds or modifies a preference by issuing the command directly in the command line, it is valid only in the current session and not persistent across sessions.

4.1.7 Frequently used commands

This section describes several commonly used commands. In these examples, all of the commands are entered on the command line and the results are displayed in the log window.

?

The ? command displays a list of all commands or, if used in combination with a command, displays a list of available options for that command.

AT

The AT command defines a breakpoint. The application program's execution is temporarily suspended when the point defined in this command is reached. The user can review the processing that has already taken place or issue other Debug Tool commands.

Example:

```
at line 334 list “about to setup files”;
go;
```

Result:

```
 AT LINE 334
   LIST "About to set up files" ;
 GO ;
EQA1140I About to set up files
```

AT CHANGE

The AT CHANGE command instructs Debug Tool to halt execution of the program whenever the contents of the defined variable are changed.

Example:

```
at change any-variable
```

CLEAR

The CLEAR command removes the actions of previously issued Debug Tool commands; this includes breakpoints.

Examples:

```
clear at;

clear log;
```

> **Note:** The CLEAR command cannot clear the contents of a log file directed to SYSOUT in a batch job.

COMPUTE (COBOL)

The COMPUTE command assigns the value of an arithmetic expression to a `WORKING-STORAGE` variable.

Example:

```
compute holdings = dec-no-shares * 10;
```

In PL/I and C/C++, the same would be done by using the assignment, as in the following example:

```
HOLDINGS = DEC_NO_SHARES * 10*
```

DESCRIBE

The DESCRIBE command displays information about the application program, variables, and the environment.

Example:

```
describe attributes ws-current-date;
```

Result:

```
  DESCRIBE ATTRIBUTES WS-CURRENT-DATE ;
 EQA1102I   ATTRIBUTES for WS-CURRENT-DATE
 EQA1105I     Its length is 8
 EQA1103I     Its address is 089826CD
 EQA1112I     02 TRADERB:>WS-CURRENT-DATE
 EQA1112I       03 TRADERB:>WS-YR   XXXX DISP
 EQA1112I       03 TRADERB:>WS-MM   XX DISP
 EQA1112I       03 TRADERB:>WS-DD   XX DISP
```

DISABLE/ENABLE

The DISABLE command makes the AT breakpoint inoperative, but does not clear it; the user can ENABLE it later without typing the entire command again.

Example:

```
disable at statement 334;
```

GO

The GO command instructs Debug Tool to start or resume running the program. The program will execute until a breakpoint is reached, or a condition occurs.

LIST

The LIST command displays information about a program, such as the values of variables, frequency information, and the like.

Use parenthesis around working storage variables to prevent any confusion with actual `LIST` operands.

Example:

```
list (ws-current-date);
```

MONITOR LIST

The MONITOR LIST command allows the user to observe changes to variables in the Monitor window while the program executes.

Example:

```
monitor list dec-no-shares;
```

MOVE (COBOL)

The MOVE command transfers data from one area of storage to another. This allows you to manipulate the contents of `WORKING-STORAGE` variables, and possibly alter the flow of the program as it executes.

Example:

```
move 250 to dec-no-shares;
```

In PL/I and C/C++, the same would be achieved by using an assignment, as in the following example:

```
DEC_NO_SHARES = 250;
```

QUERY

The QUERY command displays the values of Debug Tool settings and information about the current program. There are more than 30 forms to this command.

Example:

```
query location;
```

Result:

```
  QUERY LOCATION ;
 EQA1242I You are executing commands in the STATEMENT TRADERB ::> TRADERB :
          334.1 breakpoint.
 EQA1238I The current location is TRADERB ::> TRADERB :> 334.1.
```

SET

The SET command sets various switches that affect the operation of Debug Tool.

Example:

```
set echo off;
```

Result:

STEP and GO statements do not appear in the log window, but they do go to the log file.

STEP

The STEP command causes Debug Tool to execute one or more statements in the program.

Example:

```
step 5;
```

Result:

Debug Tool will execute five statements, one statement at a time.

4.1.8 Other useful commands

This section identifies some other commands that may be useful when debugging.

SET AUTOMONITOR (COBOL and PL/I)

Automonitor support automatically displays the variables referenced in the current statement and is available only when Debug Tool Utilities and Advanced Functions is installed. To use this feature, the user must enable automonitoring with the following command:

```
SET AUTOMONITOR ON ;
```

Refer to *Debug Tool for z/OS Reference and Messages*, SC18-9304 for more information on specific supported compilers and required PTFs.

GOTO

The GOTO command instructs Debug Tool to start or resume running the program at the specified statement, or the user can position the cursor at a specific statement and use a PF key defined as GOTO.

RUNTO

The RUNTO command instructs Debug Tool to run the program to the statement defined in the command, without setting a breakpoint; or the user can position the cursor at a specific statement and use a PF key defined as RUNTO.

GO BYPASS

The GO BYPASS command instructs Debug Tool to resume running the program after a condition occurred, hereby ignoring the condition.

PLAYBACK (COBOL)

Playback support allows you to replay and review the application paths and data values starting at the point where you began recording. You can simulate the backward execution of the application and review application data values using appropriate Debug Tool commands. To do this, playback must be enabled using the following Debug Tool command:

```
PLAYBACK ENABLE ;
```

This command starts playback recording. To view the recorded material, use the following commands:

`PLAYBACK START ;` Replay the statements starting at the current statement.
`STEP ;` (or `PF2`) Move backward one statement.
`PLAYBACK FORWARD; PLAYBACK BACKWARD ;` Change the replay direction. `STEP` will move in the direction last set.

`PLAYBACK STOP ;` Return to the point where the playback was started (using `PLAYBACK START`).

`PLAYBACK DISABLE ;` Stop recording.

This command is available only with Debug Tool Utilities and Advanced Functions.

WINDOW CLOSE/OPEN/SIZE/ZOOM

This command closes, opens, resizes, or toggles between full screen and the currently defined size of the Debug Tool full-screen mode Log, Monitor,and Source windows. The affected window is defined by the cursor position or can be added as a next parameter in the command itself.

4.1.9 Finishing a Debug Tool session

There are several ways to finish working with the Debug Tool. It is important to choose an appropriate one because it affects what actions will be performed in relation to the databases used in the program.

QUIT

Soft termination of the program occurs at the current statement, with a prompt message.

QQUIT

Soft termination of the program occurs at the current statement, with no prompt message.

QUIT ABEND

The program will be abended (ABENDU4038) at the current statement, with a prompt message.

> **Note:** When using QUIT ABEND, any non-committed database updates will be rolled back. It is a recommended setting to be used as the default for PF3.

The user can associate `QUIT ABEND` with the PF3 key by issuing the following command:

```
SET PF3 'ABEND' = QUIT ABEND;
```

> **Note:** This PF key setting will be saved to the INSPSAFE file, if one was allocated. CICS does not support INSPSAFE; therefore, this command should be placed in the INSPPREF file when running under CICS. This approach can also be used for other environments.

QUIT DEBUG

The debugging session will be terminated, but the program will continue to run to completion.

4.1.10 Built-in functions

There are several built-in functions defined in Debug Tool. Two of them are presented here.

%HEX

Returns the hexadecimal value of the operand.

%GENERATION (PL/I)

Returns a specific generation of a controlled variable in the program.

4.1.11 Dynamic Debug Facility

The Dynamic Debug facility enables the user to debug COBOL programs compiled with the NONE suboption of the TEST compiler option, assembler, and disassembled programs.

The user must activate the Dynamic Debug facility (by using the command SET DYNDEBUG ON) to debug programs that run without the Language Environment runtime.

The Dynamic Debug facility can be used to improve the performance of programs with compiled-in hooks (compiled with certain compilers) while debugging them.

Programs written in C/C++ and PL/I must be compiled with the TEST option.

If the Dynamic Debug facility has been installed, the initial setting is ON. If it was not installed, the initial setting is OFF and the facility cannot be activated by the user.

4.2 Debug Tool Utilities and Advanced Functions

Debug Tool Utilities and Advanced Functions provides enhancements to Debug Tool, and the combined strength of these products can help the user to debug and examine programs.

4.2.1 Debug Tool Utilities

Debug Tool provides a tool called Debug Tool Setup Utility (DTSU) to help with creating and managing setup files.

Debug Tool Utilities and Advanced Functions adds tools to help the user with the following tasks:

- Preparing high-level language programs for debugging by helping convert, compile, and link.
- Preparing assembler programs for debugging by helping assemble, create debug information, and link.
- Conducting analysis on test cases to determine how thoroughly test cases test programs (also called *code coverage*).
- Starting and running a program in foreground or batch by storing and using setup information. Setup information can be the run-time parameters, libraries, and names of input and output data sets.
- For IMS Version 8, browsing and editing the Language Environment run-time parameters table.
- Creating a batch job for a private IMS message region with customized load libraries and region attributes.
- Converting old COBOL source code and copybooks to new versions of COBOL by using COBOL and CICS Command Level Conversion Aid (CCCA).

The combination of DTSU and these tools is called Debug Tool Utilities. The main panel of Debug Tool Utilities is presented in Figure 4-10.

```
---------------------------- Debug Tool Utilities ----------------------------
Option ===> g_____________________________________________________________________

                                                                    More:     +
0  Manage Job Card
   For Program Preparation and Setup File Management

1  Program Preparation
   Compile old or new COBOL programs with newer compilers, convert old COBOL
   source into new COBOL source, use other compilers, and link edit.

2  Manage and Use Debug Tool Setup Files
   You can manage setup files and use them to run your program interactively
   with Debug Tool in TSO Foreground or submit your program to run in
   the background using MVS batch.

3  Code Coverage
   Measure code coverage in programs written in COBOL, PL/I, C/C++ and
   Assembler when compiled with specific IBM compilers and HLASM.

4  Manage IMS Programs
```

Figure 4-10 Debug Tool Utilities main panel

Tip: Using 32x80 or even 43x80 screen size for 3270 display is helpful when working with Debug Tool Utilities.

Enter the option **g** on this panel (optionally, scroll down by pressing PF8 to see this option) and Debug Tool Utilities will present an extensive online introduction (about 20 screens).

Debug Tool Utilities has a convenient cursor-driven help function. To use it, move the cursor to any input field and press PF1.

Creating and managing setup files

To create and manage setup files, select the option **2** on the main panel to start Debug Tool Setup Utility (DTSU), which is part of Debug Tool. You do not need Debug Tool Utilities and Advanced Functions to use this tool. The DTSU starting panel is shown in Figure 4-11.

```
------------------- Debug Tool Foreground - Edit Setup File ------------------
Command ===> ____________________________________________________________________


Setup File Library:
   Project . . . SLITVAK
   Group . . . . DTUDEMO  . . . ________ . . . ________ . . . ________
   Type  . . . . DTSF
   Member  . . . EXAMPLE1        (Blank or pattern for member selection list)
                                 (or existing or new member name)

Other Data Set Name:
   Data Set Name . . . ____________________________________________________
   Volume Serial . . . ______    (If not cataloged)


_ Initialize New setup file for DB2 (/)
```

Figure 4-11 DTSU panel

Setup files can save time when debugging a program that needs to be restarted multiple times. Setup files store information needed to run the program and start Debug Tool. The user can create several setup files for each program; each setup file can store information about starting and running the program in different circumstances.

Fill in the setup file library fields on the panel and press Enter. The next panel is shown in Figure 4-12.

```
EDIT - Edit Setup File 'SLITVAK.DTUDEMO.DTSF(EXAMPLE1)'        Row 1 to 1 of 1
Command ===> ___________________________________________ Scroll ===> PAGE

Modify information and use the  Run command to execute,
        or the  Submit command to submit to Batch.
     Press HELP for a list of all available commands.

Load Module Name SAMPLE1
Choose the format of your parameter string:
1 1 LE COBOL Default    - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs    - Run-time Options  / Program Arguments

/ Enter / to modify parameters ____________________________________________

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)            DISP
___ ________ ___ _ ***************** Top of Data ********************  _____
******************************** Bottom of data ********************************
```

Figure 4-12 Creating runtime parameters string

Select the option to enter or modify parameters and press Enter. The panel shown in Figure 4-13 will be displayed. Fill in the values.

```
--------------- Debug Tool Foreground - Modify Parameter String --------------
Command ===> ____________________________________________________________________
                                                                   More:     +
Modify Test options, other run-time options, and Program arguments

Select Test Options:

Test Option      ==> TEST___________ Test/Notest
Test Level       ==> ALL____________ All/Error/None
Commands File                        *, DDname, or Data Set Name
                 ==> *______________________________________________________
Prompt Level     ==> PROMPT_________ Prompt/NoPrompt
Preference File                      *, DDname, or Data Set Name
                 ==> INSPPREF_______________________________________________

Select (/) a session type and provide parameters:

 / Full-screen mode
   Terminal LU      ==> TRMLU001____ blank or MFI VTAM Terminal LU

 _ Remote debug mode
   Connection type ==> ________      SINGLE/MULTIPLE socket
   Address         ==> ______________________________________________________
```

Figure 4-13 Runtime options and session type selection, part 1

Press PF8 to scroll to see the lower part of it and add, if required, more run-time options and program arguments, as shown in Figure 4-14.

```
--------------- Debug Tool Foreground - Modify Parameter String --------------
Command ===> ____________________________________________________________________
                                                                   More:   -
Select (/) a session type and provide parameters:

 / Full-screen mode
   Terminal LU      ==> TRMLU001____ blank or MFI VTAM Terminal LU

 _ Remote debug mode
   Connection type ==> ________      SINGLE/MULTIPLE socket
   Address         ==> ______________________________________________________
   Port            ==> 8000________

Other run-time options: ____________________________________________________
____________________________________________________________________________
____________________________________________________________________________

Program arguments: _________________________________________________________
____________________________________________________________________________
____________________________________________________________________________
```

Figure 4-14 Runtime options and session type selection, part 2

Press PF3. The newly constructed PARM string is shown in Figure 4-15.

```
EDIT - Edit Setup File 'SLITVAK.DTUDEMO.DTSF(EXAMPLE1)'        Row 1 to 1 of 1
Command ===> ____________________________________________ Scroll ===> PAGE

Modify information and use the  Run command to execute,
        or the  Submit command to submit to Batch.
     Press HELP for a list of all available commands.

Load Module Name SAMPLE1
Choose the format of your parameter string:
1 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs    - Run-time Options  / Program Arguments

_ Enter / to modify parameters '/TEST(ALL,'*',PROMPT,'MFI%TRMLU001:INSPPREF') '

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)          DISP
___ ________ ___ _ ***************** Top of Data ********************  _____
******************************** Bottom of data ********************************
```

Figure 4-15 Runtime PARM string created

Compiling, converting, linking

Debug Tool Utilities can help the user to prepare programs for debugging. The option **1** on the Debug Tool Utilities main panel allows the user to proceed with the actions presented in Figure 4-16 and Figure 4-17.

```
------------------------ Debug Tool Program Preparation ------------------------
Option ===> ____________________________________________________________________

                                                                    More:     +
1  COBOL Compile
   Using 5655-G53 IBM Enterprise COBOL for z/OS and OS/390

2  COBOL Convert and Compile
   Using 5648-B05 COBOL and CICS Command Level Conversion Aid
   and   5655-G53 IBM Enterprise COBOL for z/OS and OS/390

3  PL/I  Compile
   Using 5655-H31  IBM(R) Enterprise PL/I for z/OS

4  C and C++ Compile
   Using 5694A01 z/OS C/C++

5  Assemble
   Using High Level Assembler

L  Link Edit
   Using z/OS Binder
```

Figure 4-16 Debug Tool Utilities program preparation panel, part 1

Scroll down using PF8 to see more options.

```
------------------------ Debug Tool Program Preparation ------------------------
Option ===> ____________________________________________________________________

                                                                    More:   -

4  C and C++ Compile
   Using 5694A01 z/OS C/C++

5  Assemble
   Using High Level Assembler

L  Link Edit
   Using z/OS Binder

F  Fault Analyzer Side file Create
   Using 5655-G74 IBM Fault Analyzer for z/OS and OS/390

C  Convert old 68/74 Std COBOL to 85 Std COBOL
   Using 5648-B05 COBOL and CICS Command Level Conversion Aid

S  Manage System and User Settings
```

Figure 4-17 Debug Tool Utilities program preparation panel, part 2

By selecting the option **s** the user can specify system and user settings for program preparation, for example, compiler, DB2 precompiler, CICS translator, link-editor and Fault Analyzer product data sets, input and output data sets, and so forth, as shown in Figure 4-18.

```
-------------- Debug Tool Program Preparation - Manage Settings --------------
Command ===> ___________________________________________________________________

                                                                    More:     +
System Settings for Program Preparation (compiler product datasets, etc):
These settings are refreshed at the start of every session.
  1  COBOL Compile
  2  COBOL Conversion (CCCA)
  3  PL/I  Compile
  4  C and C++ Compile
  5  HLASM
  6  Link Edit
  7  Fault Analyzer IDILANGX

User Settings for Program Preparation (input and output datasets, etc):
These settings are saved between sessions until RESET.
  8  COBOL Compile
  9  COBOL Convert and Compile
 10  PL/I  Compile
 11  C and C++ Compile
 12  HLASM
 13  Link Edit
 14  Fault Analyzer IDILANGX
```

Figure 4-18 Options to manage settings for programs preparation

As an example, system settings for PL/I compiler are shown in Figure 4-19.

```
--------------------Manage System Settings - PL/I Compiler--------------------
Command ===> _____________________________________________________________________


PL/I Compiler Settings:
 Active: YES
 Title . 5655-H31  IBM(R) Enterprise PL/I for z/OS
 Library . . . 'ENPLI330.SIBMZCMP'
 Module Name . IBMZPLI

 DB2 Precompiler Settings:
 Library . . . 'DSN.DB2V810.SDSNLOAD'
 Module. . . . DSNHPC

 CICS Translator Settings:
 Library . . . 'CICSVS.TS310.CICS.SDFHLOAD'
 Module. . . . DFHEPP1$


Press ENTER to update.
```

Figure 4-19 System settings for PL/I compiler

User settings for the same compiler include several data sets, for example, listing, object, DBRM, allocation parameters, and naming patterns.

Code coverage

Determining code coverage helps to improve test cases so the users can test programs more thoroughly. Debug Tool Utilities provides Debug Tool Coverage Utility, a tool to report which code statements have been run by the used test cases. DTCU can be accessed by selecting option **3** on the Debug Tool Utilities main panel. DTCU provides the user with the options shown in Figure 4-20.

```
------------------------- Debug Tool Coverage Utility -------------------------
Option ===> ______________________________________________________________________

0  Defaults      Manipulate defaults
1  CntlFile      Work with the Control File
2  Setup         Create JCL for Setup
3  StartMon      Create JCL to Start the Monitor
4  Reports       Create JCL for Reports
5  Monitor       Control the Monitor
6  FastPath      FastPath

Enter X to Terminate
```

Figure 4-20 DTCU main panel

Using the code coverage report, the user can enhance test cases so they run code statements that were not run previously.

Preparing IMS run-time environment

The user can create private IMS message regions that can be used to debug test applications and, therefore, not interfere with other regions. For IMS Version 8, the Language Environment run-time parameters table can be modified without relinking the applications.

4.2.2 Advanced functions

Debug Tool provides an extensive set of commands to debug programs. Debug Tool Utilities and Advanced Functions enhances this set of commands by adding more than 15 new commands, including:

ALLOCATE	Allocates a file to an existing or temporary data set or a concatenation of existing data sets
CALL %FA	Starts and instructs Fault Analyzer to provide a formatted dump of the current state
DESCRIBE	Displays attributes of file allocations, references, compile units, and execution environment
FREE	Releases previously allocated file
LOADDEBUGDATA	Specifies that a compile unit is an assembler compile unit and loads the corresponding debug file
PLAYBACK *	Directs Debug Tool to start or stop recording steps and data history, perform recorded STEP and RUNTO commands forward or backward
SET ASSEMBLER	Turns ON or OFF some additional information useful when debugging an assembler compile unit
SET AUTOMONITOR	Controls the automonitoring

5

Introduction to Fault Analyzer for z/OS

The purpose of Fault Analyzer (FA) is to determine the cause of abends in an application program. You do not need to read through application or system dumps because the product has the ability to isolate the exact instruction that caused the error.

In this chapter we identify the software levels that are required to use Fault Analyzer. We take a detailed look at how application programmers can use the product, and we review information that system programmers need to know to customize Fault Analyzer for their site. We briefly review the creation and use of user exits, and present some useful information that was discovered during our research. We conclude with a review of recent product updates.

5.1 Validating your software level

To effectively use Fault Analyzer, you must have the appropriate levels of software installed on your system. The Program Temporary Fix (PTF) we have listed should be reviewed to ensure that it is appropriate for your operating environment.

5.1.1 PTF information

Systems programmers responsible for installing and maintaining File Manager should review the RETAIN® PSP bucket UPGRADE.

The research for this book was done with the following maintenance level of Fault Analyzer: Fault Analyzer for z/OS V5R1M0, SYSMOD H29T510 as shown in Figure 5-1.

```
                          CSI QUERY - SYSMOD ENTRY          Row 1 to 3 of 90
===>                                                        SCROLL ===> PAGE

 To return to the previous panel, enter END .


 Primary Command: FIND

 Entry Type:  SYSMOD                              Zone Name: FAULT5T
 Entry Name:  H29T510                             Zone Type: TARGET
 Description: FAULT ANALYZER/BASE

   Type:    FUNCTION              Status: APP
   FMID:    H29T510    JCLIN
   Date/Time: 04.263   11:29:58   APP   REWORK   2004259



          -------- -------- -------- -------- -------- -------- --------
SUP       H1BK110  H26F210  H26G310  H29T410
DEL       H1BK110  H26F210  H26G310  H29T410

 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
 F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Figure 5-1 CSI query of H29T510 sysmod entry

You can see the software level by looking at the help about Fault Analyzer in the Fault Analyzer main panel. It is shown in Figure 5-2.

```
  File  Options  View  Services  Help
 ------------------------------------
IBM Fault Analyzer - Fau | _  1. Fault Analyzer User's Guide and Reference...
Command ===>             |    2. About Fault Analyzer...

Fault History File or View  : 'IDI.HIST'

{The following line commands are available: ? (Query), V (View real-time
report), I (Interactive reanalysis), B (Batch reanalysis), D (Delete).}

   Fault_ID Job/Tran User_ID  Sys/Job  Abend Date       Time
     F00006 CONOVERY CONOVER  STLABF6  SB37  2004/10/25 09:47:46
     F00005 CONOVERF CONOVER  STLABF6  S0CB  2004/10/21 11:05:18
     F00004 TIMOTHY  TIMOTHY  STLABF6  n/a   2004/10/11 16:21:13
     F00003 TIMOTHY  TIMOTHY  STLABF6  n/a   2004/10/11 14:31:39
     F00002 ZHONG    ZHONG    STLABF6  n/a   2004/10/11 10:53:17

*** Bottom of data.



 F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 5-2 Help option from Fault Analyzer Main menu

Selecting option **2** (About Fault Analyzer) from the pull-down menu will display FA general information, as shown in Figure 5-3.

```
                          About Fault Analyzer
 Copyright and General Usage Information                  Line 1 Col 1 76
 Command ===> _____________________________________ Scroll ===> CSR

 Fault Analyzer for z/OS and OS/390 V5R1M0 (H29T510 2004/09/14)

 Licensed materials - Property of IBM(*) 5655-M20

 (C) Copyright IBM Corp. 2000, 2004.  All rights reserved.

 US government users restricted rights - use, duplication or disclosure
 restricted by GSA ADP Schedule Contract with IBM Corp.

 Materials displayed or reproduced by this program may be protected by
 copyright or contract restrictions of IBM and/or others.  The user is
 responsible for having permission to display or reproduce such materials
 and for including applicable copyright notices and legends.

 If any IBM machine-readable documentation is accessed or reproduced by or
 through this program, IBM grants limited permission to licensees of the
 IBM machine-readable documentation to make hardcopy or other reproductions
  F1=Help     F3=Exit     F5=RptFind  F7=Up       F8=Down    F10=Left
 F11=Right
```

Figure 5-3 Fault Analyzer copyright and general usage information

The FA main menu is shown in Figure 5-4.

```
  File  Options  View  Services  Help
IBM Fault Analyzer - Fault Entry List                          Line 1 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR

Fault History File or View  : 'IDI.HIST'

{The following line commands are available: ? (Query), V (View real-time
report), I (Interactive reanalysis), B (Batch reanalysis), D (Delete).}

   Fault_ID Job/Tran User_ID  Sys/Job  Abend Date       Time
__   F00006 CONOVERY CONOVER  STLABF6  SB37  2004/10/25 09:47:46
__   F00005 CONOVERF CONOVER  STLABF6  S0CB  2004/10/21 11:05:18
__   F00004 TIMOTHY  TIMOTHY  STLABF6  n/a   2004/10/11 16:21:13
__   F00003 TIMOTHY  TIMOTHY  STLABF6  n/a   2004/10/11 14:31:39
__   F00002 ZHONG    ZHONG    STLABF6  n/a   2004/10/11 10:53:17

*** Bottom of data.

 F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 5-4 Fault Analyzer main menu

FA version 5 release 1 modification level 0 is currently the latest version. We recommend that you keep your maintenance level up to date.

5.2 Mechanics of Fault Analyzer

Once Fault Analyzer is installed and customized according to your system environment requirements, it is invoked automatically by intercepting and capturing whenever an application program abends. Fault Analyzer offers maximum flexibility through support for IBM CICS, IBM DB2, IBM IMS, Enterprise PL/I and COBOL, and IBM WebSphere Application Server for z/OS systems.

Figure 5-5 illustrates how FA works.

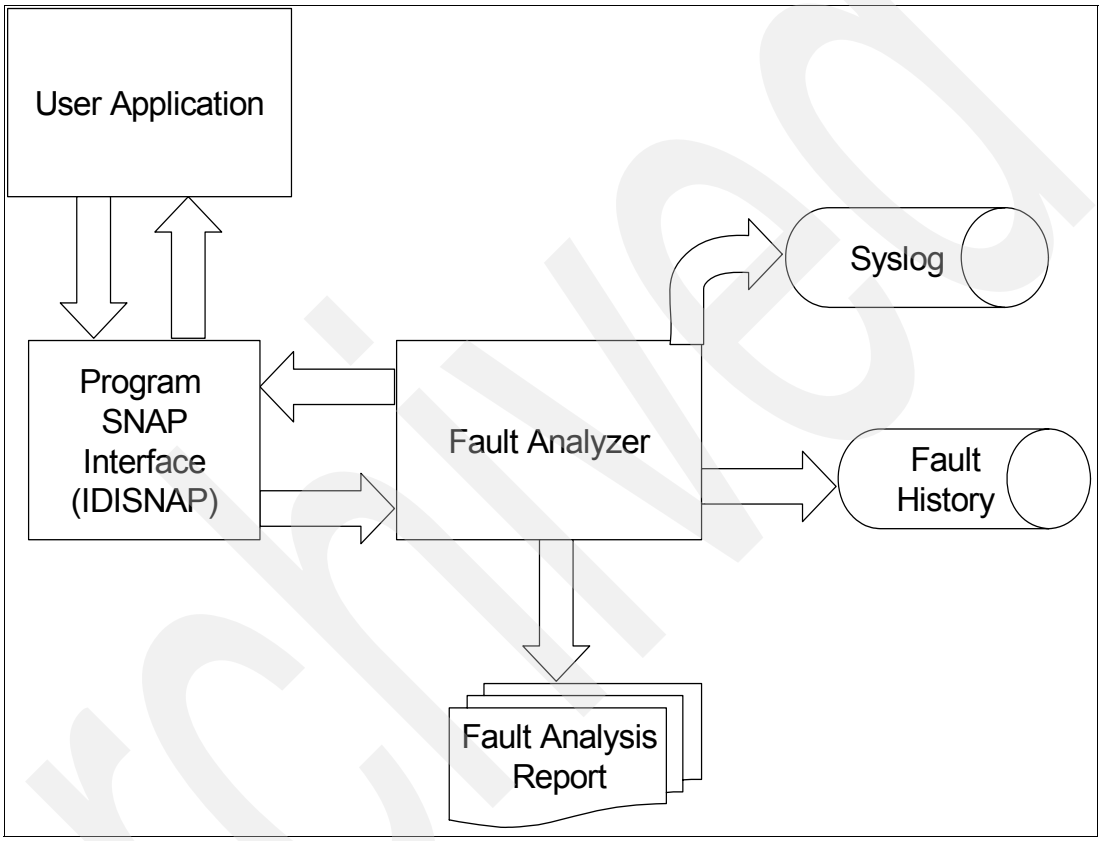


Figure 5-5 Fault Analyzer mechanism

When an application abends, the appropriate system exit obtains information and invokes FA. Depending on the system options that have been established, FA takes the information that is obtained, analyzes it, processes it, and writes it to the fault history file.

One of FA's powerful features is the ability to use the application program compiler's listing to identify the source statement of the line that caused the abend. Another feature that benefits you as a typical application programmer is its ability to make use of IBM's vast library of error messages and abend codes.

5.2.1 Fault history file

Fault Analyzer records a summary of an abend in a fault history file. The FA panel shown in Figure 5-6 provides access to the fault history file and displays the following information:

- Fault ID
- Job name or transaction ID that experienced the abend

- User ID that submitted the job
- System on which the abend occurred
- Type of abend
- Date and time of abend

Note: You can change the column configuration of this panel, by selecting 8. Column Configuration from the View pull-down menu item.

The history file also shows you the line commands that are available to process each entry in the list: **?** (Query), **V** (view real-time report), **I** (interactive re-analysis), **B** (batch re-analysis), and **D** (delete), as shown in Figure 5-6.

```
  File  Options  View  Services  Help
 ____________________________________________________________________________
IBM Fault Analyzer - Fault Entry List                        Line 1 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR

Fault History File or View  : 'IDI.HIST'

{The following line commands are available: ? (Query), V (View real-time
report), I (Interactive reanalysis), B (Batch reanalysis), D (Delete).}

   Fault_ID System   Job/Tran Abend Date       Time     User_ID  History_File_
__   F00010 STLABF6  IMWEBF6  S422  2004/11/02 16:33:22 WEBSRV   IDI.HIST
__   F00009 STLABF6  IMWEBF6  U4094 2004/11/02 16:32:51 WEBSRV   IDI.HIST
__   F00008 STLABF6  IMWEBF6  U4038 2004/11/02 07:15:14 WEBSRV   IDI.HIST
__   F00007 STLABF6  JMONITOR U4093 2004/10/29 07:34:48 n/a      IDI.HIST
__   F00006 STLABF6  CONOVERY SB37  2004/10/25 09:47:46 CONOVER  IDI.HIST
__   F00005 STLABF6  CONOVERF S0CB  2004/10/21 11:05:18 CONOVER  IDI.HIST
__   F00004 STLABF6  TIMOTHY  n/a   2004/10/11 16:21:13 TIMOTHY  IDI.HIST
__   F00003 STLABF6  TIMOTHY  n/a   2004/10/11 14:31:39 TIMOTHY  IDI.HIST
__   F00002 STLABF6  ZHONG    n/a   2004/10/11 10:53:17 ZHONG    IDI.HIST

*** Bottom of data.
 F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 5-6 Fault Main menu display

Note: You can select another history list by selecting an entry from the File pull-down menu selection 1. Last Accessed Fault History Files or Views, or by overtyping the history file name if you know the name of the file you wish to view.

5.2.2 Supported application environments

Fault Analyzer supports applications running under z/OS and OS/390 in the following applications environments:

COBOL PL/I

Assembler C/C++

Language Environment

UNIX System Services

CICS

IMS

DB2

MQSeries

WebSphere

Java

In the z/OS environment, Fault Analyzer executes in 31-bit addressing mode and performs analysis on 24-bit or 31-bit addressing mode applications. Fault Analyzer does not yet perform analysis on applications using 64-bit addressing mode.

5.2.3 Summary of real-time analysis

Real-time analysis occurs when an application abends and Fault Analyzer is invoked through one of the supplied invocation exits:

Invocation for non-CICS transaction abends (batch)

The following exits all invoke Fault Analyzer for real-time analysis when an abend, other than a CICS transaction abend, occurs (for example, batch).

- MVS change options/suppress dump IEAVTABX CSECT exit IDIXDCAP

  This exit can be used with both Language Environment and non-Language environment based batch application programs.

  The MVS IEAVTABX exit process is only called by MVS if the job step has allocated a SYSMDUMP, SYSUDUMP, or SYSABEND DDname; or if the IDITABD USERMOD has been applied.

  > **Note:** A matching SLIP TRAP with ACTION=NODUMP will prevent the MVS IEAVTABX exit from being called (for example, a CANCEL command resulting in an Sx22 abend for which most MVS systems will have a matching SLIP TRAP). To facilitate the Fault Analyzer analysis, either disable the matching SLIP TRAP or, for LE-based applications, use the batch LE abnormal termination CEEEXTAN CSECT exit IDIXCEE instead.

  Re-analysis of faults captured using this exit can be performed if a minidump was written, or if a SYSMDUMP DDname was allocated by the abending job step, and the SYSMDUMP data set is available.

  LE-enabled abends will need to run with TERMTHDACT, specifying the suboption UATRACE, UADUMP, UAONLY, or UAIMM in the LE options so that LE will call a system dump to activate this exit.

  This exit is APF-authorized and, therefore, able to extract job-related messages from the console.

- Batch LE abnormal termination CEEEXTAN CSECT exit IDIXCEE

  This exit is only effective with Language Environment based batch application programs.

  There is no requirement for a JCL SYSMDUMP DD statement to be allocated for this exit to be invoked.

  Re-analysis of faults captured using this exit can be performed if a minidump was written.

  If the LE option TERMTHDACT is used with the UATRACE, UADUMP, UAONLY, or UAIMM suboption, then the MVS change options/suppress dump exit will be invoked instead of the LE abnormal termination exit.

If both the batch Language Environment abnormal termination exit IDIXCEE and the MVS change options/suppress dump exit IDIXDCAP are installed, then the IDIXDCAP exit will intercept the abend instead of the LE exit if:

One of the following LE options is in effect:

- TERMTHDACT(UATRACE)
- TERMTHDACT(UADUMP)
- TERMTHDACT(UAONLY)
- TERMTHDACT(UAIMM)

and

A SYSABEND, SYSUDUMP, or SYSMDUMP DDname is allocated, or the IDITABD USERMOD has been applied.

Summary of exit usage

Table 5-1 indicates the exit used to invoke Fault Analyzer, depending on execution environment, options in effect, specification of MVS dump DD statement, and the IDITABD USERMOD.

Table 5-1 Non-Language Environment program

IDITABT usermod installed	SYSABEND,SYSUDUMP, SYSMDUMP dd provided YES	SYSABEND,SYSUDUMP, SYSMDUMP dd provided NO
YES	IDIXDCAP	IDIXCAP
NO	IDIXDCAP	Fault Analyzer not invoked

Invocation of CICS transaction abends

The following exits all invoke Fault Analyzer for real-time analysis when a CICS transaction abend occurs.

- CICS XPCABND and XDUREQ global user exit IDIXCX52 or IDIXCX53

  This exit is provided to invoke Fault Analyzer for CICS transaction abend analysis.

  All transaction abends are captured using this exit, except for U1xxx or U4xxx-type abends in Language Environment based applications. These transaction abend types can be handled by also installing the CICS LE abnormal termination CEECXTAN CSECT exit, IDIXCCEE, described in the next subsection.

  There is no requirement for a JCL SYSMDUMP DD statement to be allocated for this exit to be invoked.

  Re-analysis of faults captured using this exit can be performed if a minidump was written.

- CICS LE abnormal termination CEECXTAN CSECT exit IDIXCCEE

  This exit is only effective with Language Environment based CICS application programs that abend with LE U1xxx or U4xxx-type abends (for example, message IGZ0006S abend U4036 as the result of exceeding the array bounds in a COBOL application compiled using the SSRANGE option).

  There is no requirement for a JCL SYSMDUMP DD statement to be allocated for this exit to be invoked.

  Re-analysis of faults captured using this exit can be performed if a minidump was written.

  The LE option TERMTHDACT does not affect the invocation of this exit.

If both the batch Language Environment abnormal termination exit IDIXCEE and the MVS change options/suppress dump exit IDIXDCAP are installed, then the IDIXDCAP exit will intercept the abend instead of the LE exit if:

One of the following LE options is in effect:

- TERMTHDACT(UATRACE)
- TERMTHDACT(UADUMP)
- TERMTHDACT(UAONLY)
- TERMTHDACT(UAIMM)

and

A SYSABEND, SYSUDUMP, or SYSMDUMP DDname is allocated, or the IDITABD USERMOD has been applied.

SVC dump registration

One additional exit is provided with Fault Analyzer for SVC dump registration into a fault history file.

- MVS post-dump IEAVTSEL CSECT exit IDIXTSEL

  This exit is invoked whenever an SVC dump is written by the DUMPSRV address space.

  The use of this exit requires the Fault Analyzer subsystem to be active.

  No analysis is performed, but a dump registration fault entry is created. When this fault entry is first reanalyzed, a report and minidump is added.

  This exit is primarily intended for recording of CICS system dumps.

This is the first step in the fault analysis process. In most cases, the analysis will be deemed satisfactory, and you will not need to reanalyze the fault.

For a particular job you can adjust some options before you run the job.

The real-time analysis report is stored in the fault history file. You can view the report from the ISPF interface. You cannot change the report by setting options to different values at the time you view it. If you want to look at more (or less) detail, you must reanalyze the fault with adjusted options or a supplied listing or side file.

The real-time analysis report is automatically added to the job files, as if the JCL contained the following statement:

```
//IDIREPRT DD SYSOUT=
```

If you wish to divert the real-time analysis report to another file, then adjust the DD card as required, for instance, as shown in Example 5-1.

Example 5-1 DD card as required

```
//IDIREPRT DD DISP=(,CATLG),DSN=MY.REPORT.DS
// DCB=(RECFM=VB,LRECL=137),SPACE=(CYL,(1,1))
```

//IDIREPRT DD SYSOUT=

If you wish to divert the real-time analysis report to another file, then adjust the DD card as required, for instance, as shown in Example 5-2.

Example 5-2 DD card adjusted

```
//IDIREPRT DD DISP=(,CATLG),DSN=MY.REPORT.DS
// DCB=(RECFM=VB,LRECL=137),SPACE=(CYL,(1,1))
```

All virtual storage pages that were referenced during the analysis in the abending task's address space will be written to the history file as a minidump, unless the MaxMinidumpPages option in effect specifies a lower limit.

Note: LOADER restriction:

Fault Analyzer will not work correctly if using the LOADER (IEWBLDGO) since the load-and-go technique of link-editing modules does not write them to a data set. The data set copy of the load module is needed in order to determine CSECT names, lengths, and starting offsets.

5.3 Preparing your program for Fault Analyzer

FA will always provide the analysis of an abend. However, your application program must be compiled with specific compiler options for FA to display the source statement that caused the error.

Fault Analyzer uses the compiler listing to analyze the cause of abend, list the statement that caused the abend, and list the data values in the working-storage section.

5.3.1 Compiler options

Required compiler options

The following are the compiler options needed to produce listings or side files suitable for Fault Analyzer:

- C:
  - AGGREGATE
  - LIST
  - NOOFFSET
  - OPTIMIZE(0)
  - SOURCE
  - XREF
- C++:
  - LIST
  - LONGNAME
  - NOOFFSET
  - OPTIMIZE(0)
  - SOURCE
  - XREF
  - ATTRIBUTES(FULL)
- OS/VS COBOL:
  - DMAP
  - NOCLIST
  - NOLST
  - NOOPT

  - PMAP
  - SOURCE
  - VERB
  - XREF
- COBOL compilers other than OS/VS COBOL:
  - LIST,NOOFFSET
  - NOOPT
  - MAP
  - SOURCE
  - XREF(SHORT)
- VisualAge PL/I:
  - AGGREGATE
  - ATTRIBUTES(FULL)
  - LIST
  - NEST
  - OPTIONS
  - SOURCE
  - XREF
- Enterprise PL/I:
  - AGGREGATE
  - ATTRIBUTES
  - LIST
  - MAP
  - NEST
  - SOURCE
  - STMT
  - NONUMBER
  - OFFSET
  - XREF
  - OPTIONS
  - NOBLKOFF
- PL/I compilers other than VisualAge PL/I and Enterprise PL/I:
  - AGGREGATE
  - ATTRIBUTES(FULL)
  - ESD
  - LIST
  - MAP
  - NEST
  - OPTIONS
  - SOURCE
  - STMT
  - XREF
- Assembler:
  - ADATA

5.3.2 TEST option considerations

With all compilers, the additional use of the TEST option may provide program information in addition to what is available via the side files.

For COBOL specifically, the TEST option might affect the way in which the compiler listing data set name is determined.

TEST(,,SEPARATE) is used when compiling a COBOL program, then a COBOL SYSDEBUG file is written which can be automatically located by Fault Analyzer via the PROGSUM table, and used when providing source program information. This SYSDEBUG file must not be specified through any Fault Analyzer listing DDname, such as IDILCOB, nor can it be used as input to the IDILANGX side file creation program. If the SYSDEBUG file is to be used instead of a compiler listing, or an IDILANGX side file created from a compiler listing, then it must not be deleted.

If the SYSDEBUG file cannot be found by Fault Analyzer by its original data set name, then a call is made to the Debug Tool EQAUEDAT exit, which might provide a different data set name to be used.

6

Introduction to File Manager for z/OS

This chapter describes File Manager, including the following three types:

- FM/Base: File Manager for working with z/OS data sets
- FM/DB2: File Manager for DB2 data
- FM/IMS: File Manager for IMS data

6.1 Overview

All three types support a common set of tasks, plus each includes tasks unique to working with its particular data formats. There is a separate ISPF panel for each of the three main features of File Manager (Base, DB2, and IMS). Figure 6-1 shows the FM/Base panel.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 File Manager                 Primary Option Menu
 Command ===> _______________________________________________________________

 0  Settings      Set processing options                 User ID . : CHABERT
 1  Browse        Browse data                            System ID : STLABF6
 2  Edit          Edit data                              Appl ID . : FMN
 3  Utilities     Perform utility functions              Version . : 5.1.0
 4  Tapes         Tape specific functions                Terminal. : 3278
 5  Disk/VSAM     Disk track and VSAM CI functions       Screen. . : 1
 6  OAM           Work with OAM objects                  Date. . . : 2004/11/26
 7  Templates     Create, edit, or update templates      Time. . . : 18:06
 X  Exit          Terminate File Manager

                                                                     04/015
```

Figure 6-1 FM/Base panel

6.1.1 File Manager for working with MVS data sets

The product's features provide the ability to:

- Browse, edit, copy, update, create, compare, erase, and print:
  - QSAM data sets
  - VSAM data sets
  - PDS members
- Work with data formatted according to record structure, arranged into fields
- Edit entire files, regardless of size
- Work with files containing multiple record structures
- Use flexible criteria to select records
- Find and change data within particular fields
- Identify records that do not match a recognized structure, or that contain invalid values
- Create data with fields initialized according to flexible patterns
- Automate tasks in batch jobs, using File Manager functions and DFSORT or REXX procedures
- Compare data sets using templates
- Display an audit trail, either online through ISPF or using a batch job

6.1.2 File Manager for DB2 data

This feature extends the capabilities of File Manager to work with DB2 data:

- All of the standard File Manager functions are available, specifically designed for manipulating DB2:
  - Browse
  - Edit
  - Print
  - Copy
- You get to select the DB2 subsystem you want to work with.
- You can list DB2 objects in the system catalog.
- You have the ability to export and import DB2 tables and views.
- You can generate JCL and utility control statements for the following utilities:
  - Copy
  - Load
  - Rebuild
  - Reorg
  - Runstats
- You have the ability to prototype SQL SELECT statements using basic or advanced support.
- You can perform statement analysis using the EXPLAIN facility.

6.1.3 File Manager for IMS data

This feature extends the capabilities of File Manager to work with IMS data.

You can use File Manager IMS to:

- Browse, edit, or print data in an IMS database
- Extract from, load data into, or erase data from, an IMS database
- Format segments according to record structures defined in COBOL or PL/I copy books
- Access data bases via DL/I or BMP processing, and belonging to an IMS DC, DBCTL or CICS region

For many tasks, you can use the File Manager IMS features called *templates* and *views*. These allow you to define a logical view of a data set based on a COBOL or PL/I copybook. Associating a view with a data set lets you define which fields and records you want to work with, how the fields are displayed, and which segments are displayed.

6.2 Major functions

The key elements of File Manager that enable you to perform advanced or very detailed data manipulation are:

- Support for DFSORT
- Templates
- REXX functions

6.2.1 Support for DFSORT

Wherever possible, File Manager makes use of the IBM DFSORT product for copying and printing data sets and for editing large QSAM data files via an auxiliary data set. DFSORT should be used in preference to REXX wherever possible. DFSORT is used when all of the following conditions are met:

- IBM DFSORT has been installed at the correct maintenance level.
- The input data set is not partitioned.
- The record format of the input data set is not undefined.
- DFSORT is at the correct maintenance level.

When these conditions are met, performance of print and copy actions can be enhanced by supplying a procedure containing statements from a subset of DFSORT statements supported by File Manager, as an alternative to using equivalent REXX statements. DFSORT is automatically shipped with OS/390 and z/OS. As long as you have not manually removed it, you can use DFSORT with File Manager without requiring DFSORT to be licensed.

6.2.2 Templates

File Manager uses templates to provide a logical view of your data. To enable File Manager to determine the record structure of a file, you supply a copybook containing COBOL data description entries, or a PL/I include. File Manager interprets each level-01 group item in the copybook as a record structure, and each elementary item as a field.

After File Manager creates a template, you can add selection criteria and other formatting information. You use templates to map the data in your application files for a concise view of the contents. This includes the ability to view multi-record files.

You can save templates, eliminating the need to recreate them each time you browse or edit a file, and making them available for use with various File Manager utilities.

Templates can also be generated from DB2 table definitions.

6.2.3 REXX functions

REXX is an extremely versatile programming language, providing powerful functions and extensive mathematical capabilities. When used with File Manager, the possibilities for data analysis and manipulation, coupled with complex business logic, are extensive.

File Manager's external REXX functions allow you to manipulate data in the foreground, even while using templates. This gives you the opportunity to selectively work with only the records you are interested in. In addition to all of the functions available in REXX, File Manager has several product-specific functions, which include:

- VSAM support: VSAM data can be copied in and out of REXX stem variables.
- FLD: Lets you refer to a field from the current input record.
- NCONTAIN: Lets you check for the existence of numeric values in a field.
- TALLY: Lets you total a field and report the value.

You can develop REXX procedures to take the place of repetitive, manual functions, and then save these routines to a common data set.

6.2.4 Batch processing

All of the File Manager functions are available as primary commands in batch mode. You can easily enhance File Manager with your own procedures, built using either DFSORT or REXX. Compiled REXX runs considerably faster than interpreted REXX. By supplying a DD card for the REXX compiler library (if it is installed), you instruct File Manager to compile your REXX before running it. This approach reduces CPU usage and run time.

We describe how to use templates and provide sample DFSORT and REXX routines that you can use or modify for your own in Blah blah blah.

7

Introduction to Workload Simulator

This chapter describes Workload Simulator (WSim), which is a terminal and network simulation tool that can be used to evaluate network design, to perform and automate testing, and to determine system performance and response time.

WSim allows you to evaluate and test systems without having real terminals and terminal operators present. WSim can be used to simulate actions of several types of applications and terminals. The simulated resources communicate among themselves and with the real teleprocessing system (called system under test) as if they physically existed. The system under test does not need to be modified.

7.1 Overview

Workload Simulator (WSim) is an automated testing tool that can simulate terminals and other network devices and associated network traffic, and report the status and results of tests. It can be used to perform several types of tests. Workload Simulator has several components:

- Batch utilities
  - Capture data from live sessions
  - Prepare scripts
  - Run simulation tests
  - Generate reports
- WSim ISPF panels
  - Run utilities online
  - Generate JCL
  - Run simulation tests interactively
  - Review test results
- WSim Test Manager ISPF application
  - Simplifies and automates test process
  - Organizes tests by maintaining projects, test cases, network definitions, documentation, reports, and logs

Figure 7-1 provides a general overview of WSim use context.

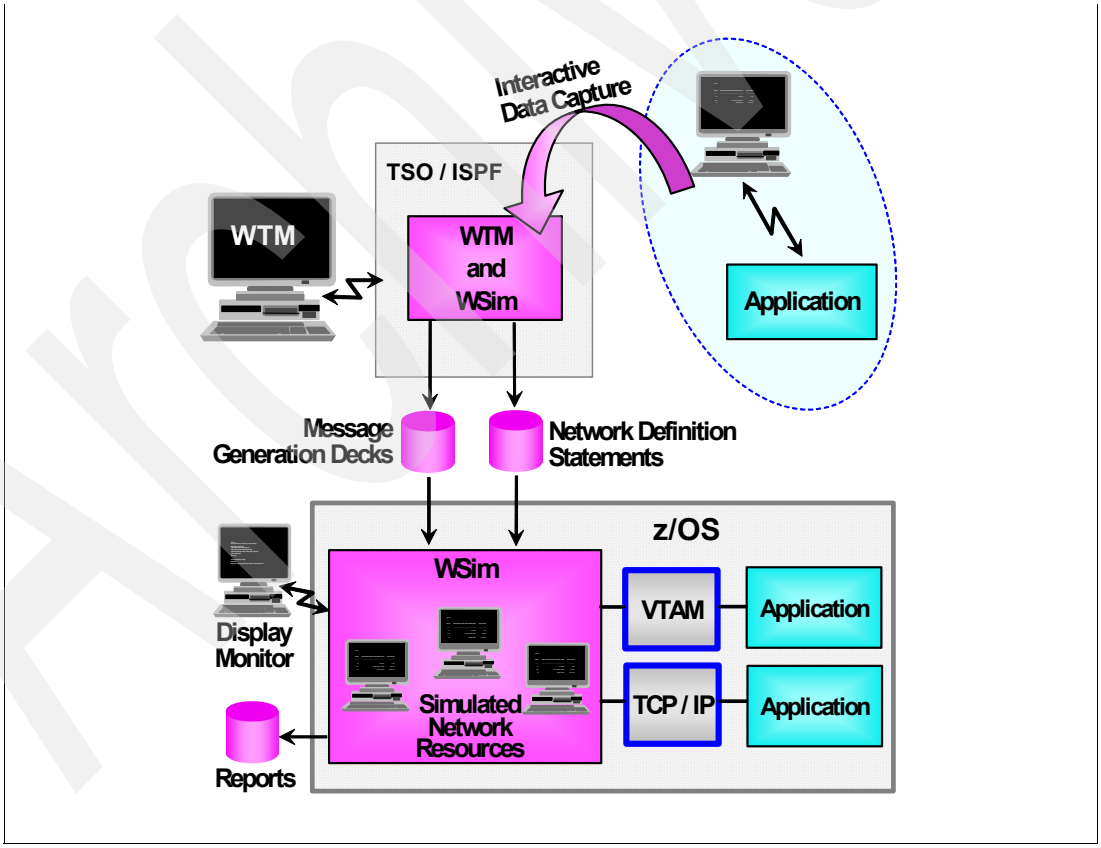


Figure 7-1 WSim - How it works

7.1.1 Use of WSim

To use WSim, two types of information must be prepared:

- Network definition statements describing the configuration of the network being simulated
- Message generation decks that send and receive messages

Network definition statements and message generation decks form a script, which WSim uses to send messages to the system under test. WSim collects and records the information received from the system under test. It also uses this information to determine what messages to send back to the system under test.

WSim enables the system under test to operate, to a certain degree, as it would under actual conditions.

The general sequence of tasks to perform when conducting a test is as follows:

1. Plan the test.

   a. Define the objectives.
   b. Prepare the test plan.

2. Configure the system.

   a. The actual system used to run WSim (physical configuration).
   b. The simulated network (logical configuration).

3. Prepare testing scripts.

   a. Prepare network definitions for the network to be simulated.
   b. Prepare message generation decks.

4. Run the test.
5. Analyze the results from WSim.

Planning is a very important task in this process. You should view it as an ongoing task and be prepared to refine the plan until desired results are achieved. Every test, especially when you begin using WSim, should start with a small sample network definition and a small simple message generation deck. After successful runs, more refined system configurations and more complicated scripts should be prepared until the simulation is done for the complete network to be tested.

7.1.2 Resources WSim can simulate

WSim can simulate the following three types of resources:

- System Network Architecture (SNA) logical units (LU) running as Virtual Telecommunications Access Method (VTAM) applications
- Common Programming Interface for Communications (CPI-C) transaction programs (TP)
- Transmission Control Protocol/Internet Protocol (TCP/IP) clients using Telnet 3270, 3270E, 5250, Network Virtual Terminal (NVT), File Transfer Protocol (FTP), or Simple TCP and UDP protocols attached to the TCP/IP network via the IBM TCP/IP for Multiple Virtual Storage (MVS) product

7.1.3 Testing with WSim

WSim can be used to conduct several types of tests:

- Function
- Regression

- Performance
- Stress
- Capacity planning

Function test is usually used to test a particular function of the system and answers the question, “Is it working correctly?” WSim can be used to test functions such as new application transactions, logon and logoff procedures, error transactions, new hardware additions, and new software products. The scripts used in functional tests can be saved and reused later for regression or stress tests.

Regression test verifies that old functions operate correctly after the addition of new ones, or after any other changes to the existing system, and answers the question, “Is it *still* working correctly?” Advantages of using WSim for regression tests include:

- Scripts are repeatable. Once created, a script can be reused many times until the tested functions are changed.
- WSim can be run automatically. Execution parameters and operator commands that control WSim operation, including ending the simulation after completing all of the test cases, may be included in scripts.

Performance testing includes taking measurements, changing parameters, and then taking measurements again; it answers the question, “How well does it perform?” WSim can be used to report terminal response times and it also provides the possibility to create a controlled, repeatable transaction load for the system under test.

Stress testing is performed when you need to find problems in interactions and resource contentions. By loading the system under test with extremely high transaction rates you can answer the question, “What will break first?” This type of test is almost impossible to conduct without a special tool. WSim can generate controlled message traffic at controlled rates.

Capacity planning helps to predict how the system under test will behave when new resources are brought online or when one or more of the existing resources are overused. This type of test helps to determine if the system under test still performs adequately under predicted increased load, and answers the question, “What will happen if this many resources are added?” WSim can drive the system under test with a higher than normal transaction rate and simulate additional terminals or different types of terminals.

When conducting performance, stress, and capacity planning tests, WSim should be run on a separate host from the system under test to avoid an impact on the results.

7.2 System configuration

The following terms are used in this chapter:

- **Logical unit (LU)** This is a port through which an end user accesses an SNA network to communicate with another end user or the system services control point (SSCP).
- **Transaction program (TP)** In WSim, this is any program using LU6.2 communications protocols to communicate with another program. WSim implements TPs using CPI-C.
- **Session** A session is a logical connection enabling two network addressable units to communicate with each other, such as an LU-LU, or an SSCP-LU session. Each half of a session is a half-session.

Both the network to be simulated and the system to be used to run WSim must be configured before testing. The configuration of the network containing the resources to be simulated by WSim and the real system to be tested (the system under test) is known as the *logical*

configuration. For each logical configuration, a specific *physical configuration* must be used, which is the configuration of the real system used to run WSim. Resources of a physical configuration include a host processor, system software, application software, WSim, and so forth.

7.2.1 Physical configurations

WSim can operate in either one of two basic physical configurations:

- **VTAM and CPI-C application configuration**

  Used to simulate LUs in the same subarea as VTAM. An LU can have a session with any other LU that VTAM will allow it to start with. It is also used to simulate client and server CPI-C TPs in the same subarea as VTAM. TPs can have a conversation with any other TP on any LU to which VTAM will allow a conversation to be started.

  This physical configuration contains WSim, VTAM, and VTAM applications or TPs under test. WSim runs as a VTAM application program.

- **TCP/IP application configuration**

  Used to simulate Telnet 3270, 3270E, 5250, NVT, and FTP clients. These clients can have a session with any Telnet 3270, 3270E, 5250, NVT, or FTP server that TCP/IP allows. This configuration can also be used to simulate Simple TCP or UDP clients in session with various servers.

  This physical configuration contains WSim, TCP/IP, and TCP/IP applications under test. WSim runs as a TCP/IP application program.

7.2.2 Logical configurations

WSim can operate in one of three basic logical configurations:

- **VTAM application configuration**

  Used to simulate SNA LUs accessing VTAM applications. LUs could be terminals or other VTAM applications.

  This logical configuration contains VTAM, VTAM applications, and VTAM applications and LUs simulated by WSim.

- **CPI-C application configuration**

  Used to simulate CPI-C client (allocates outbound conversations but does not accept inbound ones), TPs to test server (accepts inbound conversations), CPI-C TPs and network resources, or to simulate server CPI-C TPs to test client prototypes.

  This logical configuration contains VTAM, VTAM applications, and VTAM application CPI-C TPs and LUs simulated by WSim.

- **TCP/IP application configuration**

  Used to simulate TCP/IP clients in a TCP/IP network, or simple TCP or UDP clients accessing an application through a TCP/IP server.

  This logical configuration contains a TCP/IP server and any Telnet 3270, 3270E, 5250, NVT, FTP, simple TCP, and simple UDP clients simulated by WSim.

7.3 Script preparation

Once system configuration is defined, the definition of the network to be simulated is next. This is done by creating a script. Scripts contain two parts:

- Network definition statements to describe the devices to be simulated by WSim
- Message generation decks to define messages to be sent by the simulated resources to the system under test

7.3.1 Network definition statements

Network definition statements specify the following information:

- Types of the simulated resources on the network
- Attributes of the simulated resources
- Connections between the simulated resources and the system under test
- Special information about delays, logic tests, the order in which message generation decks are used, logging, or tracing of the messages, and so forth.

NTWRK is always the first statement used to define a network. It names the network and specifies characteristics that apply to the network as a whole. It also may specify operands that establish defaults for lower-level statements. All other statements in the network definition follow the NTWRK statement in a prescribed order. The statements from the general simulation statements group immediately follow the NTWRK statement.

Different statements are used depending on what type of network is being simulated. For example, when simulating LUs accessing VTAM applications, the VTAMAPPL and LU statements must be used; when simulating CPI-C TPs, APPCLU and TP statements; when simulating TCP-IP clients, TCP/IP and DEV statements must be used.

Not all statements are mandatory and some could be coded more than once, but all the statements in each and every group, including the optional ones, should follow the prescribed order.

7.3.2 Message generation decks

Message generation is the process by which terminals send and receive messages. Message generation decks are used to control messages being sent out and actions taken when messages are received by a simulated terminal.

A message generation deck contains one or more statements used to generate messages, set delays, define logic tests, define and control event actions, save data for future use, and so forth.

Any terminal can use one or more message generation decks in any order.

Preparation of message generation decks involves the following steps:

1. Decide what transactions to test.
2. Decide which application files and what data to use.
3. Create message generation decks using one of the available methods.
4. Combine created message generation decks with network definition statements to form a script.
5. Test the script and modify and revise if needed.

In WSim, a *transaction* is an exchange of data between a simulated resource and the system under test. The choice of transactions depends on the objectives of the test. Usually, it is not necessary to test all possible transactions in the application. Criteria for the inclusion can be:

- Transactions taking the most processor time
- Transactions generating the most messages
- Transactions being the most important in the application

The following items also should be considered:

- The content of the messages to be sent
- The messages expected to be received
- The mix of transactions, such as the order in which WSim executes the message generation decks and which terminals use which decks
- The transaction rate

Use the PATH statement to specify the order in which the decks are executed and the PATH operand on the DEV, LU, and TP statements to specify which paths a specified simulated resource will execute.

Example 7-1 represents a fragment of the script for the RESNET1 network. The path SMALL specifies that the deck LOGON will be executed before the deck LOGOFF by the LU TERM1. The path LONG specifies that the decks LOGON, ALLOC, BROWSE, and LOGOFF will be executed in this order by the LU TERM2.

Example 7-1 Script fragments for the RESNET1 network

```
RESNET1 NTWRK
    .
    .
SMALL PATH LOGON,LOGOFF
LONG PATH LOGON,ALLOC,BROWSE,LOGOFF
    .
    .
TERM1 LU PATH=(SMALL)
TERM2 LU PATH=(LONG)
    .
    .
```

WSim executes the paths repeatedly, that is, when the terminal has executed the last deck in the path which is defined for it, it starts again with the first deck in its path. Terminals maintain their positions in the paths and are not affected by other terminals. BRANCH, CALL, and IF statements can be used to alter linear sequences of paths. The order in which WSim executes decks in any path can be certain, random, or based on probability distribution.

WSim can generate messages with controlled intermessage delays. This can be used to simulate the delays of real operators as they view the screen, think about the information, enter more data, or even have a cup of coffee. Intermessage delays can be defined for the entire network, a specific resource, or even on a message-by-message basis.

Before starting with the creation of message generation decks, the transactions to be tested should be thoroughly analyzed. All steps should be listed.

7.3.3 Methods for creating message decks

WSim provides several methods for creating message generation decks:

- Directly written message generation statements
- Programs in Structured Translator Language (STL)

- Use of one of the script generating utilities provided with WSim to convert captured data traces

WSim provides the following script generating utilities:

- Interactive Data Capture (can produce STL programs)
- Script generator utility
- SNA 3270 Reformatter Utility

The method used depends on what is being tested and on the following:

- Familiarity with WSim
- What kind of messages will be sent to the system under test by WSim

It makes sense to trace actual system activity and use the Script Generator utility to convert the trace records if the test involves simulating a number of real users using an application.

Some or all of the methods may have to be used when preparing real tests.

Writing message generation statements

Knowledge of message generation statements is very important when interpreting the output from the STL translator, and when using the script generation utilities and debugging the scripts.

The message generation statements have to be coded by hand in situations such as these:

- When modifying the output from one of the script generation utilities
- When adding additional message generation decks in a script produced by the STL Translator or one of the script generation utilities
- When adding some special types of messages or special conditions in an SNA network
- When modifying already existing message generation decks

The syntax for message generation statements is similar to that for network definition statements.

Use the preprocessor to check the syntax and store message generation statements in data sets for use in simulations.

Using STL and the STL Translator

STL is a high-level structured programming language that can be used to create message generation decks and define terminals and devices to be simulated by WSim. STL utilizes constants, variables, expressions, and structured control statements.

An STL program is usually divided into one or more procedures. The STL Translator translates STL programs into message generation decks, each message generation deck corresponding to one STL procedure. Network definitions can be included in STL programs.

The STL Translator invokes the preprocessor to validate and store the network definition statements.

Example 7-2 shows two very simple STL procedures. Procedures begin with a MSGTXT statement and end with an ENDTXT statement.

Example 7-2 Message generation decks written in STL

```
/* STL procedure logging terminal on to RESAPPL */
Logon: Msgtxt
Initself('RESAPPL')
Endtxt
/* STL procedure testing message generation */
Tstmsg: Msgtxt
Do i = 1 to 5
Type "Hello, I expect you to respond Hi"
Transmit using PF4,
and Wait until on substr(screen,40,2) = "Hi"
End
Endtxt
```

The first STL procedure, named LOGON, defines the text that a terminal will use to log on to an application named RESAPPL. When the second STL procedure, named TSTMSG, is executed, WSim simulates a user typing "Hello, I expect you to respond Hi" and then pressing PF4 to send the message to the application. WSim will wait for the application response "Hi" to appear at position 40 on the screen. These messages will be sent five times.

The STL Translator can be invoked using JCL, a TSO CLIST, or the WSim/ISPF interface.

Using the Interactive Data Capture Utility

The Interactive Data Capture Utility (IDC) ITPIDC is a host application that can capture 3270 device session data and generate scripts. A user logs on the same way as for any other VTAM application, and through it can log on to the VTAM application to be tested and perform all the actions to be simulated by WSim. IDC capturing the session traffic is transparent to the VTAM application.

IDC can directly generate, from the captured session data, an STL program, WSim message generation decks, or both.

Using the script generator utility

The script generator utility creates message generation decks based on traces of real users using real applications. The captured trace must be put in a specified format and sorted by resource name, date, and time. The sorted trace is used as an input for ITPSGEN, which actually generates the message generation decks.

The methods that can be used to obtain a system activity trace are:

- The NetView® Performance Monitor (NPM) capturing path information units for selected LUs
- The Generalized Trace Facility (GTF) capturing the VTAM Buffer Trace
- User-written capture routines

WSim provides a special program ITPVTBRF to help with reformatting traces, which are not in the format required by ITPSGEN.

ITPSGEN also requires complete, syntactically correct network definitions as input. It uses the network definition statements to determine the terminal names for which to generate the decks. The names in the DEV and LU statements must correspond to the resource names used in the trace.

Using the SNA 3270 Reformatter Utility

The SNA 3270 Reformatter Utility (ITPLU2RF) is a batch utility for reformatting NPM log records (FNMVLOG) from LU2 sessions into log records. ITPLSGEN can be used to create STL programs or message generation decks based on ITPLU2RF output.

7.3.4 Testing scripts

The scripts must be tested to ensure that they are coded correctly, and that they function as intended. Statement syntax can be checked by using the Preprocessor or the STL Translator. To ensure that the message generation decks function as intended, the following methods can be used:

- Message trace records tracing the message generation process
- STL trace records tracing the message generation process for STL programs
- Self-checking scripts

To ensure that unexpected situations encountered during simulations are handled properly, use self-checking scripts. They do not need to be used for all simulations; for example, they can be skipped for short and simple ones. On the other hand, they definitely should be considered for a long-running test, which could be wasted if terminals were to go out of synchronization.

IF statement logic tests are added to the scripts to check for the expected response and to take action if an unexpected one is received. This action could be simple, such as stopping the device or complex, or including several possible courses of action based on the actual response. The logic tests can be written in decks created by STL, one of the script generation utilities, or manually. They can also be coded in network definitions.

7.4 WSim output

WSim provides several online and printed reports to analyze test results. Some reports are produced by default, while some must be requested by issuing specific operator commands or running one of the WSim utilities. The types of reports are the following:

- Operator reports indicating what is happening during operation
- The complete message log
- Reports generated by the following utilities based on the message log:
  - Formatted reports produced by the Loglist Utility
  - Reports on differences between 3270 display records in two message logs produced by the log compare utility
  - Detailed statistical analysis of response times produced by the response time utility
- Online response time statistics

Most of the reports are intended to represent how WSim is interacting with the system under test and not the effectiveness of the network or the application.

Interval reports monitor the current activity and status of each simulated resource in the network. The statistics are accumulated until the network is canceled or reset. End of run reports provide summary data from the simulated network. They are produced automatically and have the same format as interval reports. The inactivity report contains information about each inactive resource in the network.

The log data set is the single most valuable tool for debugging the scripts. This data set contains all data that has been transmitted or received by the WSim simulated resources. The

message logging facility is active for the entire network, but it can be deactivated completely or just partially for a VTAMAPPL in the network. A separate log data set can be used for a particular network. This is convenient when running multiple networks since the results will be logged separately.

The loglist utility uses the log data set. The control commands can be contained in a file or, as an alternative, entered at the operator console. Use the WSim/ISPF interface, JCL, or TSO CLIST to start the utility, to name the input files, and to specify where the formatted log will be printed.

The loglist utility uses different formats for each type of log records. One particularly useful feature is the printing of screen image records. These images are updated each time a message is sent or received by the device. The output from the Loglist Utility for this type of log records looks the same as the screen images a user would see at the real device.

7.5 Operating WSim

WSim can be run using JCL, TSO CLIST, or by using the WSim/ISPF Interface.

The sample JCL can be found in the WSIMPRC6 member of the data set HLQ.SITPSAMP. The sample TSO CLIST can be found in the member WSIMRUN of the data set HLQ.SITPCLS. The value of HLQ and the method to invoke the WSim/ISPF Interface depend on how WSim and this interface are installed on the site. A typical WSim/ISPF main panel is presented on Figure 7-2.

```
                    Workload Simulator (WSim)

Select one of the following.  Then press Enter.

      Command    Action
__  1. STL       Create and Process Networks and STL Programs
    2. PREP      Create and Preprocess Networks and Message Decks
    3. IDC       Interactively Capture and Build Message Decks and STL Programs
    4. GENERATE  Generate Message Decks, STL Programs, and WSim Logs

    5. RUNWSIM   Run WSim (Prepare to Run a Simulation)

    6. LOGLIST   Analyze Logged Data
    7. RESPONSE  Analyze Response Times
    8. COMPARE   Compare Logged Display Data

    9. SCREEN    Change Screen Characteristics
   10. SETUP     Change System Defaults


 5655-I39 (C)  Copyright IBM Corporation 1976, 2002.  All rights reserved.
```

Figure 7-2 WSim/ISPF main panel

7.6 WSim Test Manager

The WSim Test Manager (WTM) is a usability enhancement that provides guidance through the test process. WTM offers selectable modes of operation, test management services, automatic script generation and task automation.

The primary concept of the WTM testing structure is a project, which is a set of libraries containing schedules and test scenarios. Projects can be archived and reused. A project must be created before any schedules or test scenarios can be created using WTM.

A WTM schedule is a WSim network definition and the associated test scenario definition.

Test scenarios are organized into three levels: a test case, test group and test cycle. A test group is an ordered list of test cases. Test cases can be reused within multiple test groups. A test cycle is an ordered group of test groups and test cases.

WTM offers various ways to automate the development of test cases, which are WSim scripts written in STL. For 3270 environments, WTM can automate the script generation process from 3270 screen/keyboard captures (IDC), SNA traces, WSim or IDC logs, or from STL models and skeletons. Automated CPI-C test case generation uses SNA traces.The STL source is automatically translated into WSim MSGTXTs.

Generated test cases are paired with network resource definitions as part of developing WTM schedules. The WTM schedule is used by WTM to define and control the WSim simulation run (test). WTM schedules can be archived and reused.

The typical WTM main panel is shown in Figure 7-3.

```
                              WSim Test Manager

Select one of the following.  Then press Enter.

      Command   Action
_   1. CASE     Create and Process Testcases
    2. GROUP    Create and Process Testgroups
    3. CYCLE    Create and Process Testcycles
    4. RUN      Create WSim Networks and Schedule WSim Simulation Runs
    D. DOC      Create Test Documentation
    P. PROJECT  Add/Change Project or Alternate HLI
    U. UTIL     Run WSim Test Manager Utilities
    W. WII      Invoke WSim/ISPF Interface

Project:                                Alternate HLI:




Command ===>
PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap      10=Left     11=Right    12=Retrieve
```

Figure 7-3 Typical WTM main panel

7.7 Latest enhancements

Applying the PTF which fixes the APAR PQ94132 for the Workload Simulator provides several general enhancements to this tool, two of them of a high significance:

- Password masking on formatted 3270 screens
- WSim Adapters for Rational TestManager

Passwords are usually maintained on the 3270 screens in unprotected non-display fields. While not visible, the passwords are sent to host applications in the clear and so are captured by the Interactive Data Capture utility or generated by script generation utilities.

The enhancement will mask passwords by encrypting or hiding them using asterisks in test scripts and logs. The utility ITPGNKYZ is supplied to generate required USERMODE.

The Workload Simulator Adapters for Rational TestManager allow WSim Test Manager schedules and JCL scripts to be launched form the IBM Rational TestManager running on a remote workstation.

To run WTM schedules from Rational TestManager, the schedules must already exist in WTM on the host system. Some migration steps must be performed first. Also, a user ID and a password for a TSO user and the user ID of the WSim user (who created WTM projects and schedules) are required to run the WTM schedules from the Rational TestManager.

The white paper IBM Workload Simulator Adapters for Rational TestManager Version 1, Release 1.0.1 and the install program are included with the PTF.

Part 3

Application Monitor for z/OS

In this part we describe how to customize and use Application Monitor (AM). It is divided into four chapters:

- How to set defaults for Application Monitor
- How to select a job to monitor
- How to monitor the job
- How to monitor applications accessing subsystems:

Note: All the screen shots come from the Application Monitor installation verification programs (IVP) or from scenarios described in Part 8, “Scenarios” on page 805.

How to navigate in and use Application Monitor

Before exploring Application Monitor, it is recommended that you review the following hints.

Use the ISPF jump function on all AM panels

It is possible to navigate quickly from one panel to another while using Application monitor. For example, if you are currently viewing the “Primary Options” panel and you want to go to the “Specify Your Default Data Set Names” panel**,** which is one of the panels two levels down from your current location, you can step through the menu hierarchy, going to the interim panel, “Set your Defaults,” and from there making another selection to go to the actual panel of interest; or you can use a bit of navigational shorthand to get there in a single jump.

The long way of getting there is:

1. On the Option ==> or Command ==> line on the Primary Options menu, type **6** and press Enter. This takes you to the Set Your Defaults panel.
2. On the next Option or Command line type **2**, then press Enter. You are now at the panel of interest, the “Specify Your Default Data Set Names” panel.

The quick way to get to the same place is:

1. From *any* panel, enter **=6.2** and press Enter. This takes you directly to the Option 6.2 panel, ““Specify Your Default Data Set Names.”

This shortcut is especially useful once you are familiar with the AM panel hierarchy.

Use any cursor selectable light blue field on any AM panel

Any field that appears in light blue is either an option in a menu, an action in a menu (Go back, Save, and so forth), or a value in a report used to drill down to a more detailed Application Monitor display. These light blue fields are selectable with your cursor.

For example, if you are currently viewing the Primary Options panel and you want to change your defaults, move the cursor to the “6 Set Your Defaults” line and press Enter. You are taken directly to the Option 6 panel, Set Your Defaults.

Screen configuration

Our recommendation is to configure your screen with 32 lines x 80 columns. Then you will not have to use PF7 (scroll up) and PF8 (scroll down) in order to view all the available options on each panel.

8

Application Monitor: Settings and customization

Each application developer, or group of application developers can have their own listings data sets, their own history data set naming convention, their own DB2 and their own rules to use Application Monitor for z/OS. This chapter describes how to customize Application Monitor (AM) to work with your own environment.

The defaults we discuss here are stored in your ISPF profile, in member *tso-userid*.ISPF.ISPPROF(FBIPROF).

8.1 Application Monitor Primary Options panel

Figure 8-1 shows the Primary Options panel. You can access all AM functions from this panel.

There are two ways to select an option with Application Monitor: either enter the option number on the command line and press Enter; or move your cursor to the option's line and press Enter. (See "How to navigate in and use Application Monitor" on page 95 for more tips on navigating in AM.)

```
-------- IBM Application Monitor for z/OS - Primary Options ------------------

Option ===> _____________________________________________________________


   1 Select an Active Job to Analyze
   2 Analyze a Specified Job
   3 Analyze a Named Set of Jobs
   4 Schedule a Deferred Monitoring Request
   5 Analyze Historical Data
   6 Set Your Defaults
   7 Administration
   X Exit


     Copyright(c) 2000-2004 - SERENA Software, Inc.  All Rights Reserved.

Enter END (PF3) to exit




                                                                      03/014
```

Figure 8-1 Primary Options

8.2 Application Monitor Set Your Defaults panel

This chapter provides a detailed discussion of option 6 from the Primary Defaults panel, the **Set Your Defaults** panel, which is shown in Figure 8-2.

The defaults you specify are stored in your ISPF profile and are used for all monitoring sessions that you initiate through Primary Option 1 Select an Active Job to Analyze.

If you initiate a monitoring session with Primary Option 2 Analyze a Specified Job, and Primary Options 4 Schedule a Deferred Monitoring Request, you can override these defaults except for the name of the compilation listing data set and the prompt-to-keep option for the historical data set.

```
-------- IBM AM for z/OS - Set Your Defaults ---------------------------------

Option ===> _____________________________________________________________


   1 Specify Default Sampling Parameters
   2 Specify History Names and Options
   3 Specify History Allocation Options
   4 Specify COBOL and PL/1 Listing Data Sets
   X Go back (PF3)



Enter END (PF3) to go back



                                                                    03/014
```

Figure 8-2 Set Your Defaults

Use the Set Your Defaults panel as follows:

- Select Option 1 to specify defaults for:
  - Maximum duration of a monitoring session
  - Maximum number of samples to be taken
  - Whether to collect DB2 analysis data and the DB2 subsystem ID
  - Whether to collect delay analysis data
  - Whether to collect data set analysis data
- Select Option 2 to specify:
  - A pattern for naming history data sets
  - If you want to be prompted to keep the history data set that is created during a monitoring session
  - If you do not want to allocate a history file during real-time monitoring
- Select Option 3 to specify allocation options for the history file.
- Select Option 4 to specify the names of the COBOL and PL/I compiler-generated listing data sets.

8.2.1 Specify Default Sampling Parameters

Select option **1** from Set Your Defaults (Panel id FBIP6000). In response, Application Monitor displays the **Specify Default Sampling Parameters** panel shown in Figure 8-3.

```
-------- IBM AM for z/OS - Specify Default Sampling Parameters ---------------

Option ===> ____________________________________________________________________


Update the defaults below and then press PF3 or select an option

   Max Duration ==> 90       (1-999 minutes)
   Max Samples  ==> 10000    (100-99999999)
   Analyze DB2 activity for every job   _   Default DB2 SSID==> ____
   Analyze Delays for every job         /
   Analyze Data Sets for every job      /

   1 Reinstate your profile defaults
   2 Clear your defaults to built-in default values


Enter END (PF3) to save your updates and go back




                                                                    22/015
```

Figure 8-3 Specify Default Sampling Parameters

Here is the relevant information about each field:

Max Duration The maximum number of minutes for which Application Monitor can collect information.

Max Samples The maximum number of samples that can be taken during an analysis session. If you have to monitor a long running application, the build-in default of 10000 is much too small. We recommend setting it to around 10000 per minute.

Note: The monitoring session stops collecting analysis data when Max Duration expires, Max Samples have been taken, or the job completes, whichever occurs first.

DB2 Activity Type the **/** character in the DB2 selection field to request that Application Monitor collect DB2 data for every job.

Default DB2 SSID Enter the DB2 subsystem ID to identify the DB2 subsystem from which Application Monitor is to collect the analysis data.

Delay Analysis Type the **/** character in the Delay Analysis selection field to request that Application Monitor collect delay analysis for every job.

Data Set Analysis Type the **/** character in the Data Set Analysis selection field to request that Application Monitor collect data set analysis data for every job.

Note: The fields enable Application Monitor to collect extra information. If all or a subset of these fields are selected, an additional overhead will be incurred to gather this data.

Use this panel as follows:

- Select option **1** to reinstate the defaults that are currently stored in your ISPF profile.
- Select option **2** to clear your defaults and reinstate the built-in defaults.
- Press PF3 to save your updates in your ISPF profile and return to the prior panel.

8.2.2 Specify History File Name and Options

The analysis data that Application Monitor captures during each monitoring session is written to a history data set. We recommend that you request a separate history data set for each job that you monitor.

History data set analysis gives you more control than real-time analysis over the displays of performance data that Application Monitor collects during the monitoring session. With real-time analysis, the contents of the analysis displays are refreshed only when you press Enter during the online monitoring session. That restricts your ability to analyze the data in depth and over specific time intervals.

When you select option **2** from Set Your Defaults (Panel id FBIP6000), AM displays the **Specify History File Name And Options** panel as shown in Figure 8-4.

```
-------- IBM AM for z/OS - Specify History File Name And Options -------------

Option ===> _____________________________________________________________________


Specify history file options below and then press PF3 or select an option


   History Prefix     ==> CHABERT.AM_____________________________________
   Optional Suffix        _ date.time
                          / jobname.date.time
                          _ system.jobname.date.time
                          _ tsopref.jobname.date.time

   Prompt to keep         / (See prompt to keep history data sets)
   No History File        _ (Real time only - No history allocated)


   1 Reinstate your profile defaults
   2 Clear your defaults to built-in default values

Enter END (PF3) to save your updates and go back


                                                                        03/014
```

Figure 8-4 Specify History File Name And Options

Here is the relevant information about each field:

History Prefix High-level prefix for the historical data set name. The built-in default is TSO variable *&zuser* with .AM appended (*&zuser.*AM). *&zuser* has the value of the TSO userid you give when you log on to TSO.

> **Note:** You must have the RACF® authorization to create the historical data set with the History Prefix value specified; otherwise, Application Monitor always displays a RACF error message when you start to monitor a job, and no data will be saved.

Optional Suffix An optional suffix for the historical data set name. The variables used in the suffix patterns are:

- **date** System-generated date in format D*yyyynnn.*
- **time** System-generated time in the format T*hhmmsst*, where *t* is tenths of second.
  Application Monitor uses the current Local Time for the date and time.

jobname Name of the job or named set that is monitored.

system Name of the system where the monitoring takes place. “system” is useful in a SYSPLEX environment when you cannot predict where the job will start.

tsopref The value of the TSO variable *&zprefix*.

> **Note:** Be sure to specify a naming convention that creates a unique name. A VSAM error occurs if Application Monitor tries to create a history data set that has the same name as an existing one.

Prompt to keep If you select this option, Application Monitor asks you if you want to keep the historical data set every time you finish a real-time analysis session. If you don’t select it, Application Monitor keeps all historical data sets without prompting you.

> **Note:** Periodically, you need to delete the historical data sets that you no longer need.

No History file If you select this option, Application Monitor will not create a history file when monitoring in real time.

Use this panel as follows:

- Select option **1** to reinstate the defaults that are currently stored in your ISPF profile.
- Select option **2** to clear your defaults and reinstate the built-in defaults.
- Press PF3 to save your updates in your ISPF profile and return to the prior panel.

8.2.3 Specify History Allocation Options

When you select option **3** from Set Your Defaults, AM displays the **Specify History Allocation Options** panel shown in Figure 8-5.

```
-------- IBM AM for z/OS - Specify History Allocation Options ----------------

Option ===> ____________________________________________________________________


Specify default values for VSAM history file allocation

   Space Units      ==> / Records     _ Tracks      _ Cylinders
                        _ Kilobytes   _ Megabytes

   Space Values     ==> 4000    (Primary)  1000    (Secondary)

   Model Data Set   ==> ____________________________________________
   Volume(s)        ==> ______    ______    ______    ______    ______
   Management Class ==> ________
   Storage Class    ==> ________
   Data Class       ==> ________

   1 Reinstate your profile defaults
   2 Clear your defaults to built-in default values


Enter END (PF3) to save your updates and go back

                                                                      03/014
```

Figure 8-5 Specify History Allocation Options

Here is the relevant information about each field:

Space Units Type of space units used to allocated the history data set.

Space Values The number of space units used for allocating the history data set. A primary and secondary value may be specified.

Model Data Set If specified, Application Monitor uses the values for a MODEL() parameter when allocating the history data set.

The Model Data Set is used only when Space Units and Space values are blank.

Volume(s) If specified, Application Monitor uses the values for a VOLUME() parameter when allocating the history data set.

Volumes values are ignored if your installation uses SMS.

Management Class If specified, Application Monitor uses the value for a MANAGEMENT CLASS() parameter.

Storage Class If specified, Application Monitor uses the value for a STORAGECLASS() parameter.

Data Class If specified, Application Monitor uses the value for a DATACLASS() parameter.

Use this panel as follows:

- Select option **1** to reinstate the defaults that are currently stored in your ISPF profile.
- Select option **2** to clear your defaults and reinstate the built-in defaults.
- Press PF3 to save your updates in your ISPF profile and return to the prior panel.

8.2.4 Specify COBOL and PL/I Listing Data Sets

When you select option **4** from the Set Your Defaults panel, AM displays the **Specify COBOL and PL/I Listing Data Sets** panel shown in Figure 8-6.

```
-------- IBM AM for z/OS - Specify your Listing Data Set Names ---------------

Option ===> _____________________________________________________________________


Specify listing data set names below and then press PF3 or select an option


   COBOL Listings     ==> 'CHABERT.FBI.TEST.COBLIST'
   PL/1 Listings      ==> 'CHABERT.FBI.TEST.PLILIST'


   1 Reinstate your profile defaults
   2 Clear your defaults to built-in default values


Enter END (PF3) to save your updates and go back




                                                                      03/014
```

Figure 8-6 Specify your Listing Data Set Names

Here is the relevant information about each field:

COBOL Listings The default name of the data set where the COBOL compiler listings for the job reside.

PL/I Listings The default name of the data set where the PL/I compiler listings for the job reside.

Specify these parameters if you want Application Monitor analysis data to reference the program listing generated by the compiler in the **Statement Analysis** display.

Enter the fully qualified partitioned data set name, quotation marks are not required.

Use this panel as follows:

- Select option **1** to reinstate the defaults that are currently stored in your ISPF profile.
- Select option **2** to clear your defaults and reinstate the built-in defaults.
- Press PF3 to save your updates in your ISPF profile and return to the prior panel.

Note: Application Monitor displays the statement in source code (if it is a COBOL or PL/I language) to make your monitoring activities easier if the statements offsets in the program listing match the offsets of the instructions in the load module. Consequently, some compiler options have to be used:

COBOL compiler options

The COBOL programs must be compiled with the following options:

SOURCE,NOOPTIMIZE,OFFSET

or

SOURCE,NOOPTIMIZE,LIST

PL/I compiler options

The PL/I programs must be compiled with the following options:

SOURCE,NOOPTIMZE,LIST,NOCOMPACT

9

Application Monitor: Selecting a job

This chapter describes how to select a job or a set of jobs (called a Named Set), that has to be monitored. Hereafter we use “job” to refer to either for a single job or a Named Set. Section 9.4, “Managing and analyzing a named set of jobs” on page 113 describes why and how to use a Named Set.

Jobs selected can be active jobs, or they can be jobs that will execute in the future. This chapter describes how to select active jobs and how to create and administer deferred requests for jobs that will execute later.

For both active jobs and deferred requests, Application Monitor stores the data that it captures during the monitoring session in a history data set. While active jobs can be analyzed in real time, this data set is used to analyze deferred requests. It also can be used or reused in the future to review active job or deferred job information.

9.1 Active job selection

When you monitor in real time, the analysis data you see on Application Monitor displays is refreshed each time you press Enter. An alternative is to use the CYCLE command to periodically refresh the screens. The refreshed display reflects, in addition to the data available the last time you pressed Enter, all data obtained by Application Monitor since that time.

This analysis data can also be written to a history data set, giving you the opportunity to analyze it later at your own pace.

To select an active job to monitor in real time with the current default monitoring options, from the Application Monitor **Primary Options** menu, choose one of the following options as shown in Figure 9-1:

- Option **1 Select an Active Job To Analyze** (9.2, “Select an Active Job to Analyze” on page 106)
- Option **2 Analyze a Specified Job** (9.3, “Analyze a Specified Job” on page 110)
- Option **3 Analyze a Named Set of Jobs** (9.4, “Managing and analyzing a named set of jobs” on page 113)

```
-------- IBM Application Monitor for z/OS - Primary Options ------------------

Option ===> _____________________________________________________________


   1 Select an Active Job to Analyze
   2 Analyze a Specified Job
   3 Analyze a Named Set of Jobs
   4 Schedule a Deferred Monitoring Request
   5 Analyze Historical Data
   6 Set Your Defaults
   7 Administration
   X Exit


     Copyright(c) 2000-2004 - SERENA Software, Inc.  All Rights Reserved.

Enter END (PF3) to exit




                                                                      03/014
```

Figure 9-1 Primary Options

9.2 Select an Active Job to Analyze

When you select primary option 1, Application Monitor displays the Select an Active Job to Analyze panel as shown in Figure 9-2.

```
-------- IBM AM for z/OS - Select an Active Job to Analyze -------------------
         2004/10/27 14:16:52 to 2004/10/27 14:16:52       Row 1 of 39

System   Name     Typ C I Stepname Procstep Asid   CPU%           I/O   Real
-------- -------- --- - - -------- -------- ---- ------        ------- ------
*        *        *   * * *        *
STLABF6  FBISTART STC     FBISTART FBIAM    001F   0.07 ====>       0 111587
STLABF6  JES2     STC     JES2     IEFPROC  0018   0.06 ===>      600    448
STLABF6  TCPIP    STC     TCPIP    TCPIP    0026   0.02 >           0    751
STLABF6  ATRRRS   STC     ATRRRS   RRS      001E   0.00             0    205
STLABF6  BBODMNB  STC     BBODMNB  BBODAEMN 0072   0.00             0   7067
STLABF6  BBOS001  STC     BBOS001  BBOCTL   006C   0.00             0  32208
STLABF6  CCURUNM  STC     CCURUNM  RUNPGM   0074   0.00             0    359
STLABF6  CCURUNU  STC     CCURUNU  RUNPGM   006D   0.00             0    301
STLABF6  CHABERT  TSU     TPROC02            005B   0.00             0   1215
STLABF6  CICSC22F STC Y   CICSC22F CICS     005E   0.00             0   1611
STLABF6  CICSC31F STC Y   CICSC31F CICS     006B   0.00             0    880
STLABF6  DB1EDBM1 STC     DB1EDBM1 IEFPROC  0062   0.00             0    912
STLABF6  DB1EDIST STC     DB1EDIST IEFPROC  0064   0.00             0     81
STLABF6  DB1EIRLM STC     DB1EIRLM          0015   0.00             0     60
STLABF6  DB1EMSTR STC     DB1EMSTR IEFPROC  0029   0.00             0    136
STLABF6  DB1ESPAS STC     DB1ESPAS IEFPROC  0065   0.00             0     66
STLABF6  IM8FCTL  STC     IMS      IEFPROC  005F   0.00             0    223
Command ===> _____________________________________________ Scroll ==> CSR
                                                                     24/015
```

Figure 9-2 Select an Active Job to Analyze

9.2.1 How to read this panel

This panel is divided into three areas:

Header area

The line immediately below the panel title shows the monitoring interval during which the performance data (CPU%, I/O, and Real) that is shown on the panel was collected.

The header area shows the start date and time and the stop date and time of the monitoring interval. The start time is the time you first brought up this panel or the time you pressed Enter to refresh the display. The end time is the time that the displayed data was collected.

Each time you press Enter, Application Monitor refreshes the header area.

Column title and mask lines

The mask line enables you to filter the content of the rows that appear in the scrollable area of the panel.

Here is the relevant information about the main fields:

System Name of the system where the job is active.

Name Name of the job that is executing in the address space.

Typ Type of job that is executing in the address space. This can be one of the following:

BAT (batch job, IMS region or CICS region)

STC (Non MVS system started task)

TSU (TSO user session)

Application Monitor should not be used to analyze work done for applications in MVS system address spaces (such as JES or VTAM).

C The letter **Y** in the **C** column indicates that the job is an **C**ICS region.

The Application Monitor supplied FBII transaction must be executing in the target CICS region to enable us to monitor transactions.

I The letter **Y** in the **I** column indicates that the job is an **I**MS region.

CPU% Percentage of the total available CPU resources used by the job.

The ===> field following CPU% provides a visual indicator, it enables us to spot the jobs taking the most CPU resources.

I/O Number of MVS I/O Service Units (SUs) used by the job.

SUs are a standard measure of the amount of I/O resource used. It is meaningful to system performance and tuning personnel.

Real Number of 4k pages in real storage used by the job.

Scrollable area

The scrollable area of the panel has a line for each active job on each system where Application Monitor is running.

The scrollable area reflects the data collected during the most recent sample. The analysis data (CPU%, I/O, and Real) is an average of all the samples taken during the displayed monitoring interval.

Each time you press Enter, Application Monitor refreshes the scrollable area. Since Application Monitor sorts the rows in the scrollable area in descending sequence by CPU%, the order may change.

9.2.2 How to select your job from this panel

Tab to the desired Name (which is a cursor-selectable field) and press Enter to select a specific job for analysis. In response, Application Monitor displays the Active Monitor Sessions panel.

Because it can be hard to find the job you want to analyze, from among a huge number of jobs, Application Monitor allows you to change the sort order as well as to use a filter to display rows.

Sorting the rows in the display

You can sort the rows in the scrollable area of the Select an Active Job To Analyze panel in ascending or descending sequence by any column by placing the cursor on the column heading and pressing Enter. This action toggles between ascending and descending sequence.

Rows are initially sorted in descending sequence by CPU%. If you place the cursor on the CPU% column heading and press Enter, the rows are resorted in ascending sequence by CPU%. If you place the cursor on the CPU% column heading and press Enter again, the rows are resorted in descending sequence by CPU%.

Using a mask to filter display rows

You can filter the content of the scrollable area by specifying a mask in any column that shows an asterisk (*) on the mask line.

For example, to restrict display content to show only active CICS regions, type the letter **Y** in the C column on the mask line as shown in Figure 9-3.

```
FBIP1000 IBM AM for z/OS - Select an Active Job to Analyze -------------------
         2004/10/21 18:49:18 to 2004/10/21 18:49:18       Row 1 of 2

System   Name     Typ C I Stepname Procstep Asid   CPU%          I/O   Real
-------- -------- --- - - -------- -------- ---- ------      ------- ------
*        *        *   Y * *        *
STLABF6  CICSC22F STC Y   CICSC22F CICS     005E   0.00            0   7490
STLABF6  CICSC31F STC Y   CICSC31F CICS     006B   0.00            0   4547
****************************** BOTTOM OF DATA ******************************
```

Figure 9-3 Select an Active Job to Analyze

As another example, if you want to restrict display content to only those SCT jobs whose names begin with the letters IM, type `IM*` in the Name column and SCT in the Typ column on the mask line as shown in Figure 9-4.

```
FBIP1000 IBM AM for z/OS - Select an Active Job to Analyze -------------------
         2004/10/21 18:49:18 to 2004/10/21 18:49:18       Row 1 of 6

System   Name     Typ C I Stepname Procstep Asid   CPU%          I/O   Real
-------- -------- --- - - -------- -------- ---- ------      ------- ------
*        IM*      STC * * *        *
STLABF6  IM8FCTL  STC     IMS      IEFPROC  005F   0.00            0   1348
STLABF6  IM8FDLI  STC     IM8FDLI  IEFPROC  0060   0.00            0    475
STLABF6  IM8FDRC  STC     IM8FDRC  IEFPROC  0061   0.00            0    350
STLABF6  IM9FCTL  STC     IMS      IEFPROC  0066   0.00            0   1454
STLABF6  IM9FDLI  STC     IM9FDLI  IEFPROC  0067   0.00            0    423
STLABF6  IM9FDRC  STC     IM9FDRC  IEFPROC  0068   0.00            0    354
****************************** BOTTOM OF DATA ******************************
```

Figure 9-4 Select an Active Job to Analyze

After you type in the mask and press Enter, Application Monitor refreshes the display and lists only those active jobs that satisfy the mask criteria.

These two Application Monitor features (sort and mask) allow you to quickly select the active job that you want to analyze. When your job has been selected, Application Monitor displays the Active Monitor Sessions panel as shown in Figure 9-5.

```
-------- IBM AM for z/OS - Active Monitor Sessions ---------------------------
         2004/10/27 14:19:08 to 2004/10/27 14:19:21       Row 1 of 1

  Named Set: N/A
System   Jobname  Stepname Pgm  CICS IMS  DB2  Dly  Dsn  CPU%   Sample Duration
-------- -------- -------- Rpts Rpts Rpts Rpts Rpts Rpts ------  Count --------
STLABF6  CHABERT  TPROC02     9                YES  YES    0.05     45 00:00:12
******************************** BOTTOM OF DATA ********************************

Command ===> _____________________________________________ Scroll ==> CSR
                                                                     24/015
```

Figure 9-5 Active Monitor Sessions

The next two chapters describe the relevant information for the Active Monitor Sessions panel and its associated reports.

From time to time you might need to be more accurate in specifying the details of the job you want to analyze, or you might need to override the monitoring defaults that are currently in effect. Do this by selecting option **2 Analyze a Specified Job** from the Application Monitor Primary Options panel. This selection is described in detail in the next section.

9.3 Analyze a Specified Job

In response the selection of option **2 Analyze a Specified Job**, Application Monitor displays the panel shown in Figure 9-6.

```
-------- IBM AM for z/OS - Analyze a Specified Job ---------------------------

Option ===> _____________________________________________________________________
                                                                   More:     +
Specify the Job to analyze and then select an option below

   Job Name  ==> COBOLRUN (Full name or mask ending in "*")
                 (Inactive. Press Enter to refresh)

   Batch: Step Name        ==> *_______ Proc Step Name      ==> *_______
   CICS : Transaction Name ==> ________ Transaction Number  ==> ________
   IMS  : Message Id       ==> ________
   Enter / to select special types of analysis:
     Delays         /
     DB2 activity:  _      DB2 Subsystem Id ==> ____
     Data Sets:     /

   History DSN  ==> 'CHABERT.AM.COBOLRUN.D2004301.T1421002'_______

   Max Duration ==> 90_ (1-999 minutes)      Max Samples   ==> 10000___


   1 Analyze the specified Job
   X Go back without analyzing (PF3)
                                                                     03/014
```

Figure 9-6 Analyze a Specified Job

The fields are primed with the monitoring options that you specified for the last job you analyzed with Primary Option 2 except for the name of the history data set (History DSN), which is generated dynamically from the default naming convention specified with Option 6, Set Your Defaults. You can accept or override these values as follows:

Job Name This field is primed with the name of the job you last monitored through Primary Option 2.

You can enter the name of a specific job or a mask ending in an asterisk. If you enter a mask and press Enter, Application Monitor displays the Select an Active Job for Option 2 panel with a list of jobs that match the mask.

Batch Step Name Enter the name of the job step you want to analyze. Application Monitor collects analysis data for only that step. An * (asterisk) is the default value. It means all job steps.

If more than one job step has the same name as the name you specify for this field, Application Monitor monitors only the first step with that name.

Proc Step Name Same value as that for Batch Step Name, but for the procedure you want to analyze.

CICS Transaction Name Enter the name of the CICS transaction that you want to analyze if the Job Name identifies a CICS region. If this field is blank, all transactions will be monitored.

CICS Transaction Number The CICS TaskId of the transaction instance you want to analyze. This field is valid only if you specify a CICS Transaction Name.

IMS Message Id Enter the name of the IMS message you want to analyze if Job Name identifies an IMS dependent region. If this field is blank, all messages that run in the region identified will be monitored.

This field is mutually exclusive with CICS Transaction Name and Number.

The other fields were already explained in Chapter 8, “Application Monitor: Settings and customization” on page 97. They are primed with the monitoring options that you specified for the last job, but each of them can be overwritten with a new value.

Select option **1 Analyze the Specified Job** to begin analyzing the job.

If the job is already active, Application Monitor displays the Active Monitor Sessions panel as shown in Figure 9-7.

```
-------- IBM AM for z/OS - Active Monitor Sessions ---------------------------
         2004/10/27 14:24:40 to 2004/10/27 14:24:53       Row 1 of 1

  Named Set: N/A
System   Jobname  Stepname Pgm  CICS IMS  DB2  Dly  Dsn  CPU%   Sample Duration
-------- -------- -------- Rpts Rpts Rpts Rpts Rpts Rpts ------  Count --------
STLABF6  COBOLRUN STEP01      4                 YES  YES   23.08  2,263 00:00:12
****************************** BOTTOM OF DATA ******************************

Command ===> _____________________________________________ Scroll ==> CSR
                                                                     24/015
```

Figure 9-7 Active Monitor Sessions

The next two chapters explain the relevant information for the Active Monitor Sessions panel and its associated reports.

If the job is not active, Application Monitor displays the message shown in Figure 9-8.

```
FBIPYDF0 IBM AM for z/OS - Prompt for a Deferred Monitor

  Job COBOLRUN, Step (ALL), Procstep (ALL) is
  not active. Press ENTER to schedule a deferred monitor
  request. Press PF3 (END) to Cancel.

  If you submit a deferred monitor request but the job
  does not run within 90 minutes, AM will
  discard the request.

  The samples will be recorded in history data set
  'CHABERT.AM.COBOLRUN.D2004295.T2008160'
  After data collection has completed, you will be able
  to analyze the job using primary option 5 "Analyze
  Historical Data".
Command ===> ____________________________________
 F1=Help    F2=Split   F3=End     F4=Return  F5=Nop
 F6=Nop     F7=Up      F8=Down    F9=Swap    F10=Left
```

Figure 9-8 Prompt for a Deferred Monitor

If you get this message, press PF3 to return to the Analyze a Specified Job panel. If you press Enter, Application Monitor will automatically schedule a deferred monitoring request.

In that request, Application Monitor will watch for Max Duration minutes for the job to become active. If the job does not become active within Max Duration minutes, the deferred request will be discarded. If the job does become active, Application Monitor will monitor the job for Max Duration minutes, or until Max Samples have been taken, or until the job ends, whichever occurs first.

Note: If a deferred request is scheduled, the Max Duration field serves two purposes:

- Max Duration specifies the number of minutes that Application Monitor will watch for the job to become active.
- If the job starts, Application Monitor will then monitor it for Max Duration minutes from the time it starts.

9.4 Managing and analyzing a named set of jobs

A *Named Set* is a collection of jobs that you want to analyze together as a group.

This feature is very useful in tracking, for example, a CICS transaction that can execute on CICSplex configuration. Often, a CICS transaction starts in a Terminal Owning Region (TOR) and may be dispatched to other regions (for example, AORs and FORs) during its execution.

Since you may not know the regions that will process the transaction, you create a named set with an entry for each region in which a transaction could execute. You can launch the named set using Application Monitor Primary Option 3, Analyze a Named Set of Jobs, and Application Monitor will monitor the regions named and collect performance data about the target transaction when it becomes active in those regions.

9.4.1 Managing named sets

To perform the administrative functions (creation, editing, and deletion), select option **7 Administration** from the Primary Options panel as shown in Figure 9-9.

```
-------- IBM Application Monitor for z/OS - Primary Options ------------------

Option ===> _____________________________________________________________


   1 Select an Active Job to Analyze
   2 Analyze a Specified Job
   3 Analyze a Named Set of Jobs
   4 Schedule a Deferred Monitoring Request
   5 Analyze Historical Data
   6 Set Your Defaults
   7 Administration
   X Exit


     Copyright(c) 2000-2004 - SERENA Software, Inc.  All Rights Reserved.

Enter END (PF3) to exit




                                                                    03/014
```

Figure 9-9 Primary Options

In response, Application Monitor displays the Administration panel shown in Figure 9-10.

```
-------- IBM AM for z/OS - Administration -----------------------------------

Option ===> _____________________________________________________________


  1 Administer Deferred Monitoring Requests
  2 Create, Edit, and Delete Named Sets
  X Go back (PF3)



Enter END (PF3) to go back




                                                                    03/014
```

Figure 9-10 Administration

Select option **2 Create, Edit, and Delete Named Sets** to manage named sets as shown in Figure 9-11.

```
-------- IBM AM for z/OS - Create, Edit and Delete Named Sets  Row 1 to 2 of 2


Enter a line command to process a Named Set

  N - New (Create a new Named Set primed from the selected Named Set)
  E - Edit (Update the contents of the Named Set)
  D - Delete (Delete the Named Set)

C Name     Description                          Created             Count
_   ?      Blank Template for new Named Set     2004/10/27 14:28:39     0
_ CICSPLEX CICS TS22 (TOR) CICS TS 31 (AOR)     2004/10/22 17:07:38     2
****************************** Bottom of data ******************************









Command ===> _____________________________________________ Scroll ==> CSR
                                                                   24/015
```

Figure 9-11 Create, Edit, and Delete Named Sets

This panel lists the named sets, (the list is first empty) and enables you to create a new named set.

Here is the relevant information about the main fields:

C Line command field:

- **N** Creates a new named set. If you specify **N** for an existing named set in the list, the next panel provides a template for creating the new set, primed with the contents of the selected model.
- **E** Selects the named set for editing.
- **D** Deletes the named set.

Name Name of the named set.

Count Number of jobs identified in the named set.

Creating or editing a named set

If you enter an N or E line command, Application Monitor displays the Create and Edit Named Set panel as shown in Figure 9-12.

```
-------- IBM AM for z/OS - Create and Edit a Named Set ------- Row 1 to 3 of 8


Fill in the Named Set's details and select an option below

  Named Set Name ==> CICSPLEX
  Description    ==> CICS TS22 (TOR) CICS TS 31 (AOR)


  1 Save the Named Set and exit
  X Go back without saving (PF3)

Enter END (PF3) to go back without saving


Use line commands to define the jobs in the Named Set
A  - Add a new entry      E  - Edit an entry        D  - Delete an entry

C Job      Description
_ CICSC31F CICS TS 31 (AOR - FOR)
_ CICSC22F CICS TS 22 (TOR)
_

Option ===> _____________________________________________ Scroll ==> CSR
                                                                 24/014
```

Figure 9-12 Create and Edit a Named Set

You must enter a unique name for a new set, and you can use the free text line for a description. Enter one of the following values in the line command field:

A Add a new job to the named set.

E Edit an existing entry or create a new entry with the fields primed with the values of that job.

D Delete the selected entry from the named set.

When all the jobs have been created, select option **1 Save the Named Set and exit** and press Enter to return to the Create, Edit and Delete Named Sets panel.

Adding or editing a named set entry

If you specify the `A` or `E` line command for an entry, Application Monitor displays the Specify a Named Set Entry panel as shown in Figure 9-13.

```
-------- IBM AM for z/OS - Specify a Named Set Entry -------------------------


Named Set CICSPLEX - CICS TS22 (TOR) CICS TS 31 (AOR)

Fill in the Named Job details below and then select an option

   Job Name    ==> CICSC22F
   Description ==> CICS TS 22 (TOR)
   Step Name   ==> *       Proc Step Name ==> *
   CICS Tran   ==> ____    or IMS Msg Id  ==> ____
   System      ==> STLABF6
   Enter / to select special types of analysis:
     Delays        /
     DB2 activity: /      DB2 Subsystem Id ==> DB1E
     Data Sets:    /


  1 Save the changes to this Named Set
  X Go back without saving changes (PF3)


Enter END (PF3) to exit without saving
Option ===> ____________________________________________________________
                                                                  24/014
```

Figure 9-13 Specify a Named Set Entry

Here is the relevant information about the main fields:

Job Name The name of the job to be monitored.

If you have to monitor a CICS transaction or IMS message, it is the name of the region where the transaction or message can be dispatched for execution.

System Name of the system where the job executes. This field is required.

If you are working in a sysplex environment and the job can be scheduled to run on any system in the sysplex, we recommend having a separate entry for the job with the name of each system in which the job can execute.

All the other fields are either self-explanatory or have been described previously.

When the definition has been completed, select option **1 Save the changes to this Named Set** and press Enter to return to the Create and Edit Named Set panel to specify the details for the next job in the set.

9.4.2 Analyze a named set of jobs

Once you have created the named set, you can initiate real-time monitoring for the jobs that are identified in it by selecting primary option **3 Analyze a Named Set of Jobs**. Application Monitor displays the Specify a Named Set To Analyze panel as shown in Figure 9-14. This panel lists each named set that you have created.

```
-------- IBM AM for z/OS - Specify a Named Set to Analyze ---- Row 1 to 1 of 1


Enter a line command to process a Named Set:
Go to the Administration screen to Create, Edit or Delete Named Sets

  L Launch (Initiate monitoring of all the Jobs in the Named Set)

C Name     Description                          Created             Count
_ CICSPLEX CICS TS22 (TOR) CICS TS 31 (AOR)     2004/10/22 17:07:38   2
****************************** Bottom of data ******************************









Command ===> _____________________________________________ Scroll ==> CSR
                                                                   24/015
```

Figure 9-14 Specify a Named Set to Analyze

Enter the **L** (Launch) command in the C (command line) field and press Enter to initiate real-time monitoring for the jobs in the target named set.

If all the jobs identified in the selected named set are active when you enter the **L** line command, Application Monitor displays the Active Monitor Sessions panel as shown in Figure 9-15. This panel lists each job identified in the named set. You can select the job you want to analyze in real time from this panel.

```
-------- IBM AM for z/OS - Active Monitor Sessions --------------------------
         2004/10/27 14:36:03 to 2004/10/27 14:36:14       Row 1 of 2

  Named Set: CICSPLEX
System   Jobname  Stepname Pgm  CICS IMS  DB2  Dly  Dsn  CPU%   Sample Duration
-------- -------- -------- Rpts Rpts Rpts Rpts Rpts Rpts ------  Count --------
STLABF6  CICSC31F CICSC31F    3 YES       YES  YES  YES    2.72      8 00:00:10
STLABF6  CICSC22F CICSC22F    1 YES       YES  YES  YES   23.34      7 00:00:10
****************************** BOTTOM OF DATA ******************************









Command ===> _____________________________________________ Scroll ==> CSR
                                                                   24/015
```

Figure 9-15 Active Monitor Sessions

9.5 Deferred monitoring request

Deferred monitoring requests let you schedule monitoring on jobs for which you cannot predict exactly when the job will begin executing, and on jobs run at night or at times when you are not available to initiate real-time monitoring. Deferred requests can be for a specific job or for a named set.

Application Monitor stores the data that it captures during the monitoring session in a history data set. You can access this data set later to review the analysis data. Once you have set up one or more deferred requests, you can manage them (edit or delete them) with primary option **7, Administration**.

9.5.1 How to define a deferred monitoring request

To schedule a deferred monitoring request, select option **4 Schedule a Deferred Monitoring Request** from the Primary Options panel as shown in Figure 9-16.

```
-------- IBM Application Monitor for z/OS - Primary Options -----------------

Option ===> _____________________________________________________________________


   1 Select an Active Job to Analyze
   2 Analyze a Specified Job
   3 Analyze a Named Set of Jobs
   4 Schedule a Deferred Monitoring Request
   5 Analyze Historical Data
   6 Set Your Defaults
   7 Administration
   X Exit


     Copyright(c) 2000-2004 - SERENA Software, Inc.  All Rights Reserved.

Enter END (PF3) to exit




                                                                      03/014
```

Figure 9-16 Primary Options

In response, Application Monitor displays the panel shown in Figure 9-17.

```
-------- IBM AM for z/OS - Schedule a Deferred Monitoring Request ------------

Option ===> _____________________________________________________________


Specify the monitoring schedule below and then select an option

  Job Name          ==> COBOLRUN   or Named Set   ==> ________

  Start YYYY/MM/DD  ==> 2004/10/25 Start hh:mm:ss ==> 15:00:00
  Stop  YYYY/MM/DD  ==> 2004/10/25 Stop  hh:mm:ss ==> 15:05:00

  Max duration      ==> 90         Max samples    ==> 10000

  History Recording ==> 'CHABERT.AM.COBOLRUN.D2004301.T1438312'

  1 Activate this schedule
  2 Specify Job Details
  X Go back without activating this schedule (PF3)

Enter END (PF3) to go back without activating the schedule



                                                                      03/014
```

Figure 9-17 Schedule a Deferred Monitoring Request

Here is the relevant information about the main fields:

Job name Name of the job you want to monitor. This field can identify a non-MVS-system started task, TSO user session, CICS or IMS region, or a user application that you submit for execution as a batch job.

Named Set Name of the named set you want to monitor.

This field is mutually exclusive with Job Name.

Start The date and time that you want Application Monitor to start watching for the job (or jobs in the named set) to become active. The start date and time you specify must be in the future or the entry is not accepted. These fields are required.

Stop The date and time that you want Application Monitor to stop watching for the job (or jobs in the named set) to become active and stop monitoring them if a monitoring session is already under way.

You can place the cursor at the beginning of the Start mm/dd/yy field, delete the start date and press Enter to update the Start/Stop dates and times with the current system date and time.

Max Duration Maximum duration of the monitoring session (in minutes).

The Max Duration default that you set with option 6.1, Specify Default Sampling Parameters does not apply to deferred requests.

Max Samples Maximum number of samples to be taken during the monitoring session.

History Recording Recording Name of the history data set to which monitoring data will be written. This field is primed with the default name that is currently in effect for the History Recording data set.

For a single job, Application Monitor starts watching for the job to become active at the Start Date and Time specified in the request. If the job becomes active at any time during the Start/Stop period, Application Monitor monitors the job.

For a named set request, Application Monitor starts watching at the Start Date and Time for each job in the set to become active. Application Monitor starts monitoring a job when it becomes active. Application Monitor does this for each job in the named set that becomes active during the Start/Stop interval.

If a single job or a job in a named set request does not become active during the Start/Stop time interval, Application Monitor does not monitor that job.

Note: If the job (or a job in a named set) becomes active within the specified Start/Stop time frame, Application Monitor monitors the job for Max Duration, until Max Samples have been taken, or until the job completes, whichever occurs first.

If the request specifies a Job Name (instead of a named set), select Option 2 to specify additional monitoring criteria. In response, Application Monitor displays the Specify Deferred Job Details panel as shown in Figure 9-18.

```
-------- IBM AM for z/OS - Specify Deferred Job Details ----------------------

Command ===> _____________________________________________________________________


Complete the job details below and then return to Deferred Monitoring (PF3)

   Job Name     ==> COBOLRUN        Select:
   System       ==> STLABF6          _ DB2 analysis  DB2 SSID ==> ____
                                     _ Delay Analysis
                                     _ Data Sets Analysis

   Job Step     ==> *_______  Proc Step ==> *_______
   CICS Tran    ==> ________
   IMS Message  ==> ________


Enter END (PF3) to go back

                                                                        03/015
```

Figure 9-18 Specify Deferred Job Details

All the fields have already been explained earlier in this chapter.

After you have specified job details and returned from the Specify Deferred Job Details panel, select **Option 1** to activate this schedule.

9.5.2 How to administer a deferred monitoring request

To perform administration functions select option **7 Administration** from the Primary Options panel to get the panel shown in Figure 9-19.

```
-------- IBM AM for z/OS - Administration -----------------------------------

Option ===> _____________________________________________________________


  1 Administer Deferred Monitoring Requests
  2 Create, Edit, and Delete Named Sets
  X Go back (PF3)



Enter END (PF3) to go back

                                                                    03/014
```

Figure 9-19 Administration

Select option **1 Administer Deferred Monitoring Requests**; the panel shown in Figure 9-20 is displayed.

```
FBIP7100 IBM AM for z/OS - Administer Deferred Requests ------ Row 1 to 3 of 3


Enter a line command to administer a monitoring request

  D - Delete (Delete the monitoring request)
  P - Stop (Stop monitoring)
  V - View (See details about the request)


C Status   System   Name        Request Enabled     Request Disabled
_ Active   N/A      CICSPLEX    2004/10/25 15:00:00 2004/10/25 15:05:00
_ Pending  STLABF6  PLIRUN      2004/10/25 15:00:00 2004/10/25 15:10:00
_ Complete STLABF6  COBOLRUN    2004/10/25 14:58:00 2004/10/25 15:00:00
****************************** Bottom of data ******************************





Command ===> ____________________________________________ Scroll ==> PAGE
 F1=Help      F2=Split     F3=End       F4=Return    F5=Nop       F6=Nop
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cretriev
                                                                    22/015
```

Figure 9-20 Administer Deferred Requests

This panel lists the deferred requests that have been scheduled and enables you to:

- Delete a deferred request.
- Stop the monitoring for a currently active deferred request.
- View information about a selected request.

Here is the relevant information about the main fields:

C Line command field. Line commands are:

D Deletes the request or removes it from this list, as follows:

If the request Status is Active, the request is removed from this list and monitoring terminates.

If the request Status is Pending, it is removed from this list and the request is deleted.

If the request Status is Complete, it is removed from this list.

Note: The D line command deletes a deferred request from this list. It does not delete a job from the system.

P Stops monitoring if the request Status is Active. Its status changes to Complete so that you can analyze the data in its historical data set.

Note: The P line command stops job monitoring. It does not cancel the job in the system.

V View the details of the selected request. For a single job, the View a Deferred Job Request panel is displayed. For a named set, the View a Deferred Named Set Request panel is displayed. This panel lists the jobs in the named set so you can select a job to view.

Note: Currently, you can process only one line item at a time with a D, P, or V line command.

Status **Pending**: Request is scheduled but not yet Active. Pending is also displayed if the target job is between job steps.

Active: Monitoring for the request is currently under way.

Complete: Monitoring for the request is complete.

Note: If you schedule a deferred request and specify the * (all steps) for the Job Step or Proc Step name, either explicitly or as the default, there might be a delay of up to 3 minutes after the request completes execution before Application Monitor recognizes that the job has ended and makes the history data set available for analysis. This is normal.

Note: If the Application Monitor goes down for some reason before the deferred request is run, the request Status is given as Complete even though the deferred request was never run.

The other fields have already been explained earlier or are self-explanatory.

The next two chapters describe the reports produced by Application Monitor.

10

Application Monitor: Analyzing a job

This chapter describes reports provided by Application Monitor. These reports are the same for both real-time and history data set analysis, but extra options are available if you are using history data set. The first part of the chapter explains the special functions available with a history data set analysis.

The remainder of the chapter describes the following AM reports:

- Program module reports

  To illustrate Program module reports, we used a batch application. The main program is a COBOL program called COBPLIAM. It calls three CSECTs, of which two are COBOL and one is PL/I. Neither CICS, nor DB2, nor IMS is used for that batch program.
- Delay analysis reports
- Data set analysis reports

  To illustrate both Delay analysis reports and Data set analysis reports we used a CICS sample transaction called AMNU. The purpose is only to describe Delay analysis Application Monitor IVP calls.

10.1 Special features of history data sets

When you analyze the contents of the history data set online, you can specify the time range that you want to analyze. Therefore, you can avoid getting average values calculated for the full monitoring time frame, and you can analyze the peak time frame, where most of your concerns can come from. With these values your analysis will be more accurate, so your correcting actions will be also.

Select option **5 Analyze Historical Data** from the Application Monitor Primary Options panel shown in Figure 10-1 to view the contents of a history data set.

```
-------- IBM Application Monitor for z/OS - Primary Options ------------------

Option ===> _____________________________________________________________


   1 Select an Active Job to Analyze
   2 Analyze a Specified Job
   3 Analyze a Named Set of Jobs
   4 Schedule a Deferred Monitoring Request
   5 Analyze Historical Data
   6 Set Your Defaults
   7 Administration
   X Exit


     Copyright(c) 2000-2004 - SERENA Software, Inc.  All Rights Reserved.

Enter END (PF3) to exit




                                                                    03/014
```

Figure 10-1 Primary Options

10.1.1 Analyze Historical Data

Figure 10-2 shows the Analyze Historical Data panel displayed when you select option 5.

```
-------- IBM AM for z/OS - Analyze Historical Data ---------------------------


Specify a historical data set below and then press ENTER


   Data Set  ==> CHABERT.AM.*_______________________________
   Enter a fully qualified data set name without quotes or a generic
   data set name ending in "*" to select from a list.


Press ENTER to analyze data or to select from a list
Enter END (PF3) to go back without analysis









Command ===> ____________________________________________________________________
                                                                        07/031
```

Figure 10-2 Analyze Historical Data

Select the history data set to analyze in one of the following ways:

- Type in a fully qualified data set name, without quotes. If the data set you specify exists, Application Monitor displays the Specify the Time Intervals To Analyze panel where you can set the start and end time frame for the analysis data you want to see and the duration of the sampling intervals.
- Type in a generic data set name ending with an asterisk, for example, CHABERT.AM.*, and press Enter. In response, Application Monitor displays the Select a Historical Data Set panel with a list of data set names that match the generic mask.

Note: If you enter only the asterisk, you will get a list of all data sets under your high-level qualifier, not just the historical data sets.

10.1.2 Select a History Data Set

Figure 10-3 shows the **Select a History Data Set** panel. Application Monitor displays this panel if you enter a generic name for the historical data set on the Analyze Historical Data panel. The panel lists all historical data sets that match the generic name so you can choose the one you want to analyze.

```
-------- IBM AM for z/OS - Select a History Data Set --------- Row 1 to 3 of 3


Select a History Data Set to analyze

C Data Set Name
_ CHABERT.AM.CICSPLEX.D2004299.T1459566
_ CHABERT.AM.COBOLRUN.D2004299.T1456425
_ CHABERT.AM.COBOLRUN.D2004300.T1005465
****************************** Bottom of data ******************************









Command ===> _____________________________________________ Scroll ==> CSR
                                                                    24/015
```

Figure 10-3 Select a History Data Set

Tab to the name of the data set that you want to analyze and type any character in the line command (C) field to the left of the data set name; press Enter to select it. In response, Application Monitor re-displays the Analyze Historical Data panel with the name of the data set you selected in the Data Set Name field.

10.1.3 Specify the Time Intervals to Analyze

Figure 10-4 shows the **Specify the Time Intervals To Analyze** panel. This panel is displayed after you have selected the historical data set that you want to analyze.

```
-------- IBM AM for z/OS - Specify the Time Intervals to Analyze -------------
                                                        Row 1 of 1
History Data Set: CHABERT.AM.COBOLRUN.D2004299.T1456425
Time spanned:     2004/10/25 14:58:37 to 2004/10/25 14:59:05
  Named Set:  N/A

Specify the time range and analysis interval and then select an option

   Start Date ==> 2004/10/25 Time ==> 14:58:37
   End Date   ==> 2004/10/25 Time ==> 14:59:05
   Interval   ==> 10____ Time Units: / Minutes _ Seconds

   1 Analyze the intervals in the specified time range
   X Go back without analyzing (PF3)

Begin    End      System   Job      Step     Pgm  CICS IMS  DB2  Dly  Dsn
-------- -------- -------- -------- -------- Rpts Rpts Rpts Rpts Rpts Rpts
14:58:37 14:59:05 STLABF6  COBOLRUN STEP01   YES
****************************** BOTTOM OF DATA ******************************



Enter END (PF3) to go back without analysis
Option ===> _____________________________________________ Scroll ==> CSR
                                                                    24/014
```

Figure 10-4 Specify the Time Intervals to Analyze

Here is the relevant information about the main fields:

Time Spanned The from-to period of time spanned by the analysis data in the selected data set in the format yyyy/mm/dd hh:mm:ss. This field shows when Application Monitor started and stopped monitoring the entire job.

The start time and stop time fields are point-and-shoot fields. That is, place the cursor on the start time or end time of time spanned to reset the Start Time or End Time displayed in the middle of the panel. This allows you to have details about peak time activity.

Interval Enter the duration of the monitoring interval over which you want to view the analysis data.

Time Units Select the unit (minutes or seconds) that applies to the value you entered for Interval by typing the / character in the Minutes or Seconds entry field.

For example, if you enter 10 for Interval and select Seconds for Time Units, the data that is extracted for display will be taken from the samples that Application Monitor collected over successive ten-second intervals beginning with the Start Date/Time and ending with the End Date/Time.

Note: Display content reflects the data collected from only the samples taken during that interval; the data is not cumulative from the start of the first interval.

The other fields have already been explained previously, or are self-explanatory.

How to use this panel

Fill in the desired values and select option **1 Analyze the intervals in the specified time range**. Application Monitor displays the **Historical Reports Available** display.

The history data set analysis has three more lines than the real-time analysis. These lines are:

- History Data Set
- Interval Start
- Interval Stop

Figure 10-5 and Figure 10-6 highlight this difference.

```
FBIP1H10 IBM AM for z/OS - Historical Reports Available -----------------------
         2004/10/25 14:58:37.0 to 2004/10/25 18:57:31.8   Row 1 of 1
History Data Set: CHABERT.AM.COBOLRUN.D2004299.T1456425
  Interval Start: 2004/10/25  14:58:37
   Interval Stop: 2004/10/25  14:58:47

System   Jobname  Stepname Pgm  CICS IMS  DB2  Dly  Dsn  CPU%   Sample
```

Figure 10-5 Historical Reports Available

```
FBIP1A00 IBM AM for z/OS - Active Monitor Sessions ----------------------------
         2004/10/25 18:55:21.6 to 2004/10/25 18:55:28.7   Row 1 of 1

  Named Set: N/A
System   Jobname  Stepname Pgm  CICS IMS  DB2  Dly  Dsn  CPU%   Sample Duration
```

Figure 10-6 Active Monitor Sessions

10.2 Program module reports

We use a real-time monitoring of a multi-csects, multi language batch application to illustrate these reports as shown in Figure 10-7.

```
FBIP1A00 IBM AM for z/OS - Active Monitor Sessions ---------------------------
         2004/10/26 10:25:13 to 2004/10/26 10:25:55        Row 1 of 1

  Named Set: N/A
System   Jobname  Stepname Pgm  CICS IMS  DB2  Dly  Dsn  CPU%   Sample Duration
-------- -------- -------- Rpts Rpts Rpts Rpts Rpts Rpts ------  Count --------
STLABF6  COBOLRUN STEP01     10                            23.66  9,462 00:00:28
****************************** BOTTOM OF DATA ******************************



Command ===> _____________________________________________ Scroll ==> CSR
 F1=Help      F2=Split     F3=End       F4=Return    F5=Nop       F6=Nop
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cretriev
                                                                   22/015
```

Figure 10-7 Active Monitor Sessions

Figure 10-7 shows that our Application Monitoring defaults have been set up to only record Program Reports (Pgm Rpts). Therefore only the following displays about program modules are described in detail:

- CPU and Loadmod Analysis
- CSECT Utilization Analysis
- Statement Analysis (for COBOL and PL/I programs)
- Detail Statement Display
- Instruction Analysis
- Detail Instruction Analysis

By cursor-selecting an item in the Pgm Rpts column on the Active Monitor Session display, we will get these reports.

10.2.1 CPU and Loadmod Analysis

Figure 10-8 depicts the CPU and Loadmod Analysis display. This display identifies all of the active load modules in the target job step. An active load module is one that was executing when Application Monitor took a sample during the monitoring interval displayed on the line directly beneath the display title.

```
FBIP1D00 IBM AM for z/OS - CPU and Loadmod Analysis --------------------------
         2004/10/26 10:25:13 to 2004/10/26 10:25:41       Row 1 of 10

    Jobname: COBOLRUN        Stepname: STEP01          Duration: 00:00:28
    Samples: 9,462              %Supv: 0.29               %Prob: 99.70
    CPU Pct: 23.66           %InReady: 105.59           %InWait: 0.09
       %Out: 0.00             %OutRdy: 0.00             %OutLog: 0.00
Module   Area Address  <  10   20   30   40   50   60   70   80   90   > Pct.
-------- ---- -------- ------------------------------------------------- ------
IGZCPAC  EPVT 0D8DD080 ====================================>             78.93
COBOLSP2 EPVT 0D97B178 ====>                                             11.29
FBIEPLI  EPVT 0D97F620 ==>                                                7.47
COBOLSP1 EPVT 0D9776E0 .                                                  1.10
COBPLIAM EPVT 0D7001E0 .                                                  0.57
IGZEINI  PVT  00017008 .                                                  0.31
CSAHIGH  ECSA 074C7000 .                                                  0.11
IECVPST  NUC  01297C78 .                                                  0.09
CEEPLPKA EPVT 0D710350 .                                                  0.04
IGVVSTOR NUC  01436A60 .                                                  0.04
****************************** BOTTOM OF DATA ******************************

Command ===> ____________________________________________ Scroll ==> CSR
 F1=Help      F2=Split     F3=End       F4=Return    F5=Nop       F6=Nop
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cretriev
                                                                   22/015
```

Figure 10-8 CPU and Loadmod Analysis

The rows are sorted in descending sequence by Pct. You can sort the rows in ascending or descending sequence by any column by placing the cursor on the column heading and pressing Enter.

Here is the relevant information about the main fields:

%Supv The percentage of the total number of samples taken where the program is determined to be in supervisor state.

%Prob The percentage of the total number of samples taken where the program is determined to be in problem state.

Note: %Supv + % Prob = approximately 100% due to rounding.

CPU Pct The percentage of available CPU that the program has used. This value is an average of the CPU Pct values from the samples that have been collected.

%In Ready The percentage of the total number of samples taken where the program is ready to execute but cannot access the CPU because other jobs are executing. In multiprogramming environments, a certain amount of this time may appear.

%In Wait The percentage of the total number of samples taken where the program has been swapped in, waiting to execute. A value in this field does not necessarily indicate a problem. The program may simply be waiting for some work, for example, a planned wait.

You can determine why the program associated with the target Stepname is waiting from the information that Application Monitor displays on the Delay Analysis panel.

%Out The percentage of the total number of samples taken where the program was logically or physically swapped out.

%Out Rdy The percentage of the total number of samples taken where the swapped-out program was actually ready to execute.

%Out Log The percentage of the total number of samples taken where the program was logically or physically swapped out.

Note: This value can be less than or equal to %Out. If %Out Log is less than %Out, paging and swapping operations account for the difference.

Module This column lists the load modules that have been active for the target Stepname during the displayed monitoring interval.

Values can be as follows:
- module name
- Don't Know (the module name cannot be determined)
- AR-MODE (Code executed from AR-specified virtual)
- XM-MODE (Code executed from secondary virtual)

The following module name values describe an area of code that does not have an associated program name. They represent areas of executable storage that do not have entries in the Job Pack Queue (JPQ), Contents Directory Entry (CDE), or Link Pack Directory Entry (LPDE) chains. This can occur when an application gets some storage and moves executable code into one of these areas:

?CSAHI CSA above the 16 MB line.
?CSALO CSA below the 16 MB line.
?SQAHI SQA above the 16 MB line
?SQALO SQA below the 16 MB line.

Area The area of virtual storage where the module is executing.

Address The starting address of the module in storage in hexadecimal format.

(Pct indicator) This ===> indicator gives a graphical representation of the Pct value so you can easily spot the modules that are using the most CPU.

Pct The percentage of CPU that each load module has used. The values in this field should total approximately 100 percent due to rounding.

Note: If the CPU Pct for job step STEP01 is 23.66% and module COBOLSP2 is using 11.29% of that percentage, then COBOLSP2 is using 0.1129 x 0.2366 = 0.02671 or approximately 2.67% of total CPU time for the target job step.

Because COBOLSP2 is using a high percentage of the CPU, we will analyze the CSECT utilization by placing the cursor in the COBOLSP2 name and pressing Enter to drill down to the **CSECT Utilization Analysis** displayed in Figure 10-9.

```
FBIP1F00 IBM AM for z/OS - CSECT Utilization Analysis ------------------------
         2004/10/26 10:25:13 to 2004/10/26 10:25:41       Row 1 of 1

    Jobname: COBOLRUN        Stepname: STEP01          Duration: 00:00:28
    Samples: 9,462             Module: COBOLSP2
     Csects: 12
   From Dsn: CHABERT.FBI.TEST.LOAD
       Desc:

CSECT     Language  Start     End       < 10 20 30 40 50 60 70 80 90 >  Pct.
--------  --------  --------  --------  ------------------------------  ------
COBOLSP2  COBOL*    0D97B178  0D97E3B1  ============================>  100.00
****************************** BOTTOM OF DATA ******************************




Command ===> ____________________________________________________ Scroll ==> CSR
 F1=Help      F2=Split     F3=End       F4=Return    F5=Nop       F6=Nop
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cretriev
                                                                   22/015
```

Figure 10-9 CSECT Utilization Analysis

Here is the relevant information about the main fields:

Csects The total number of CSECTS in the target load module.

Note: This number may be larger than the number of CSECTS listed in the CSECT column. Only the CSECTs that were active during the displayed monitoring interval are listed.

From Dsn The data set where the target load module resides.

Desc Prefix description that identifies the type of load module.

Language The source language of the target load module. An asterisk following the language name indicates that the compiler listing for the program is automatically available for referencing on the Statement Analysis display.

CSECT Name of a CSECT in the target load module. An asterisk precedes the CSECT name if the compiler (rather than the programmer) created the CSECT name.

Some compilers, PL/I for example, generate internal CSECT names that the programmer has not created.

Start The starting address of the CSECT (hexadecimal).

End The ending address of the CSECT (hexadecimal).

Because an asterisk follows the name of the source name language, it means that **Statement Analysis** is available. Place the cursor on this field COBOL* and press Enter to bring it up.

Note: If the language name is not followed by an asterisk, you can place the cursor on this filed and press Enter to bring up Statement Analysis Member selection. This display allows you to manually select a listing member from the default listing data set.

10.2.2 Statement Analysis

Figure 10-10 shows the **Statement Analysis** display. It shows the source statement that is contained in the target program module and the percentage of total CPU that the statement is using with respect to all other statements in the module.

```
FBIP1F00 IBM AM for z/OS - Statement Analysis ---------------------------------
         2004/10/26 10:25:13 to 2004/10/26 10:25:41        Row 1 of 10

    Jobname: COBOLRUN        Stepname: STEP01          Duration: 00:00:28
    Samples: 9,462             Module: COBOLSP2         CPU Pct: 23.66



Stmtno Statement                                        <   CPU%   >  Pct.
------ ------------------------------------------------ ------------ ------
    41 COMPUTE LOAN-AMOUNT = FUNCTION NUMVAL(LOAN-AMOUNT .              0.56
    42 COMPUTE INTEREST = FUNCTION NUMVAL(INTEREST-IN).  .              1.12
    43 COMPUTE NO-OF-PERIODS = FUNCTION NUMVAL(NO-OF-PER .              1.68
    45 PERFORM GET-AMOUNTS VARYING COUNTER FROM 1 BY 1 U .              2.38
    48 COMPUTE PAYMENT = FUNCTION PRESENT-VALUE(INTEREST ========>     79.49
    51 MOVE PAYMENT TO PAYMENT-OUT.                      .              3.93
    52 STRING "COBOLSP2:_PRESENT_VALUE_FOR_RATE_OF_" INT .              1.12
    63 MOVE "OK" TO CALL-FEEDBACK.                       .              3.37
    68 MOVE BUFFER-ARRAY (COUNTER) TO INPUT-1.           .              1.26
    69 COMPUTE VALUE-AMOUNT (COUNTER) = FUNCTION NUMVAL( .              5.05
****************************** BOTTOM OF DATA ********************************

Command ===> _____________________________________________ Scroll ==> CSR_
                                                                     24/015
```

Figure 10-10 Statement Analysis

Here is the relevant information about the main fields:

Stmtno The number assigned to the statement in the compiler listing. Each statement is displayed as a separate line item.

Statement The text of the statements contained in the target module. Only the first 49 characters of the statement are shown.

If the statement is longer than 49 characters, cursor-select the statement number in the Stmtno column and press Enter to drill down to the Detail Statement Display.

You can sort the rows in descending sequence by Pct. to spot the statement that is taking the most CPU as shown in Figure 10-11.

```
FBIP1F00 IBM AM for z/OS - Statement Analysis ---------------------------------
         2004/10/26 10:25:13 to 2004/10/26 10:25:41       Row 1 of 10

    Jobname: COBOLRUN        Stepname: STEP01          Duration: 00:00:28
    Samples: 9,462             Module: COBOLSP2         CPU Pct: 23.66



Stmtno Statement                                        <   CPU%   >  Pct.
------ ------------------------------------------------ ------------ ------
    48 COMPUTE PAYMENT = FUNCTION PRESENT-VALUE(INTEREST ========>     79.49
    69 COMPUTE VALUE-AMOUNT (COUNTER) = FUNCTION NUMVAL( .              5.05
    51 MOVE PAYMENT TO PAYMENT-OUT.                      .              3.93
    63 MOVE "OK" TO CALL-FEEDBACK.                       .              3.37
    45 PERFORM GET-AMOUNTS VARYING COUNTER FROM 1 BY 1 U .              2.38
    43 COMPUTE NO-OF-PERIODS = FUNCTION NUMVAL(NO-OF-PER .              1.68
    68 MOVE BUFFER-ARRAY (COUNTER) TO INPUT-1.           .              1.26
    42 COMPUTE INTEREST = FUNCTION NUMVAL(INTEREST-IN).  .              1.12
    52 STRING "COBOLSP2:_PRESENT_VALUE_FOR_RATE_OF_" INT .              1.12
    41 COMPUTE LOAN-AMOUNT = FUNCTION NUMVAL(LOAN-AMOUNT .              0.56
****************************** BOTTOM OF DATA ******************************

Command ===> _____________________________________________ Scroll ==> CSR
                                                                     24/015
```

Figure 10-11 Statement Analysis sorted by Pct

You can easily see that statement number **48** uses the most CPU (more than 79% of the total). But the text of the statement exceeds the length of the Statement column, so to view the full code, cursor-select the number (48) of the statement and press Enter to bring up the **Detail Statement Display**.

10.2.3 Detail Statement Display

Figure 10-12 shows the Detail Statement Display. It contains the entire text of the target statement.

```
FBIP1D00 IBM AM for z/OS - Detail Statement Display --------------------------
         2004/10/26 10:25:13 to 2004/10/26 10:25:41       Row 1 of 2

    Jobname: COBOLRUN        Stepname: STEP01          Duration: 00:00:28
    Samples: 9,462             Module: COBOLSP2         CPU Pct: 23.66


Stmtno Statement                                        <   CPU%   >  Pct.
------ ------------------------------------------------ ------------ ------
    48 COMPUTE PAYMENT = FUNCTION PRESENT-VALUE(INTEREST ========>     79.49
       VALUE-AMOUNT(ALL) ).
****************************** BOTTOM OF DATA ******************************









Command ===> _____________________________________________ Scroll ==> CSR
                                                                     24/015
```

Figure 10-12 Detail Statement Display

All the fields have already been explained or are self-explanatory.

Because you now have the entire text of the target statement (COMPUTE in this example), you can try to find another way to reach the same goal, but using less CPU.

> **Note:** The Statement Analysis display is available only for COBOL and PL/I programs and if:
>
> 1. A SYSPRINT data set for the compiler listing has been created.
> 2. The name of this listing data set has been specify with option **6.4 Specify COBOL and PL/1 Listing Data Sets.**

If for any reason the Statement Analysis display is not available, the **Instruction Analysis** display provides some useful information too, even if it is less user-friendly than the Statement Analysis display.

10.2.4 Instruction Analysis

Instruction Analysis is shown if you cursor-select a CSECT on the CSECT Utilization Analysis display. In response, Application Monitor displays the panel shown in Figure 10-13.

```
FBIP1D00 IBM AM for z/OS - Instruction Analysis ------------------------------
         2004/10/26 10:25:13 to 2004/10/26 10:25:41       Row 1 of 63

    Jobname: COBOLRUN        Stepname: STEP01          Duration: 00:00:28
    Samples: 9,462             Module: COBOLSP2           CSECT: COBOLSP2
    CPU Pct: 23.66         Module Addr: 0D97B178      CSECT Addr: 0D97B178

Offset   Sample Data     Instruction                        <  CPU%  > Pct.
-------- --------------- ---------------------------------- ---------- ------
00000000 47F0 F028       BC    15,X'28'(,R15)               .            0.18
000005FA D203 D148 D138  MVC   X'148'(4,R13),X'138'(R13)    .            7.01
00000600 4120 B096       LA    R2,X'96'(,R11)               .            0.37
00000604 5020 D138       ST    R2,X'138'(,R13)              .            0.56
00000608 5820 D4F4       L     R2,X'4F4'(,R13)              .            1.49
0000060C 07F2            BCR   15,R2                        .            3.36
0000060E D203 D138 D148  MVC   X'138'(4,R13),X'148'(R13)    .            7.48
00000614 5820 D4E8       L     R2,X'4E8'(,R13)              .            3.08
00000618 5A20 D514       A     R2,X'514'(,R13)              .            1.49
0000061C 5020 D4E8       ST    R2,X'4E8'(,R13)              .            0.74
00000620 5820 D530       L     R2,X'530'(,R13)              .            1.12
00000624 0620            BCTR  R2,R0                        .            1.12
00000626 5020 D530       ST    R2,X'530'(,R13)              .            1.77
0000062A 1222            LTR   R2,R2                        .            0.37
Command ===> ____________________________________________________ Scroll ==> CSR
                                                                      24/015
```

Figure 10-13 Instruction Analysis

The Instruction Analysis display shows the percentages of CPU time per instruction group at the designated offset location in the target CSECT. Each *instruction group* consists of the contents of the first 16 bytes from the beginning of the Offset location.

A separate line item appears on this display for each sample that is taken during the displayed monitoring interval. Each line item shows only the first 2-, 4-, or 6-byte instruction of the instruction group, which is taken from the PSW.

Here is the relevant information about the main fields:

Offset The hexadecimal offset of the instruction group in the target CSECT.

Sample Data The hexadecimal representation of the first 2-, 4-, or 6-byte instruction of the instruction group beginning at the Offset location.

Instructions The mnemonic for the first instruction and an Assembler-like representation of the registers, operands, and data used in the instruction.

To view the entire 16-byte instruction group, cursor-select a value in the Offset column and press Enter to drill down to the **Detail Instruction Display**.

10.2.5 Detail Instruction Display

Figure 10-14 shows the **Detail Instruction Display**. It shows the contents of the first 16 bytes of the instruction group beginning at the target Offset location within the CSECT that is being analyzed. The display has a line item for each 2-, 4-, or 6-byte instruction in the group.

```
FBIP1D00 IBM AM for z/OS - Detail Instruction Display ------------------------
         2004/10/26 10:25:13 to 2004/10/26 10:25:41       Row 1 of 5

    Jobname: COBOLRUN        Stepname: STEP01          Duration: 00:00:28
    Samples: 9,462             Module: COBOLSP2           CSECT: COBOLSP2
    CPU Pct: 23.66        Module Addr: 0D97B178      CSECT Addr: 0D97B178

Offset   Sample Data     Instructions                      <  CPU%  > Pct.
-------- --------------- -------------------------------- ---------- ------
000005FA D203 D148 D138  MVC   X'148'(4,R13),X'138'(R13)   .            9.44
         4120 B096       LA    R2,X'96'(,R11)
         5020 D138       ST    R2,X'138'(,R13)
         5820            L     ...

******************************** BOTTOM OF DATA ********************************




Command ===> _____________________________________________ Scroll ==> CSR
                                                                      24/015
```

Figure 10-14 Detail Instruction Display

Here is the relevant information about the main fields:

Offset The hexadecimal offset of the beginning of the target instruction group in the CSECT that is being analyzed.

Sample Data The hexadecimal representation of the 2-, 4-, or 6-byte instruction.

Instructions The mnemonic for each instruction and an Assembler-like representation of the registers, operands, and data used in the instruction. The **...** appears in an entry to indicate that the length of the data exceeds the 16-byte display limit.

10.2.6 Statement Analysis Member selection

When you cursor-select a **Language** not followed by an asterisk from **CSECT Utilization Analysis,** AM returns the panel shown in Figure 10-15.

```
FBIP1F00 IBM AM for z/OS - CSECT Utilization Analysis ------------------------
         2004/10/26 10:25:13 to 2004/10/26 10:25:41       Row 1 of 1

    Jobname: COBOLRUN        Stepname: STEP01          Duration: 00:00:28
    Samples: 9,462             Module: FBIEPLI
     Csects: 20
   From Dsn: CHABERT.FBI.TEST.LOAD
       Desc: APPLICATION MONITOR

CSECT     Language  Start     End       < 10 20 30 40 50 60 70 80 90 >  Pct.
--------  --------  --------  --------  ------------------------------  ------
FBIEPLI1  PL/1      0D97F6A0  0D97FF47  ============================>  100.00
****************************** BOTTOM OF DATA ******************************

Command ===> _____________________________________________ Scroll ==> CSR
                                                                  24/015
```

Figure 10-15 CSECT Utilization Analysis

Then Application Monitor displays Figure 10-16.

```
FBIP1L00 IBM AM for z/OS - Statement Analysis Member selection ----------------
Member List  CHABERT.FBI.TEST.PLILIST                     Row 00001 of 00002
   Name     Prompt          Size    Created          Changed            ID
. FBIEPLI
. FBIZS01
  **End**

Command ===> _____________________________________________ Scroll ===> PAGE
                                                                  24/015
```

Figure 10-16 Statement Analysis Member selection

From here you can select a listing member if it has been created, and Application Monitor displays the statement analysis (if it is the right member) shown in Figure 10-17.

```
FBIP1F00 IBM AM for z/OS - Statement Analysis ---------------------------------
         2004/10/26 10:25:13 to 2004/10/26 10:25:41       Row 1 of 45

    Jobname: COBOLRUN        Stepname: STEP01          Duration: 00:00:28
    Samples: 9,462             Module: FBIEPLI          CPU Pct: 23.66



Stmtno Statement                                        <   CPU%   >  Pct.
------ ------------------------------------------------ ------------ ------
    36 2             CHAR2 = 'for applications and sys p .              1.13
    37 2             CHAR3 = 'A non-intrusive monitor    .              6.08
    38 2             CHAR4 = 'provides resource utilizat .              3.53
    39 2             CHAR5 = 'information for your appli .              1.98
    40 2             CHAR6 = 'APM provides info on your  .              3.39
    41 2             CHAR7 = 'The output can be the curr .              2.26
    42 2             CHAR8 = 'system data                .              1.69
    43 2             CHAR9 = '(Real time analysis)       .              1.13
    44 2             CHAR11= 'the collected data appears .              1.69
    45 2             CHAR12= 'real time or printed forma .              3.39
    46 2             CHAR13= 'or it might be the data    .              0.56
    47 2             CHAR14= 'collected over a period of .              1.69
Command ===> ____________________________________________________ Scroll ==> CSR
                                                                       24/015
```

Figure 10-17 Statement Analysis

10.3 Delay Analysis Report

Both Delay and Data set reports are optional. To analyze these reports, the proper options must have been activated, either using Application Monitor option **6.1 Specify Default Sampling Parameters,** option **2 Analyze a Specified Job,** or option **4.2 Specify Deferred Job Details,** or when you **Specify a Named Set Entry** for a named set. Figure 10-18 is an example using option **6.1 Specify Default Sampling Parameters.**

```
FBIP6100 IBM AM for z/OS - Specify Default Sampling Parameters ---------------

Option ===> ____________________________________________________________________


Update the defaults below and then press PF3 or select an option

   Max Duration ==> 90      (1-999 minutes)
   Max Samples  ==> 10000   (100-99999999)
   Analyze DB2 activity for every job   _   Default DB2 SSID==> ____
   Analyze Delays for every job         /
   Analyze Data Sets for every job      /

   1 Reinstate your profile defaults
   2 Clear your defaults to built-in default values


Enter END (PF3) to save your updates and go back




                                                                       03/014
```

Figure 10-18 Specify Default Sampling Parameters

Both Analyze Delays for every job and Analyze Data Sets for every job have to be selected.

Select an item in the Dly Rpts column on the Active Monitor Sessions panel as shown in Figure 10-19.

```
FBIP1A00 IBM AM for z/OS - Active Monitor Sessions ---------------------------
         2004/10/27 15:09:15 to 2004/10/27 15:09:29       Row 1 of 1

  Named Set: N/A
System   Jobname  Stepname Pgm  CICS IMS  DB2  Dly  Dsn  CPU%   Sample Duration
-------- -------- -------- Rpts Rpts Rpts Rpts Rpts Rpts ------  Count --------
STLABF6  CICSC31F CICSC31F    3 YES            YES  YES    0.89     12 00:00:13
******************************** BOTTOM OF DATA ********************************

Command ===> _____________________________________________ Scroll ==> CSR_
                                                                     24/015
```

Figure 10-19 Active Monitor Sessions

This will bring up the **Delay Analysis Report** display as shown in Figure 10-20. This display identifies the reason the programs associated with the target job are not executing (are in a wait state). The display has a line item for each wait reason.

```
-------- IBM AM for z/OS - Delay Analysis Report ------------------------------
         2004/10/28 09:41:37 to 2004/10/28 09:47:33       Row 1 of 6

     System: STLABF6          Jobname: CICSC22F        Stepname: CICSC22F
   Programs: 3                CPU Pct: 14.03           Duration: 00:05:55
Dly Samples: 1,770            Samples: 10,000
Module     Offset SVC          Data                  <---- Delay Impact ------>
-------- -------- ---          ----                  -----
DFSPAT00 00000ADC 01  WAIT     ECBWaSyc               8.33 ==>
DFSPRRA0 00002476 01  WAIT     ECBLaSyc               8.33 ==>
DFHKETCB 00010F8E 01  WAIT     ECBLaSyc               7.58 ==>
DFHKETCB 00011152 01  WAIT     ECBWaSyc              49.99 ==================>
DFHKETCB 0004F5BC 01  WAIT     ECBWaSyc               8.33 ==>
DFHKETCB 00131AB6 01  WAIT     ECBWaSyc               8.33 ==>
******************************** BOTTOM OF DATA ********************************

Command ===> _____________________________________________ Scroll ==> CSR_
                                                                     24/015
```

Figure 10-20 Delay Analysis Report

Here is the relevant information about the main fields:

Dly Samples The number of samples that Application Monitor has collected for the target job step that indicated a program was in a wait state.

Dly Samples are taken according to a timer algorithm, and are gathered independently of the sample count shown in the Samples field. Thus, Dly Samples can be greater or less than Samples.

Samples The number of samples (DIE samples) that Application Monitor has collected for the target job step during the monitoring interval.

Module The name of the program module where a delay is detected.

Offset The offset within the program module where the delay is detected.

SVC The number, in hexadecimal format, of the Supervisor Call (SVC) instruction that is issued to perform the service.

The hexadecimal code is followed with more information about the SVC or service being executed. For an SVC, the name of the SVC is displayed, for example, WAIT, ECBWAIT, SUSPEND, ECBLIST, STIMER, or IO ACTV.

Data This field provides two columns of detailed information about the wait.

See *IBM Application Monitor for z/OS: User's Guide* for a discussion of the information in these columns.

Delay Impact This field shows a percentage based on the total number of Dly Samples for the associated offset with respect to the number of Dly Samples in a wait state.

10.4 Data Set analysis

The following reports are available for data set analysis:

- Data Set Analysis Report - for all access methods
- VSAM Data Set Analysis Report - for VSAM data sets only
- VSAM Local Shared Resource Pools Reports - for VSAM LSR data sets only

Select an item in the Dsn Rpts column on the Active Monitor Sessions panel as shown in Figure 10-21.

```
FBIP1A00 IBM AM for z/OS - Active Monitor Sessions ---------------------------
         2004/10/26 16:57:16 to 2004/10/26 16:58:17       Row 1 of 1

  Named Set: N/A
System   Jobname  Stepname Pgm  CICS IMS  DB2  Dly  Dsn  CPU%   Sample Duration
-------- -------- -------- Rpts Rpts Rpts Rpts Rpts Rpts ------  Count --------
STLABF6  CICSC31F CICSC31F   17 YES            YES  YES    0.14    427 00:01:00
******************************** BOTTOM OF DATA ********************************

Command ===> _____________________________________________ Scroll ==> CSR
                                                                     24/015
```

Figure 10-21 Active Monitor Sessions

This will bring up a **Data Set Analysis Report** as shown in Figure 10-22.

10.4.1 Data Set Analysis Report

This report provides information about the data sets that the target job step accesses.

```
FBIP1C00 IBM AM for z/OS - Data Set Analysis Report --------------------------
         2004/10/26 16:57:16 to 2004/10/26 17:21:59       Row 36 of 51        >

     System: STLABF6          Jobname: CICSC31F        Stepname: CICSC31F
   Duration: 00:24:42

DD        Acc  Rec    Rec CI/Blk       EXCP Data Set
Name      Meth Form Length   Size      Count Name
-------- ---- ---- ------ ------ ---------- ----------------------------------
CAFF     BSAM V       133    137          0 CICPROD.CICSC31F.STC02347.D0000112.
CEEMSG   BSAM V       161    165          0 CICPROD.CICSC31F.STC02347.D0000113.
CEEOUT   BSAM V       133    137          0 CICPROD.CICSC31F.STC02347.D0000114.
CRPO     BSAM V       133    137          0 CICPROD.CICSC31F.STC02347.D0000115.
INTR     BSAM F        80                  0 CICPROD.CICSC31F.STC02347.D0000116.
CCTLDD   BPAM U           32,760         22 IMSVS.IM8F.SDFSRESL
FILEA    VSAM              3,072          0 CHABERT.FBI.TS31.FILE
DFHDPFMP VSAM              3,072          0 CICSVS.TS310.CICS.DFHDPFMP
DFHDBFK  VSAM              2,048          0 CICSVS.C31F.DFHDBFK
DFHLRQ   VSAM              2,048          0 CICSVS.C31F.DFHLRQ
DFHCSD   VSAM             18,432          0 CICSVS.C31F.DFHCSD
DFHGCD   VSAM              8,192          6 CICSVS.C31F.DFHGCD
DFHLCD   VSAM              2,048          0 CICSVS.C31F.DFHLCD
DFHINTRA VSAM              4,096          0 CICSVS.C31F.DFHINTRA
Command ===> _____________________________________________ Scroll ==> CSR
                                                                     24/015
```

Figure 10-22 Data Set Analysis Report

The **>** character on the second line indicates that you can press PF11 to scroll left to view the remaining characters in the Data Set Name column on the right.

If the < character is visible, it indicates that you can press PF10 to scroll right to view more data on the left side of the panel.

Here is the relevant information about the main fields:

DD Name The DDNAME of the data set being accessed.

Acc Meth The access method that is used to process the data set. Access methods that Application Monitor supports are: BDAM, BPAM, BSAM, ISAM and VSAM.

More detailed drilldown information is available only for VSAM.

Rec Form The record format. Possible values are:

A = ASCII
B = Blocked
F = Fixed
ICF = ICF catalog
T = Track
U = Undefined
V = Variable

Rec Length The logical record length for non-VSAM data sets only. For VSAM data sets, this information appears on the VSAM Data Set Analysis Details Report.

CI/Blk Size The control interval size for VSAM data sets or the block size for non-VSAM data sets.

EXCP Count The number of I/O operations performed on the data set during the displayed monitoring interval.

Note: A high value here can be a concern for our response time (specially with CICS, where it could be a good indicator to create a CICS Table).

You can cursor-select a VSAM value in the Acc Meth field and press Enter to drill down to the **VSAM Data Set Analysis Detail Report** for the target data set.

10.4.2 VSAM Data Set Analysis Detail Report

Figure 10-23 shows the report containing information about the target VSAM data set.

```
FBIPI080 IBM AM for z/OS - VSAM Data Set Analysis Detail Report --------------
         2004/10/26 16:57:16 to 2004/10/26 17:39:52

Command ===> _____________________________________________________________________

     System: STLABF6          Jobname: CICSC31F        Stepname: CICSC31F
   Duration: 00:42:35
    DD Name: FILEA            DS Name: CHABERT.FBI.TS31.FILE
                            Data Source: JOB CTL BLOCKS
Status   Type Format  Access   Opts            LSR  Misc
OPEN-OUT KSDS         LSR      DIR   ADRKEY    001  31B
                                                     DSNSHR UPAD
DATA:-----------------------------------------------------------------------
 CIFR  CAFR   CISP   CASP   CISZ   CI-CA   Keyln  Keyps  Maxlr
10     10     0      0      3,072  225     17     1      2,560

EXTNT EXCPs      Reccount     Retrved     Updated     Deleted     Insrted
1     0          2            30          0           0           0

INDEX:----------------------------------------------------------------------
 CIFR   CAFR  CISP       CASP  CISZ    CA     LVL    EXT Reccnt    EXCP
0       0     0          0     2,560   17     1      1   1         0


                                                                    04/015
```

Figure 10-23 VSAM Data Set analysis Detail Report

Here is the relevant information about the main fields:

Data Source The source of the data on the display. This can be one of the following:

JOB CTL, BLOCKS - The data set is open; the analysis data is taken from the control blocks of the address space that is being monitored.

SYSTEM, CATALOG - The data set is closed; the analysis data is taken from the VSAM catalog and may reflect operations that have been performed by other jobs after the target data set was closed.

Status The data set status. Possible values are:

OPEN-IN
OPEN-OUT
NOT-OPEN

Type The data set type. Possible values are

KSDS
ESDS
RRDS
LDS

Format The data set format. Possible values are:

blank
XF (eXtended Format)
XFCMP (eXtended Format Compressed)

Access Access method options. Possible values are a combination of the following:

DF (Deferred writes)
GSR (Global Shared Resources)
LSR (Local Shared Resources)
NSR (Non-Shared Resources)
RLS (Record Level Sharing)

Opts The data set options. Possible values can be a combination of the following access:

AD (Addressed)
CNV (Control interval)
DIR (Direct)
KEY (Key)
SEQ (Sequential)
UBF (User buffers)

LSR The resource pool ID. Possible values are blank or a number from 1 to 255.

A numeric value from 1 to 255 indicates that VSAM Local Shared Resource Pools information is available. Cursor-select this field to drill down to the VSAM Local Shared Resource Pools display.

Misc Miscellaneous options. Possible values are a combination of 31B (31-bit), CBIC (Control blocks in common), CFX (Control blocks fixed), DSNSHR (Data set name sharing), IMB (IMBED), JRND (JOURNAD exit specified), KYR (Key range), REP (REPLICATE), RES (Open with reset), SIS (Sequential Insert Strategy), SKP (Skip sequential processing), SPN (Spanned) and UPAD (UPAD exit specified)

DATA: The following fields describe the data portion of the VSAM cluster:

CIFR Data set CI free space percentage.
CAFR Data set CA free space percentage.
CISP Number of CI splits for the data set.
A non-zero value in this field can indicate poor performance.
CASP Number of CA splits for the data set.
A non-zero value in this field can indicate poor performance.
CISZ The size of a control interval.
CI-CA Number of control intervals in a control area.
KeyIn Length of the key if data set Type is KSDS.
Keyps The position of the key's offset in the record if data set Type is KSDS.
MaxIr Maximum logical record length.
EXTNT Number of extents that the data set occupies.
EXCPs Number of I/O operations issued against the data set.
Reccount Number of records in the data set.
Retrved Number of records retrieved during the displayed monitoring interval.
Updated Number of records updated during the displayed monitoring interval.
Deleted Number of records deleted during the displayed monitoring interval.
Insrted Number of records inserted during the displayed monitoring interval.

INDEX: The following fields describe the index portion of the VSAM cluster:

CIFR Index CI free space percentage.
CAFR Index CA free space percentage.
CISP Number of CI splits for the data set.
A non-zero value in this field can indicate poor performance.
CASP Number of CA splits for the data set.
A non-zero value in this field can indicate poor performance.
CISZ Index control interval size.
CA Number of index CIs in a CA.
LVL Number of index levels.
EXT Number of extents that the index occupies.
Reccnt Number of index records.
EXCP Number of I/O operations issued against the index.

You can cursor-select a value in the LSR field and press Enter to drill down to the **VSAM Local Shared Resource Pools** report as shown in Figure 10-24.

```
FBIP1C00 IBM AM for z/OS - VSAM Local Shared Resource Pools ------------------
         2004/10/26 16:57:16 to 2004/10/26 17:49:23       Row 1 of 22         >

     System: STLABF6          Jobname: CICSC31F        Stepname: CICSC31F
   Duration: 00:52:06

   User Key Str Type    Buf    Buf   Hbuf ------ Gets - ------ Puts - -Hiper Re
ID Cnt  Len Nbr         Len    Nbr   Nbr  --IO-- ------ -User- ------ --OK-- --
-- ---- --- --- ---- ------ ------ ------ ------ ------ ------ ------ ------ --
 1    2 255  50 D       512     10      0      0      0      0      0      0
 1    2 255  50 D     1,024     10      0      0      0      0      0      0
 1    2 255  50 D     2,048     10      0      0      0      0      0      0
 1    2 255  50 D     4,096     10      0      1    164      0      0      0
 1    2 255  50 D     8,192     10      0      1      3      0      0      0
 1    2 255  50 D    12,288     10      0      0      0      0      0      0
 1    2 255  50 D    16,384     10      0      0      0      0      0      0
 1    2 255  50 D    20,480     10      0      4    313      4      0      0
 1    2 255  50 D    24,576     10      0      0      0      0      0      0
 1    2 255  50 D    28,672     10      0      0      0      0      0      0
 1    2 255  50 D    32,768     10      0      0      0      0      0      0
 1    2 255  50 I       512      5      0      4    320      1      0      0
 1    2 255  50 D     1,024      5      0      0      0      0      0      0
 1    2 255  50 D     2,048      5      0      0      0      0      0      0
Command ===> _____________________________________________ Scroll ==> CSR
                                                                     24/015
```

Figure 10-24 VSAM Local Shared Resource Pools

The **>** character on the second line indicates that you can press PF11 to scroll left to view more data on the right.

Here is the relevant information about the main fields:

ID The pool ID of the local shared resource pool to which this data is related.

User Cnt The number of ACBs opened for LSR with this pool ID.

Key Len The maximum key length of the data sets that share the resource pool.

Str Nbr The total number of placeholders needed for all the data sets that share this resource pool.

This number should equal the potential number of requests that can be issued concurrently for all the data sets sharing the pool.

Type The type of component that uses the resource pool. Possible values are:

ANY (Both data and index)
DATA (Only data)
INDEX (Only index)

Buf Len The length of the buffers in this pool.

Buf Nbr The number of virtual buffers of this length in this pool.

Hbuf Nbr The number of hyperspace buffers of this length in the pool.

Gets IO The number of buffer reads that caused an I/O operation.

A high value in this field can indicate poor performance.

Gets ------ The number of buffer reads that were satisfied without causing an I/O (the number of look asides).

A low value in this field can indicate poor performance. The Gets----- / Gets IO ratio must be as high as possible.

Puts User The number of user-initiated write operations.

Puts ------ The number of non-user-initiated write operations.

Hiper Reads OK The number of successful reads from hiperspace.

A low value in this field can indicate poor performance. The Hiper Reads OK / Hiper Read ----- ratio must be as high as possible.

Hiper Reads ----- The number of failed reads from hiperspace.

A high value in this field can indicate poor performance.

Hiper Writes OK The number of successful writes to hiperspace.

A low value in this field can indicate poor performance. The Hiper Writes OK / Hiper Writes ----- ratio must be as high as possible.

Hiper Writes ----- The number of failed writes to hiperspace.

A high value in this field can indicate poor performance.

11

Application Monitor: Analyzing a job with subsystems

This chapter describes reports provided by Application Monitor when the monitored application uses a subsystem or a combination of subsystems like CICS, DB2, and IMS. It describes how to customize Application Monitor for each subsystem, and presents a detailed view of the reports generated. These reports are the same for both real-time and history data set analysis, but extra options are available if you are using history data set.

11.1 CICS analysis

In this section we describe how to customize CICS in order to analyze a CICS transaction. We do not spend much effort on the things that are common to Application Monitor and that have been described in the previous chapter, but rather focus on the unique aspects of using Application Monitor in a CICS environment.

11.1.1 CICS customization

This section describes the tasks for configuring the optional CICS component. After the base product has been installed, repeat the following steps for each CICS subsystem where Application Monitor CICS support is desired. These steps are generally performed by your systems programmer.

1. Update the CICS system definition data set, using member FBICIC4J in your hlq.SFBIJCL library. This member needs first to be customized. It creates five new definitions (two programs and three transactions) in a CICS group named FBI210.

   FBIMEI20 program This program enables Application Monitor to gather data samples from the CICS exits.

   FBIMPL20 program This program processes the FBII, FBIO, and FBIR transactions.

   FBII transaction This transaction enables Application Monitor on the target CICS subsystem.

   FBIO transaction This transaction deactivates Application Monitor on the target CICS subsystem.

   FBIR transaction This transaction removes Application Monitor from the target CICS.

2. Add group FBI210 to one of the active lists.
3. Update the CICS Program List Table Post Initialization (DFHPLTxx) with the following lines:

```
DFHPLT TYPE=INITIAL,SUFFIX=xx
DFHPLT TYPE=ENTRY,PROGRAM=yyyyyyyy
.................................
DFHPLT TYPE=ENTRY,PROGRAM=DFHDELIM
DFHPLT TYPE=ENTRY,PROGRAM=FBIMPL20
DFHPLT TYPE=ENTRY,PROGRAM=zzzzzzzz
.................................
DFHPLT TYPE=FINAL
```

   This table will start Application Monitor during the CICS initialization.

4. Updated the CICS DFHRPL concatenation with the following libraries:

   hlq.SFBIAUTH

> **Note:** If you want to run the IVPs:
>
> 1. Edit, customize, and submit sample job FBIIVSCR in hlq.SFBIIVP. (It creates the VSAM file used during IVPs.)
> 2. Edit and customize member FBIIDCI2 in hql.SFBIIVP (used as input to FBIDCIC).
> 3. Edit, customize, and submit sample job FBIIDCIC in hlq.SFBIIVP to create the CICS resource definition in a CICS group named FBI210VP.
> 4. Add FBI210VP to one of the active lists.
> 5. Add hlq.SFBIILOD in your DFHRPL concatenation.

11.1.2 CICS displays

IVPs are a set of two CICS transactions, FBI1 and FBI2, which are used to test Application Monitor under CICS.

FBI1 displays the map and enables you to specify the following three options:

- Initiate the FDR2 transaction. (FBI2 and FDR2 are the same transactions.)
- Write to a Temporary Storage queue.
- Write and delete records from the FBIFILE data set.

FBI2 only issues SEND MAP and RECEIVE MAP commands.

Since IVP only uses FBI1 and FBI2 transactions, and you know which CICS will be used, select option **2 Analyze a Specified Job**, and fill the screen as shown in Figure 11-1.

```
FBIP2000 IBM AM for z/OS - Analyze a Specified Job ---------------------------

Option ===> ____________________________________________________________________
                                                                   More:     +
Specify the Job to analyze and then select an option below

   Job Name  ==> CICSC22F (Full name or mask ending in "*")
   Current Step CICSC22F Procstep CICS Asid 005E System STLABF6

   Batch: Step Name        ==> *_______ Proc Step Name      ==> *________
   CICS : Transaction Name ==> FBI*____ Transaction Number  ==> ________
   IMS  : Message Id       ==> ________
   Enter / to select special types of analysis:
     Delays          _
     DB2 activity:   _      DB2 Subsystem Id ==> ____
     Data Sets:      _

   History DSN  ==> 'CHABERT.AM.CICSC22F.D2004301.T1725384'_______

   Max Duration ==> 90_ (1-999 minutes)      Max Samples   ==> 10000__


   1 Analyze the specified Job
   X Go back without analyzing (PF3)
                                                                     03/014
```

Figure 11-1 Analyze a Specified Job

CICS: Transaction Name contains FBI*, so only transactions that match the generic mask will be monitored.

Choose option **1 Analyze the Specified Job** to begin analyzing CICS; the panel shown in Figure 11-2 is returned.

```
FBIP1A00 IBM AM for z/OS - Active Monitor Sessions ---------------------------
         2004/10/27 17:31:58 to 2004/10/27 17:31:59       Row 1 of 1

  Named Set: N/A
System   Jobname  Stepname Pgm  CICS IMS  DB2  Dly  Dsn  CPU%   Sample Duration
-------- -------- -------- Rpts Rpts Rpts Rpts Rpts Rpts ------  Count --------
STLABF6  CICSC22F CICSC22F    0 YES                        0.00      0 00:00:00
******************************** BOTTOM OF DATA ********************************

Command ===> _____________________________________________ Scroll ==> CSR
                                                                     24/015
```

Figure 11-2 Active Monitor Sessions

In another session, log onto the target CICS subsystem and issue the FBI1 transaction ID. Run several options and return to the **Active Monitor Sessions** panel.

11.1.3 CICS Transactions display

Select a YES value in the CICS Rpts column on the Active Monitor Sessions display to bring up the **CICS Transactions** display as shown in Figure 11-3. This display lists all the transactions that are or have been active in the target CICS region during the displayed monitoring interval.

```
FBIP1B00 IBM AM for z/OS - CICS Transactions --------------------------------
         2004/10/27 17:31:58 to 2004/10/27 17:36:10       Row 1 of 2

   CICS Job: CICSC22F     CICS APPLID: CICSC22F          System: STLABF6

Tran Program   Count  Busy%
---- -------- ------ ------
FBI1 FBIDDTR1    102  88.82 ==============>
FBI2 FBIDDTR2     19  11.17 >
******************************** BOTTOM OF DATA ********************************

Command ===> _____________________________________________ Scroll ==> CSR
                                                                     24/015
```

Figure 11-3 CICS Transactions

As expected, only FBI* transactions have been monitored. This report is very easy to read and gives you information about how often each transaction was executed and an idea of how much time it took.

Here is the relevant information about the main fields:

Tran This column lists the transaction IDs of all transactions that have been active in the target region during the displayed monitoring interval.

Program The name of the first program that CICS dispatches to process the associated transaction ID.

Count The number of instances of the associated transaction ID that are or have been executing in the target region during the displayed monitoring interval.

Busy% The percentage of the elapsed time of the monitoring interval that this transaction was active with respect to all other transactions listed. Values in this column should total approximately 100% due to rounding.

The ===> provides a graphical indicator of this percentage so that you can easily spot the transactions that are the most active.

How to use this display

- Cursor-select a transaction ID in the Tran column and press Enter to bring up the **CICS Calls in a Transaction** display. This display shows the calls that all instances of the target transaction have made to CICS during the displayed monitoring interval.
- Cursor-select a program in the Program column and press Enter to drill down to program analysis displays. The first display that appears is **CPU and Loadmod Analysis.**
- Cursor-select an item in the Count column and press Enter to bring up the **CICS Transaction Executions** display. This display provides a list of the active instances of the target transaction.

11.1.4 CICS Calls in a Transaction display

Figure 11-4 shows the **CICS Calls in a Transaction** display. This display shows the elapsed wall-clock time that all the instances of a CICS call from the same location in the program spent in application code before the call was issued and spent in CICS code to execute the call's function. You reach this display by cursor-selecting FBI1 in the Tran column on the CICS Transactions display.

```
FBIP1C00 IBM AM for z/OS - CICS Calls in a Transaction ------------------------
         2004/10/27 17:31:58 to 2004/10/27 17:54:32       Row 18 of 34

   CICS Job: CICSC22F     CICS APPLID: CICSC22F          System: STLABF6
Transaction: FBI1          CICS calls: 17

Program  Offset   Count Activity                     Total time
-------- -------- ----- ---------------------- ---------------
FBIDDTR1 00000A9A    27 EXEC CICS                     0.001236
FBIDDTR1 00000A3C    27   FORMATTIME                  2.768963 >
FBIDDTR1 00000A3A    27 EXEC CICS                     0.001073
FBIDDTR1 00000942    27   ASKTIME ABSTIME             3.074125 >
FBIDDTR1 00000940    27 EXEC CICS                     0.001194
FBIDDTR1 00000904    19   RETURN                      0.000000
FBIDDTR1 00000902    19 EXEC CICS                     0.000880
FBIDDTR1 00000796    83   RECEIVE MAP                 5.765129 ==>
FBIDDTR1 00000794    83 EXEC CICS                     0.003835 .
FBIDDTR1 000006DC    83   HANDLE AID                  3.932547 =>
FBIDDTR1 000006DA    83 EXEC CICS                     0.000000
FBIDDTR1 0000068A    83   SEND MAP                   12.427105 ======>
FBIDDTR1 00000688    83 EXEC CICS                     0.003023
DFHPEP   000000CC     1   RETURN                      0.000000
DFHPEP   000000CA     1 EXEC CICS                     0.000031
Command ===> ____________________________________________ Scroll ==> CSR
                                                                 24/015
```

Figure 11-4 CICS Calls in a Transaction

Here is the relevant information about the main fields:

CICS Calls The number of places in the program where EXEC CICS calls were issued during the displayed monitoring interval. If multiple calls (see the Count column) were issued from the same program location/offset, they are counted as one call in the CICS Calls value.

Program This column shows the name of the programs that issued the associated calls to CICS.

Offset The offset (hexadecimal) within the program where the call is executed. The offset in the first row for the associated call gives the location where the program made the call to CICS.
The offset in the second row for the associated call gives the location where the program received control back from CICS.

Count Count value gives the number of instances of the call that have been issued from this program/offset location during the displayed monitoring interval.

Activity
Total Time These columns give two rows of information about each CICS call. The first row shows EXEC CICS. The associated Total Time value shows the elapsed wall-clock time that the transaction spent in application code before this call was issued.

The second row shows the EXEC CICS call function. The associated Total Time value shows the elapsed wall-clock time that the transaction spent in CICS to process the instances of this call. Total Time is displayed in the format s.tttttt, where s is the number of seconds, and tttttt is millionths of a second.

The ===> indicator to the right of Total Time provides a graphical representation of the Total Time with respect to the other Total Time values on the display. This indicator enables you to quickly spot the program/offset locations where the most time is spent in application code and CICS code respectively.

Note: If UNKNOWN xx yy appears in the Activity column, an undocumented CICS call has taken place. This is usually in IBM-supplied components that are object code only. If this appears in application code, a probable storage corruption has taken place. Examine your compiled source code and verify that the correct command code has been generated.

In the example shown in Figure 11-4, Total time for an EXEC CICS SEND MAP is more than 12 of the elapsed wall-clock. It could be useful to analyze this SEND MAP to see if a SEND MAP DATAONLY or a SEND MAP MAPONLY could be used instead of SEND MAP and check if the Total time is decreased.

How to use this display

- Cursor-select the Program name and press Enter to bring up the program analysis displays. The first display that appears is the **CPU and Loadmod Analysis** display.
- Cursor-select an item in the Taskid column and press Enter to bring up the **CICS Calls in a Unit of Work** display.

11.1.5 CICS Transaction Executions display

Figure 11-5 shows the **CICS Transaction Executions** display. This display provides information about each instance of the target transaction that is active during the displayed monitoring interval. You reach this display by cursor-selecting an item in the Count column on the CICS Transactions Executions panel.

```
FBIP1C00 IBM AM for z/OS - CICS Transaction Executions ------------------------
         2004/10/27 17:31:58 to 2004/10/27 18:09:24       Row 1 of 104

   CICS Job: CICSC22F     CICS APPLID: CICSC22F          System: STLABF6
Transaction: FBI1         1st Program: FBIDDTR1

Started       Taskid Userid   Term  Time in Appl.         Time in CICS
----------- -------- -------- ---- --------------        --------------
17:34:58.93    25023 CHABERT  0029        0.00035               11.58427 =====>
17:35:13.71    25040 CHABERT  0029        0.00035               10.56273 ====>
17:38:31.33    25084 CHABERT  0029        0.00555 =====>         5.54218 =>
17:35:29.88    25057 CHABERT  0029        0.00021                3.66660 >
17:34:27.58    24993 CHABERT  0029        0.00025                0.01102 .
17:34:23.34    24988 CHABERT  0029        0.00023                0.00431 .
17:34:30.96    24997 CHABERT  0029        0.00024                0.00431 .
17:34:28.45    24994 CHABERT  0029        0.00022                0.00426 .
17:34:44.97    25010 CHABERT  0029        0.00026                0.00398 .
17:34:42.64    25008 CHABERT  0029        0.00024                0.00391 .
17:34:26.75    24992 CHABERT  0029        0.00027                0.00390 .
17:34:43.87    25009 CHABERT  0029        0.00024                0.00388 .
17:34:24.20    24989 CHABERT  0029        0.00024                0.00387 .
17:34:39.98    25006 CHABERT  0029        0.00023                0.00387 .
17:34:35.14    25001 CHABERT  0029        0.00024                0.00386 .
Command ===> _____________________________________________ Scroll ==> CSR_
                                                                    24/015
```

Figure 11-5 CICS Transaction Executions

Figure 11-5 gives you information about each instance of the transaction that was active during your analysis period. It allows you to look at the information for each specific instance of the transaction. In this example the data has been sorted in descending sequence by Time in CICS. It is easy to see which task spent the most time in CICS and to drill down to the CICS Calls in a Unit of Work by cursor-selecting its Taskid (in this example, Taskid 25023).

Here is the relevant information about the main fields:

1st Program The name of the first program that is dispatched to process the target transaction. (As the transaction executes, it may invoke other programs.)

Started The time that Application Monitor first noticed this transaction instance and started to monitor it, in hh:mm:ss.tt format, where tt is hundredths of a second.

Taskid The number that CICS assigned to this instance of the transaction.

Note: Multiple rows with the same Taskid identify a pseudo conversational transaction, and a row exists for each task in the pseudo conversational sequence.

Time in Appl The amount of time that the associated transaction instance spent in application code between calls to CICS, in hh:mm:ss.ttttt format, where ttttt is hundred-thousandths of a second.

Time in CICS The amount of time that the associated transaction instance spent in CICS code, in hh:mm:ss.ttttt format, where ttttt is hundred-thousandths of a second.

11.1.6 CICS Calls in a Unit of Work display

Figure 11-6 shows the **Calls in a Unit of Work** display. This display provides information about all the calls processed in the target transaction instance. You reach this panel by cursor-selecting an entry in the Taskid column on the CICS Transaction Executions display.

```
FBIP1D00 IBM AM for z/OS - CICS Calls in a Unit of Work ----------------------
         2004/10/27 17:31:58 to 2004/10/27 18:12:26       Row 1 of 14

   CICS Job: CICSC22F     CICS APPLID: CICSC22F          System: STLABF6
Transaction: FBI1              Userid: CHABERT      Terminal id: 0029
Task id no.: 25023        Trans Start: 17:34:58.933   Trans End: 17:35:10.518
 CICS calls: 7
Program  Offset   Count Activity                    Total time
-------- -------- ----- --------------------- ---------------
FBIDDTR1 00000C26     1 EXEC CICS                     0.000058
FBIDDTR1 00000C28     1   RETURN                      0.000000
FBIDDTR1 00000688     1 EXEC CICS                     0.000063
FBIDDTR1 0000068A     1   SEND MAP                    4.887696 ======>
FBIDDTR1 00000A9A     1 EXEC CICS                     0.000061
FBIDDTR1 00000A9C     1   WRITEQ TS                  1.775067 =>
FBIDDTR1 00000A3A     1 EXEC CICS                     0.000053
FBIDDTR1 00000A3C     1   FORMATTIME                  1.001544 >
FBIDDTR1 00000940     1 EXEC CICS                     0.000062
FBIDDTR1 00000942     1   ASKTIME ABSTIME             1.189049 >
FBIDDTR1 00000794     1 EXEC CICS                     0.000058
FBIDDTR1 00000796     1   RECEIVE MAP                 1.697938 =>
FBIDDTR1 000006DA     1 EXEC CICS                     0.000000
FBIDDTR1 000006DC     1   HANDLE AID                  1.032982 >
Command ===> _____________________________________________ Scroll ==> CSR
                                                                   24/015
```

Figure 11-6 CICS Calls in a Unit of Work

Here is the relevant information about the main fields:

Trans Start The time that Application Monitor first noticed this transaction instance and started to monitor it, in hh:mm:ss.ttt format, where ttt is thousandths of a second.

Trans End The time that Application Monitor last saw this instance of the transaction active in the region, in hh:mm:ss.ttt format, where ttt is thousandths of a second.

CICS Calls The number of CICS calls that were processed by this transaction instance during the displayed monitoring interval.

Program The name of the program that issues the associated call.

Offset The offset (hexadecimal) within the program where the associated call is made.

Count This value gives the number of instances of the call that have been issued from this program/offset location during the displayed monitoring interval.

Activity The text of the CICS call. Information about each call is on two lines.

Total Time The first line shows EXEC CICS. The associated Total Time value shows the elapsed wall-clock time that the target transaction instance spent in application code before this call was issued.

The second line shows the EXEC CICS call function and the associated elapsed wall-clock time that the target transaction instance spent in CICS to process the instances of this call.

Total Time is displayed in the format s.tttttt, where s is the number of seconds, and tttttt is millionths of a second. The ===> indicator to the right of Total Time provides a graphical representation of the Total Time with respect to the other Total Time values on the display. This indicator enables you to quickly spot the program/offset locations where the most time is spent in application code and CICS code respectively.

You can cursor-select the Program name and press Enter to bring up the program analysis displays. The first display that appears is the **CPU and Loadmod Analysis** display.

11.2 DB2 analysis

In this section we describe how to customize DB2 in order to analyze the display panels that are available if the job you are monitoring accesses a DB2 database.

11.2.1 DB2 customization

Which authorization ID to use

Before running the tasks described in the following sections, you must choose a unique authid. The authid is the same authid that you will use for the DB2 AAS RACF STARTED class. You must use the same authid to bind a DB2 plan and create a PLAN_TABLE.

Creating and binding a DB2 UDB access plan

To create and bind a DB2 UDB access plan:

1. Edit member FBIPLINS in your hlq.SFBIJCL library.
2. Customize FBIPLINS, for the authid in the QUALIFIER keyword, using the same user ID that you defined for the DB2 AAS RACF STARTED class.
3. Submit the sample job. You will get a return code of 0 if the job runs correctly.

Enabling the use of dynamic explain

To use *dynamic explain* with the Application Monitor DB2 UDB component, Application Monitor started tasks must have access to a PLAN_TABLE.

1. Create PLAN_TABLE in an appropriate database and tablespace. You can use the sample Data Definition Language (DDL) in member FBIPLCR8 of hlq.SFBIJCL to create PLAN_TABLE.
2. Edit, customize and submit job FBIPLCR8 for Application Monitor started tasks to access PLAN_TABLE.

   For the userid in the table name, use the same user ID that you defined for the DB2 AAS RACF STARTED class.
3. Issue a GRANT to make PLAN_TABLE accessible to other users.

Setting up the DB2 Associated Address Space (AAS)

When you analyze DB2 activity, the Application Monitor subsystem creates an Associated Address Space (AAS). Its name is composed of a four-character prefix (FBID is the default) followed by the DB2 subsystem ID.

To set up the DB2 AAS so that Application Monitor can collect DB2 analysis data, perform the following steps:

1. Edit and customize member FBIDBAAS in your hlq.SFBIJCL library.

   Change *baas* on the PROC and EXEC statements to the three-character or four-character subsystem ID.
2. Place this procedure in a PROCLIB that is accessible to JES. Rename the member FBIDBAAS to FBIDxxxx, where FBID is the default prefix and xxxx is your three-character or four-character subsystem ID.

   Create a separate procedure named FBIDxxxx for each DB2 subsystem from which you want IBM Application Monitor to collect DB2 analysis data.
3. Use the same user ID for the FBIDxxxx started task and for the plan.
4. Grant the user ID READ access to Application Monitor libraries.
5. Define a profile in the RACF STARTED class for the FBIDxxxx started task so that it can access protected resources.

Note: Do not start these procedures. The Application Monitor started tasks start them automatically when they are needed.

Enabling DB2 AAS to use DB2 trace data

Enter the following DB2 GRANT statements to allow the DB2 AAS to use DB2 trace data:

- GRANT TRACE TO *authid*
- GRANT MONITOR2 TO *authid*
- GRANT ALL ON authid.PLAN_TABLE TO *authid*

where *authid* is the user ID defined in the RACF STARTED class profile for DB2 AAS.

Setting CICS/DB2 interface

This step is required if you are monitoring CICS/DB2 transactions.There are three CICS resources that must be set correctly to allow Application Monitor to collect DB2 information while using DB2.

Note: If these objects are not set correctly, you might not see any DB2 reports or SQL call statements in the CICS reports.

The three objects are:

DB2CONN Defines the CICS connection to DB2.

DB2ENTRY Defines the resources to be used by a specific transaction or by a group of transactions when accessing DB2. The parameter values entered for DB2ENTRY override those in DB2CONN.

DB2TRAN Associates a transaction or group of transactions with a DB2ENTRY. The parameter values entered for DB2TRAN override those in DB2CONN and DB2ENTRY.

You must consider how they work together for the transactions you are monitoring. The key point is the **ACCOUNTREC** parameter.

ACCOUNTREC To allow Application Monitor to collect DB2 information, there is a specific requirement for coding the ACCOUNTREC parameter. You can set this parameter on the DB2CONN and DB2ENTRY objects. The object must have a value set other than None. Possible values are: None, TXid, TAsk and Uow

Note: If the ACCOUNTREC parameter has a value of None at the highest level, you will not see any DB2 reports or SQL call statements in the CICS reports.

11.2.2 DB2 displays

Application Monitor collects DB2 analysis data only if you have specifically requested it. You can request DB2 analysis on the following panels:

- **Specify Default Sampling Parameters** panel
- **Analyze a Specified Job** panel
- **Specified Deferred Job Details** panel

In the following example we use the IVP to monitor batch jobs accessing DB2 data as shown in Figure 11-7.

```
FBIP4200 IBM AM for z/OS - Specify Deferred Job Details ----------------------

Command ===> ____________________________________________________________________


Complete the job details below and then return to Deferred Monitoring (PF3)

   Job Name    ==> FBIRUN2        Select:
   System      ==> ADT1            / DB2 analysis  DB2 SSID ==> DB0A
                                   _ Delay Analysis
                                   _ Data Sets Analysis

   Job Step    ==> *       Proc Step ==> *
   CICS Tran   ==> ________
   IMS Message ==> ________


Enter END (PF3) to go back

                                                                        11/040
```

Figure 11-7 Specify Deferred Job Details

You get have the same reports when you monitor a job in a real time or when you analyze historical data. In this example we use a history data set a shown in Figure 11-8.

```
FBIP1H10 IBM AM for z/OS - Historical Reports Available ----------------------
         2004/10/28 19:59:33 to 2004/10/28 20:01:23       Row 1 of 1
History Data Set: ERIC.AM.FBIRUN2.D2004302.T1959057
  Interval Start: 2004/10/28  19:59:33
   Interval Stop: 2004/10/28  20:00:40

System   Jobname  Stepname Pgm  CICS IMS  DB2  Dly  Dsn  CPU%   Sample
-------- -------- -------- Rpts Rpts Rpts Rpts Rpts Rpts ------  Count
ADT1     FBIRUN2  DB2RR      28           YES              6.17  7,277
****************************** BOTTOM OF DATA ******************************




Command ===> ____________________________________________ Scroll ==> CSR
                                                                        24/015
```

Figure 11-8 Historical Reports Available

The following displays are available for DB2 analysis; the following sections describe these displays in detail:

- SQL Analysis
- SQL Statement Summary
- SQL Statement View
- SQL Statement Full Text

- SQL Detailed Explain Analysis
- SQL Explain Detail
- SQL End Completion Statistics
- SQL Statement Completion
- DB2 Table Information

11.2.3 SQL Analysis display

The SQL Analysis display is the first display that is available for DB2 analysis; an example is shown in Figure 11-9. Cursor-select a YES entry in the DB2 Rpts column and press Enter to drill down to it.

It identifies the application plan and Database Request Modules (DBRMs) that are created during the program development process and that provide information about the SQL statements that are executed in the target job.

From the panel shown in Figure 11-9 you can drill down to display the text of a selected SQL statement, statistics about statement execution, and statement completion information that is returned to the program in the SQL Communication Area (SQLCA).

```
FBIP1C00 IBM AM for z/OS - SQL Analysis -------------------------------------
         2004/10/28 19:59:33 to 2004/10/28 20:00:40       Row 1 of 23
History Data Set: ERIC.AM.FBIRUN2.D2004302.T1959057
    Jobname: FBIRUN2          Stepname: DB2RR              System: ADT1
    LU Name: A6P2T101         DB2 SSID: DB0A               Module: FBIDBTBD
  Plan Name: FBIDBTBD         Auth Id.: ERIC             Oper Id.: ERIC
DB2  Auth-ID  Plan     DBRM     St No O SQLRC Type     Count Rel. DB2 CPU Time
---- -------- -------- -------- ----- - ----- -------- ----- <--------------->
DB0A ERIC     FBIDBTBD FBIDBTBD   160 S  -811 SELECT     198 .
DB0A ERIC     FBIDBTBD FBIDBTBD   245 S   100 FETCH      198 .
DB0A ERIC     FBIDBTBD FBIDBTBD   350 S   100 FETCH      198
DB0A ERIC     FBIDBTBD FBIDBTBD   360 S     0 FETCH      198 .
DB0A ERIC     FBIDBTBD FBIDBTBD   462 S     0 FETCH    9,478 .
DB0A ERIC     FBIDBTBD FBIDBTBD   472 S   100 FETCH      198 .
DB0A ERIC     FBIDBTBD FBIDBTBD   534 S     0 SELECT     19K .
DB0A ERIC     FBIDBTBD FBIDBTBD   502 S     0 CLOSE      198 .
DB0A ERIC     FBIDBTBD FBIDBTBD   472 S     0 FETCH    9,278 .
DB0A ERIC     FBIDBTBD FBIDBTBD   446 S     0 OPEN       196 .
DB0A ERIC     FBIDBTBD FBIDBTBD   430 S     0 PREPARE    194 .
DB0A ERIC     FBIDBTBD FBIDBTBD   397 S     0 CLOSE      197 .
DB0A ERIC     FBIDBTBD FBIDBTBD   394 S     0 CLOSE      198 .
DB0A ERIC     FBIDBTBD FBIDBTBD   350 S     0 FETCH      198 .
DB0A ERIC     FBIDBTBD FBIDBTBD   334 S     0 OPEN       197 .
Command ===> ___________________________________________________ Scroll ==> CSR
                                                                     24/015
```

Figure 11-9 SQL Analysis

Here is the relevant information about the main fields:

Module The name of the target load module.

Oper Id The user ID associated with the submitter of the job.

Auth-Id The DB2 authorization ID that is used when the associated SQL statement is executed.

Plan Name of the application plan.

DBRM Name of the DBRM that is used. The DBRM contains information about the associated SQL statement.

St No. A number that identifies the statement being explained.

> **Note:** If the EXPLAIN statement is embedded in the application program or if the EXPLAIN(YES) option is specified when the package or plan is bound or rebound, the statement number displayed in this field is the one that is assigned by the DB2 precompiler and placed in the target DBRM. Cursor-select an entry in this column to drill down to the **SQL Statement Summary** display.

O This column identifies the technique in which the SQL statement is implemented within the program: D (Dynamic SQL) or S (Static SQL).

SQLRC The SQL return code for this statement.

> **Note:** Cursor-select an entry in this column to drill down to the **SQL Statement Completion** display.

Type The type of operation that the SQL statement performs.

Count The number of times the program has issued this statement.

Rel. DB2 CPU Time The percentage of the program's CPU time that the collective executions of this statement have used processing this DB2 statement. The values in this column add up to approximately 100 percent (because of rounding) for all SQL statements issued from the target program.

How to use this display

- Cursor-select an entry in the St No column to drill down to the **SQL Statement Summary** display.
- Cursor-select an entry in the SQLRC column to drill down to the **SQL Statement Completion** display.
- Cursor-select an entry in the Count column to drill down to the **SQL End Completion Statistics** display.

Select statement number 334 to drill down to the **SQL Statement Summary** display described in the next section.

11.2.4 SQL Statement Summary display

Figure 11-10 shows the text of the target SQL statement.

> **Note:** If a DECLARE CURSOR statement is associated with the target SQL statement, the first line of the text of the DECLARE CURSOR statement is also shown.

```
FBIP1D00 IBM AM for z/OS - SQL Statement Summary ----------------------------
         2004/10/28 19:59:33 to 2004/10/28 20:00:40       Row 1 of 2
History Data Set: ERIC.AM.FBIRUN2.D2004302.T1959057
    Jobname: FBIRUN2         Stepname: DB2RR             System: ADT1
    LU Name: A6P2T101        DB2 SSID: DB0A              Module: FBIDBTBD
  Plan Name: FBIDBTBD        Auth Id.: ERIC            Oper Id.: ERIC
  Statement: 334                 Type: OPEN
X  <-------------------- S T A T E M E N T   T E X T------------------------->
.  OPEN C29
.  DECLARE C29 CURSOR FOR SELECT I . NAME , I . TBNAME , I . UNIQUERULE FROM
****************************** BOTTOM OF DATA ********************************
Command ===> _____________________________________________ Scroll ==> CSR
                                                                     24/015
```

Figure 11-10 SQL Statement Summary

Here is the relevant information about the main fields:

Type The type of SQL statement (OPEN, SELECT, INSERT, and so on).

X Explain analysis selection column.

Statement Text The text of the target SQL statement.

Note: Associated DECLARE statements will also be listed if located.

How to use this display:

- Place the cursor on the X selection field and press Enter to bring up the **SQL Detailed Explain Analysis** display for the target statement.
- If the target statement is too long to be displayed entirely, place the cursor on this field and press Enter to bring up the **SQL Statement Full Text** display.

Drill down to the **SQL Statement Full Text** display.

11.2.5 SQL Statement Full Text

Figure 11-11 shows the full text of the target SQL statement.

Note: The full text of the associated DECLARE CURSOR statement is also shown if applicable.

```
FBIP1D00 IBM AM for z/OS - SQL Statement Full Text ---------------------------
         2004/10/28 19:59:33 to 2004/10/28 20:00:40       Row 1 of 7
History Data Set: ERIC.AM.FBIRUN2.D2004302.T1959057
    Jobname: FBIRUN2         Stepname: DB2RR             System: ADT1
    LU Name: A6P2T101        DB2 SSID: DB0A              Module: FBIDBTBD
  Plan Name: FBIDBTBD        Auth Id.: ERIC            Oper Id.: ERIC
  Statement: 334                 Type: OPEN
X  <-------------------- S T A T E M E N T   T E X T------------------------->
.  OPEN C29
     V0001: TBT
.  DECLARE C29 CURSOR FOR SELECT I . NAME , I . TBNAME , I . UNIQUERULE FROM
.  SYSIBM . SYSINDEXES I , SYSIBM . SYSTABLES T WHERE I . UNIQUERULE = 'D'
.  AND I . FULLKEYCARD = T . CARD AND I . TBNAME = T . NAME AND I . CREATOR =
.   T . CREATOR AND I . TBNAME LIKE : H
     V0001: TBT
******************************** BOTTOM OF DATA ********************************




Command ===> _____________________________________________ Scroll ==> CSR
                                                                     24/015
```

Figure 11-11 SQL Statement Full Text

All the fields are the same as for SQL Statement Summary.

The only difference is that you have several rows to display the text.

Place the cursor on the X selection field and press Enter to bring up the **SQL Detailed Explain Analysis** display.

11.2.6 SQL Detailed Explain Analysis

The **SQL Detailed Explain Analysis** display provides information from the plan table that is created to hold EXPLAIN output as shown in Figure 11-12.

```
FBIP1D00 IBM AM for z/OS - SQL Detailed Explain Analysis ---------------------
         2004/10/28 19:59:33 to 2004/10/28 20:00:40       Row 1 of 2
History Data Set: ERIC.AM.FBIRUN2.D2004302.T1959057
    Jobname: FBIRUN2         Stepname: DB2RR             System: ADT1
    LU Name: A6P2T101        DB2 SSID: DB0A              Module: FBIDBTBD
  Plan Name: FBIDBTBD        Auth Id.: ERIC            Oper Id.: ERIC
  Statement: 334                 Type: OPEN
Table                Plan Method Match Acc  Join Index UQ JN OB GB   UQ JN OB GB
Name                  Nbr          Col Type Grp. Only
SYSTABLES               2      4     2 I       0 N     N  Y  N  N    N  Y  N  N
SYSINDEXES              1      0     0 R       0 N     N  N  N  N    N  N  N  N
****************************** BOTTOM OF DATA ******************************



Command ===> _____________________________________________ Scroll ==> CSR
                                                                     24/015
```

Figure 11-12 SQL Detailed Explain Analysis

Here is the relevant information about the main fields:

Table Name

The name of a table, created temporary table, declared temporary table, materialized view, table expression, or an intermediate result table for an outer join that is accessed in this step.

The value for this field is blank if Method is 3.

For an outer join, this column contains the created temporary table or the declared temporary table name of the work file in the form DSNWFQB(qblockno). Merged views show the base table names and correlation names. A materialized view is another query block with its own materialized views, tables, and so forth.

Plan Nbr

Number of the step in which the query indicated in Query Number was processed. This field indicates the order in which the steps were executed.

The Query Number is shown on the SQL Explain Detail display. Cursor-select an entry in the Plan Nbr column to drill down to it.

Method

A number that indicates the join method used for the step, as follows:

- **0** First table accessed, continuation of the previous table accessed, or not used.
- **1** Nested loop join. For each row of the present composite table, matching rows of a new table are found and joined.
- **2** Merge scan join. The present composite table and the new table are scanned in the order of the join columns, and matching rows are joined.
- **3** Sorts needed by ORDER BY, GROUP BY, SELECT DISTINCT, UNION, a quantified predicate, or an IN predicate. This step does not access a new table.
- **4** Hybrid join. The current composite table is scanned in the order of the join-column rows of the new table. The new table is accessed using list prefetch.

Match Col The number of index keys used in an index scan if Acc Type is I, I1, N, or MX. The value is zero if there was no index scan.

Acc Type The method used to access the new table, as follows:

- **I** By an index (which is identified in the Access Creator and Access Name fields on this display).
- **I1** By a one-fetch index scan.
- **M** By a multiple index scan (followed by MI, MU, or MX).
- **MI** By an intersection of multiple indexes.
- **MU** By a union of multiple indexes.
- **MX** By an index scan on the index named in the Access Name field on this display.
- **N** By an index scan when the matching predicate contains the IN keyword**.**
- **R** By a table space scan.
- **T** By a sparse index (star join work files).
- **blank** Not applicable.

Join Grp The identifier of the parallel group for joining the composite table with the new table.

Index Only Specifies if access to an index alone is enough to carry out the step or if data must also be accessed (Y=Yes; N=No).

UQ JN OB GB The first set of UQ, JN, OB, and GB fields applies to the new table (possible values Y=Yes; N=No):

- **UQ** Specifies whether the new table is sorted to remove duplicate rows.
- **JN** Specifies if the new table is sorted for join Method 2 or 4.
- **OB** Specifies whether the new table is sorted for an ORDER BY clause.
- **GB** Specifies whether the new table is sorted for a GROUP BY clause.

UQ JN OB GB The second set of UQ, JN, OB, and GB fields applies to the composite table (possible values Y=Yes; N=No):

- **UQ** Specifies whether the composite table is sorted to remove duplicate rows.
- **JN** Specifies whether the composite table is sorted for join Method 1, 2, or 4.
- **OB** Specifies whether the composite table is sorted for an ORDER BY clause or a quantified predicate.
- **GB** Specifies whether the composite table is sorted for a GROUP BY clause.

Note: Refer to the DB2 Administration Guide for the version of DB2 that you are using for a complete description of Explain analysis information and the implications for SQL statement performance.

How to use this display

- Cursor-select an item in the Table Name column and press Enter to drill down to the **DB2 Table Information** display.
- Cursor-select an item in the Plan Nbr column and press Enter to drill down to the **SQL Explain Detail** display.

We cursor-selected plan number 1, to drill down to the display shown in Figure 11-13.

11.2.7 SQL Explain Detail display

The SQL Explain Detail display, as shown in Figure 11-13, provides information from the plan table that is created to hold EXPLAIN output.

```
FBIPXPLA IBM AM for z/OS - SQL Explain Detail --------------------------------
         2004/10/28 19:59:33 to 2004/10/28 20:00:40
History Data Set: ERIC.AM.FBIRUN2.D2004302.T1959057
Command ===> _____________________________________________________________
                                                                 More:     +
    Jobname: FBIRUN2         Stepname: DB2RR             System: ADT1
    LU Name: A6P2T101        DB2 SSID: DB0A              Module: FBIDBTBD
  Plan Name: FBIDBTBD        Auth Id.: ERIC            Oper Id.: ERIC
  Statement: 334                 Type: OPEN
                          Table Name: SYSINDEXES
MixOP Seq . .: 0          Table Number.: 1          Plan Number .: 1
Query Number.: 1          Method. . . .: 0          Match Columns: 0
Acces Creator:            Access Degree: 0          Prefetch. . .: S
Access Group.: 0          Column Eval .:            Join Degree .: 0
Access Type .: R          Join Group. .: 0          Index Only. .: N
Sort New. . . . .: NNNN                Lock Mode.  . . .: IS
Sort Composite. .: NNNN                Access Name . . .:
Package Version .:                     Collection ID.  .:
Time Stamp. . . .: 2004-10-28 19.59.38

                                                                  04/015
```

Figure 11-13 SQL Explain Detail

Here is the relevant information about the main fields:

Auth ID The DB2 authorization ID that is used when the SQL statements are executed.

Oper ID The user ID associated with the submitter of the job.

Statement The SQL statement number.

Type The type of SQL statement.

Table Name Defined in the previous section.

MixOP Seq The sequence number of a step in a multiple index operation. Can have the following values:

1, 2, . . . n - For the steps of the multiple index procedure (where Access Type is MX, MI, or MU).

0 - For any other rows (where Access Type is I, I1, M, N, R, or blank).

Table Number The sequence in which the table identified in the Table Name field is processed in this query.

Plan Number Defined in the previous section.

Query Number A number intended to identify the statement being explained. For a row produced by an EXPLAIN statement, this number is specified in the QUERYNO clause. For a row produced by non-EXPLAIN statement, this number is specified in the QUERYNO clause, which is an optional part of the SELECT, INSERT, UPDATE and DELETE statement syntax. Otherwise, DB2 assigns a number based on the line number of the SQL statement in the source program.

Note: When the value of Query Number is based on the statement number in the source program, a value greater than 32767 is reported as

0. Hence, in a very long program, the value is not guaranteed to be unique. If Query Number is not unique, the value of Time Stamp is unique.

Method Defined in the previous section.

Match Columns Defined in the previous section.

Access Creator The creator of the index if an index was used to access the target table; otherwise, the value of this field is blank.

Access Degree The number of parallel tasks or operations activated by a query. This field contains zero if there is a host variable.

Prefetch Prefetch indicator. Specifies whether data pages are to be read in advance by prefetch. Can have one of the following values:
S Pure sequential prefetch.
L Prefetch through a page list.
blank Unknown or no prefetch.

Access Group The identifier of the parallel group for accessing the new table. A parallel group is a set of consecutive operations that are executed in parallel and that have the same number of parallel tasks.

Column Eval This field specifies when a SQL column function is evaluated, as follows:
R While the data is being read from the table or index.
S While a sort to satisfy a GROUP BY clause is being performed.
blank After data retrieval and after any sorts.

Join Degree The number of parallel operations or tasks used in joining the composite table with the new table.

Access Type Defined in the previous section.

Join Group The identifier of the parallel group for joining the composite table with the new table.

Index Only This field specifies whether access to an index is enough to carry out the step or whether access to data is also required (Y=Yes; N=No).

Sort New This 4-value field identifies the sort operations that are performed on the new table, in sequence from left to right, as follows:
– The first value specifies whether the new table is sorted to remove duplicate rows (Y=Yes; N=No).
– The second value specifies whether the new table is sorted for join method 2 or 4 (Y=Yes; N=No). (The join method is identified in the Method field on this display).
– The third value specifies whether the new table is sorted for ORDER BY (Y=Yes; N=No).
– The fourth value specifies whether the new table is sorted for GROUP BY (Y=Yes; N=No).

Lock Mode This field indicates the mode of lock to be acquired on either the new table or its table space or table space partitions.

If the isolation level can be determined at bind time, the possible values are:
IS Intent share lock.
IX Intent exclusive lock.
N UR isolation; no lock.
S Share lock.
SIX Share with intent exclusive lock.
U Update lock.

X Exclusive lock.

If the isolation level cannot be determined at bind time, the lock mode determined by the isolation level at run time is one of the following values:

NIS UR isolation, no lock; for CS, RR, or RS isolation, an IS lock.
NS UR isolation, no lock; for CS, RR, or RS isolation, an S lock.
NSS UR isolation, no lock; for CS or RS isolation, an IS lock; for RR isolation, an S lock.
SS CS, RS, or UR isolation, an IS lock; for RR isolation, an S lock.

If the field is all blank, no lock is acquired.

Isolation Levels

RR Allows the application to read the same pages or rows more than once without allowing any UPDATE, INSERT, or DELETE by another process. All accessed rows or pages are locked, even if they do not satisfy the predicate.
RS Allows the application to read the same pages or rows more than once without allowing qualifying rows to be updated or deleted by another process.
CS Allows maximum concurrency with data integrity.
UR Allows the application to read while acquiring few locks, at the risk of reading uncommitted data. UR isolation applies only to read-only operations (SELECT, SELECT INTO, or FETCH from a read-only result table).

Note: See the Application Programming and SQL Guide for the version of DB2 that you are using for a complete discussion of locking and isolation levels.

Sort Composite This four-value field identifies the sort operations that are performed on the composite table, in sequence from left to right, as follows:
- The first value specifies whether the composite table is sorted to remove duplicate rows (Y=Yes; N=No).
- The second value specifies whether the composite table is sorted for join method 1, 2, or 4 (Y=Yes; N=No). The join method is identified in the Method field on this display.
- The third value specifies whether the composite table is sorted for ORDER BY (Y=Yes; N=No).
- The fourth value specifies whether the composite table is sorted for GROUP BY (Y=Yes; N=No).

Access Name The name of the index if an index is used to access the target table.

Package Version Name of the package version, if any, that is bound into the target application plan.

Collection ID The name of the package collection, if any, that is bound into the target application plan.

Note: Refer to the DB2 Administration Guide for the version of DB2 that you are using for a complete description of Explain analysis information and the implications for SQL statement performance.

11.2.8 DB2 Table Information display

Figure 11-14 shows the DB2 Table Information display. This display provides information about the DB2 table that the target SQL statement is accessing. You reach this display by cursor-selecting the Table Name on the previous **SQL Detailed Explain Analysis** panel.

```
FBIPXPLA IBM AM for z/OS - DB2 Table Information ----------------------------
         2004/10/28 19:59:33 to 2004/10/28 20:00:40
History Data Set: ERIC.AM.FBIRUN2.D2004302.T1959057
Command ===> _____________________________________________________________
                                                                More:     +
    Jobname: FBIRUN2         Stepname: DB2RR            System: ADT1
    LU Name: A6P2T101        DB2 SSID: DB0A             Module: FBIDBTBD
  Plan Name: FBIDBTBD        Auth Id.: ERIC            Oper Id.: ERIC
  Statement: 334                 Type: OPEN
                          Table Name: SYSINDEXES
Plan Number .: 1          Mix OP Seq. .: 0          Table Number.: 1
Table Type. .: T          Table Stat. .: X          Edit Proc . .:
Valid Proc. .:            Database ID .: 6          Column count.: 40
Table OBID. .: 23         Percent Pages: -1         Uniq Index ID: 63
Key Columns .: 2          Parents . . .: 1          Children. . .: 4
Record Length . .: 490                 % Compression . .: -1
Creator . . . . .: SYSIBM              Creation Tbl Spac: SYSDBASE
Creation Database: DSNDB06             Base Table . . . .
Creation RBA. . .: 000000000000        Cardinality . . .: -1
Alter RBA . . . .: 00000145F4A1        Pages with rows .: -1
Check Flag. . . .:                     Check RID . . . .:
Data Capture. . .:                     Auditing. . . . .:
Created by  . . .: SYSIBM              Create Date . . .: 1985-04-01-00.00.00
Alter Date stamp.: 2000-11-29-15.50.25 Runstats Date . .: 0001-01-01-00.00.00
                                                                     04/015
```

Figure 11-14 DB2 Table Information

Here is the relevant information about the main fields:

Table Name The name of the table that is being accessed by the target SQL statement. The name of the base table appears below in Base Table if Table Name is an alias.

Plan number Explained previously.

Mix OP Seq Explained previously.

Table number Explained previously.

Table Type The object type, as follows:

- **A** Alias
- **T** Table
- **V** View

Table Status Table status, as follows:

- **I** Incomplete. The definition of the table is incomplete because it lacks a primary index.
- **X** The table has a primary index.
- **blank** The table has no primary key, is a catalog table, or its type is A (Alias) or V (View).

Note: Remaining fields on this panel are taken from the SYSIBM.SYSTABLES catalog table. Refer to the SQL Reference for the version of DB2 that you are using for a description of these fields.

11.2.9 SQL End Completion Statistics display

Figure 11-15 depicts the **SQL End Completion Statistics** display. This display has a line item for each execution of each statement. You get this display by cursor-selecting an item in the Count column on the **SQL Analysis** display.

```
FBIP1C00 IBM AM for z/OS - SQL End Completion Statistics ---------------------
         2004/10/28 19:59:33 to 2004/10/28 20:00:40        Row 1 of 23
History Data Set: ERIC.AM.FBIRUN2.D2004302.T1959057
    Jobname: FBIRUN2          Stepname: DB2RR              System: ADT1
    LU Name: A6P2T101         DB2 SSID: DB0A               Module: FBIDBTBD
  Plan Name: FBIDBTBD         Auth Id.: ERIC             Oper Id.: ERIC
Stmt  Stmt           . - - - - - - - - -  R O W S - - - - - - - - - - .  Pages
Nbr.  Type     SQLRC Processed Examned Qualifyd Insertd Updated Deleted Scannd
  160 SELECT    -811    12,078   8,712      396       0       0       0    594
  245 FETCH      100         0       0        0       0       0       0      0
  350 FETCH      100         0       0        0       0       0       0      0
  360 FETCH        0       792     396      594     198       0       0  1,782
  462 FETCH        0    50,038  30,460   36,376  20,196       0       0  8,293
  472 FETCH      100         0       0        0       0       0       0      0
  534 SELECT       0    38,280  19,140   19,140       0       0       0    594
  502 CLOSE        0         0       0        0       0       0       0      0
  472 FETCH        0    17,551  10,058   17,551       0       0       0  6,897
  446 OPEN         0 1,064,672  37,044   19,992  19,992       0       0 69,580
  430 PREPARE      0     1,552     776      776       0       0       0  2,328
  397 CLOSE        0         0       0        0       0       0       0      0
  394 CLOSE        0         0       0        0       0       0       0      0
  350 FETCH        0       792     396      594     198       0       0  1,782
  334 OPEN         0 1,070,104  37,233      197     197       0       0 69,738
Command ===> _____________________________________________ Scroll ==> CSR
                                                                     24/015
```

Figure 11-15 SQL End Completion Statistics

Here is the relevant information about the main fields:

Stmt Nbr A number that identifies the statement being explained. If the EXPLAIN statement is embedded in the application program or if the EXPLAIN(YES) option is specified when the package or plan is bound or rebound, the statement number displayed in this field is the one that is assigned by the DB2 precompiler and placed in the target DBRM.

Stmt Type The type of SQL statement.

SQLRC The SQL return code.

Rows Processed The number of rows that were processed for statement execution.

Rows Examined The number of rows that were examined for statement execution.

Rows Qualified The number of rows that qualified for statement execution.

Rows Inserted The number of rows inserted into the target table as a result of this statement.

Rows Updated The number of rows updated in the target table as a result of this statement.

Rows Deleted The number of rows deleted from the target table as a result of this statement.

Pages Scanned The number of pages requested through a get page operation.

Cursor-select the SQLRC -811 and press Enter to drill down to the **SQL Statement Completion** display shown in the next figure.

11.2.10 SQL Statement Completion display

Figure 11-16 is the SQL Statement Completion display, which shows the completion data about SQL statement execution that is returned to the program in the SQL Communication Area (SQLCA). You reach this panel by cursor-selecting an entry in the SQLRC column on either the **SQL End Completion Statistics** display or the **SQL Analysis** display.

There are two versions of this display:

1. If the target application calls program DSNTIAR to interpret the SQL return code issued after the target SQL statement has executed, DSNTxxxx messages are given along with the contents of the SQLCA data items (SQLCODE, SQLSTATE, SQLERRD, and so on).
2. If the target application does not call program DSNTIAR to interpret the SQL return code issued after the target SQL statement has executed, DSNTxxxs messages are not displayed.

```
FBIP1D00 IBM AM for z/OS - SQL Statement Completion --------------------------
         2004/10/28 19:59:33 to 2004/10/28 20:00:40       Row 1 of 10
History Data Set: ERIC.AM.FBIRUN2.D2004302.T1959057
    Jobname: FBIRUN2          Stepname: DB2RR              System: ADT1
    LU Name: A6P2T101         DB2 SSID: DB0A               Module: FBIDBTBD
  Plan Name: FBIDBTBD         Auth Id.: ERIC             Oper Id.: ERIC
  Statement: 160                  Type: SELECT             SQLRC: -811
Message  <------------------------- S Q L C A --------------------------->
DSNT408I SQLCODE = -811, ERROR:  THE RESULT OF AN EMBEDDED SELECT
         STATEMENT OR A SUBSELECT IN THE SET CLAUSE OF AN UPDATE
         STATEMENT IS A TABLE OF MORE THAN ONE ROW, OR THE RESULT OF A
         SUBQUERY OF A BASIC PREDICATE IS MORE THAN ONE VALUE
DSNT418I SQLSTATE   = 21000 SQLSTATE RETURN CODE
DSNT415I SQLERRP    = DSNXREMS SQL PROCEDURE DETECTING ERROR
DSNT416I SQLERRD    = -140  0  0  -1  0  0 SQL DIAGNOSTIC INFORMATION
DSNT416I SQLERRD    = X'FFFFFF74'  X'00000000'  X'00000000'  X'FFFFFFFF'
         X'00000000'  X'00000000' SQL DIAGNOSTIC INFORMATION

****************************** BOTTOM OF DATA ******************************



Command ===> _____________________________________________ Scroll ==> CSR
                                                                   24/015
```

Figure 11-16 SQL Statement Completion

Here is the relevant information about the main fields:

SQLRC The SQL return code.

Message and SQLCA If the application program has called DSNTIAR to interpret the results of statement execution, the Message field contains a series of DSNTxxxx message numbers and the SQLCA field contains the associated message text.

If the application program has not called DSNTIAR, the Message field contains a list of SQLCA data items and the SQLCA field contains the values of those data items.

> **Note:** Refer to the SQL Reference for the version of DB2 that you are using for a detailed description of SQLCA items.

11.3 IMS analysis

Application Monitor monitors IMS applications and collects data that indicates where IMS tasks are spending time. Application Monitor only monitors IMS dependent regions. An IMS dependent region is defined as one where a job step executes an IMS program. These regions include batch, Batch Message Processing (BMP), Message Processing Program (MPP), and Fast Path (FP) regions.

If a batch job calls IMS by attaching either DL/I or a BMP, Application Monitor monitors such a job as an MVS batch job and not as an IMS application.

11.3.1 IMS customization

No special customization has to be done to access the following displays:

- IMS Messages Processed by Region
- IMS Message Analysis
- IMS Message Call Analysis
- IMS Segment Search Criteria Analysis

The following sections describe these displays in detail.

11.3.2 IMS messages processed by region display

Figure 11-17 is displayed if you select an item in the IMS column for a specific System, Job, and Step on the Active Monitor Sessions panel, or if you select an IMS region on the Select an Active Job to Analyze panel.

```
FBIP1000 IBM AM for z/OS - Select an Active Job to Analyze -------------------
         2004/10/29 01:09:10 to 2004/10/29 01:09:10       Row 1 of 5

System   Name     Typ C I Stepname Procstep Asid   CPU%           I/O   Real
-------- -------- --- - - -------- -------- ---- ------       ------- ------
*        *        *   * Y *        *
ADT1     IMS8F11  BAT   Y IMS8F11  IFP      0027   0.00             0    302
ADT1     IMS8F12  BAT   Y IMS8F12  IFP      0028   0.00             0    302
ADT1     IMS8F13  BAT   Y IMS8F13  IFP      0029   0.00             0    304
ADT1     IMS8M11  BAT   Y IMS8M11  REGION   0026   0.00             0    597
ADT1     IMS8M12  BAT   Y IMS8M11  REGION   002C   0.00             0   1951
****************************** BOTTOM OF DATA ********************************



Command ===> _____________________________________________ Scroll ==> CSR
                                                                   24/015
```

Figure 11-17 Select an Active Job to Analyze

Cursor-select IMS8M11 and press Enter; Application Monitor displays the panel shown in Figure 11-18.

```
FBIP1A00 IBM AM for z/OS - Active Monitor Sessions ---------------------------
         2004/10/29 01:24:01 to 2004/10/29 01:24:04       Row 1 of 1

  Named Set: N/A
System   Jobname  Stepname Pgm  CICS IMS  DB2  Dly  Dsn  CPU%   Sample Duration
-------- -------- -------- Rpts Rpts Rpts Rpts Rpts Rpts ------  Count --------
ADT1     IMS8M11  IMS8M11     0      YES                    0.00      0 00:00:02
******************************** BOTTOM OF DATA ********************************

Command ===> _____________________________________________ Scroll ==> CSR
                                                                      24/015
```

Figure 11-18 Active Monitor Sessions

Cursor-select the Y value in the IMS Rpts column and press Enter; Application Monitor returns the display shown in Figure 11-19.

Note: If your initial monitoring selection specified a unique IMS message ID, the displayed data is limited to that message. Otherwise, all messages executing in the selected region will be displayed.

```
FBIP1C00 IBM AM for z/OS - IMS Messages Processed By Region ------------------
         2004/10/28 23:52:15 to 2004/10/28 23:58:37       Row 1 of 3

     System: ADT1            Jobname: IMS8M11         Messages: 3
    CPU Pct: 0.00         Active Time: 1.241357         Samples: 252

               Assoc.
Message  Count Samples <----- % Time in IMS -----> <----- % Time Ready ------>
-------- ----- -------
 .I3TCB      1       0   0.00                          0.00
 .I1TCB      2       7  89.06 ================>     100.00 ===================>
IVTNO        3       0  70.22 =============>          0.00
******************************** BOTTOM OF DATA ********************************

Command ===> _____________________________________________ Scroll ==> CSR
                                                                      24/015
```

Figure 11-19 IMS Messages Processed By Region

Here is the relevant information about the main fields:

Jobname The name of the target IMS batch job or region.

Messages The number of active messages in the region during this monitoring interval.

If you selected a batch IMS job to analyze, the number 1 is displayed in this field.

CPU Pct The percentage of total available CPU being used by all active messages in the region during the monitoring interval. This value is an average of the CPU Pct values collected from the samples taken during the displayed monitoring interval.

Message This column lists the message IDs that are or have been active during the displayed monitoring interval.

N/A (not available) is displayed in this field if:
- An IMS batch (non-message-processing) program is executing.
- A BMP or MPP program is executing and Application Monitor misses the initial GU call (to retrieve the message ID) when it takes a sample.

Count The number of instances of the associated message that have been active during the monitoring interval.

If you selected a batch IMS job to analyze, the number 1 is displayed in this field.

Assoc Samples The number of samples that Application Monitor has taken during the monitoring interval that included information about this message or batch job when the program was in application code. Assoc Samples is not incremented if the batch job or program that is processing the message is in IMS code at the time the sample is taken. Thus, Assoc Samples may be less than Samples.

%Time in IMS The percentage of Active Time that the target batch job or all instances of the message were executing and waiting for IMS to respond to a request. The percentage is computed as the time spent in IMS divided by the total time that the target batch job or the message instances took to execute.

The ===> symbol to the right of the percentage is a graphical representation of % Time in IMS. This enables you to easily spot the messages that are spending the most time in IMS.

%Time Ready The total number of samples taken divided into the number of samples that indicate that the target batch job or message was ready to run.

Note: An IMS batch job or message that is not in IMS code 80+ percent of the time, should be READY to run 100 percent of the time. Otherwise, it is waiting on other system events.
If the application is in IMS processing most of the time, the application will be waiting and not READY to run.

The ===> symbol to the right of the percentage is a graphical representation of % Time Ready. This graph enables you to easily spot the messages that are spending the most time being ready to run.

Cursor-select the IVTNO message and press Enter to drill down to the **IMS Message Analysis** display.

11.3.3 IMS Message Analysis display

The fixed area of the display in Figure 11-20 summarizes all instances of the target message (or Jobname if you are analyzing an IMS batch (non-message-processing) job).

The scrollable area of the display summarizes, as one entry, all calls with the same operands from the same location in the application. The data on the scrollable area of the panel is ordered by Program.

```
FBIP1D00 IBM AM for z/OS - IMS Message Analysis ------------------------------
         2004/10/28 23:52:37 to 2004/10/28 23:55:48       Row 1 of 6

     System: ADT1              Jobname: IMS8M11          Sub Sys: IMS1
    Message: IVTNO            Messages: 3                   PSB: DFSIVP1
    CPU Pct: 0.00          Active Time: 0.001573

Program  PCB DBD      Segment  Call  Proc   % Total  % Total
Name     Nbr Name     Name     Type  OPTS  App Time IMS Time Calls  SSC
-------- --- -------- -------- ----- ----- -------- -------- ----- ----
DFSSBSP0   3 IVPDB1   RC=GE    STAT  A        25.53     0.82     1    1
DFSSBSP0   3 IVPDB1            STAT  A        62.30     3.84     2    1
DFSIVP1    1                   GU             0.00     0.00     1    0
DFSIVP1    1 ERIC              ISRT           9.45     5.42     2    0
DFSIVP1    3 IVPDB1   A1111111 GU    A        2.70    89.91     2    1
DFSIVP1    1 ERIC              GU             0.00     0.00     2    0
******************************** BOTTOM OF DATA ********************************




Command ===> _____________________________________________ Scroll ==> CSR
                                                                     24/015
```

Figure 11-20 IMS Message Analysis

Here is the relevant information about the main fields:

Message The ID of the message you selected to analyze.

N/A is displayed in this field if:

- An IMS batch (non-message-processing) program is executing.
- A BMP or MPP program is executing and Application Monitor misses the initial GU call (to retrieve the message Id) when it takes a sample.

If you selected a batch IMS job to analyze, the number 1 is displayed in this field.

Messages The number of instances of the target message that are or have been running during this monitoring interval.

PSB The name of the Program Specification Block.

Program Name The name of the program that is issued the associated calls.

PCB Nbr Number of the Program Communication Block within the PSB where IMS returns the results of the associated call to the application program.

DBD Name Can be one of the following:

- The name of the Database Descriptor associated with the database if this call is accessing a database.
- The device (for example, dynamic message terminal) name for calls handled through non-database PCBs.

Segment Name The name of the database segment that is accessed with this call. If IMS was unable to access the target segment successfully, the return code is displayed in this field (RC=xx).

This field is blank if the associated call is not accessing a database.

Call Type An abbreviation of the type of call (GU, GN, ISRT, and so on).

Proc OPTS Processing options used in this call.

% Total App Time The relative percentage of wall-clock time spent in the application code from the previous IMS call to the beginning of this IMS call. The total wall-clock time for all calls is used to compute this percentage. The values in this column should total approximately 100 percent. If the time spent in application code is not significant, the value will be zero.

% Total IMS Time The relative percentage of wall-clock time used to execute this call for IMS services. The total wall-clock time for all calls is used to compute this percentage. The values in this column should total approximately 100 percent. If the time used is not significant, the value will be zero.

Calls The number of times, during the displayed monitoring interval, that the associated program that is identified in the Program Name column has issued this call on behalf of the target job or all instances of the target message.

SSC The Segment Search Criteria. This field shows the number of arguments the programmer coded for this call. If the same call is made from the same address in the program with a varying number of SSC arguments, each call is listed on a separate line.

How to use this display

- Cursor-select a value in the Program Name column to drill down to the **CPU and Loadmod Analysis** display for this program.
- Cursor-select a value in the Calls column and press Enter to drill down to the **IMS Message Call Analysis** display.
- Cursor-select a value in the SSC column and press Enter to drill down to the **IMS SSC Analysis** display.

Cursor-select a value in the Calls column and press Enter; Application Monitor returns the IMS Message Call Analysis display described in the next section.

11.3.4 IMS Message Call Analysis display

Figure 11-21 depicts the **IMS Message Call Analysis** display. The display shows all calls made from the target program during the displayed monitoring interval.

```
FBIP1D00 IBM AM for z/OS - IMS Message Call Analysis -------------------------
         2004/10/28 23:52:37 to 2004/10/28 23:55:48       Row 1 of 4          >

     System: ADT1             Jobname: IMS8M11          Message: IVTNO
    Program: DFSIVP1          Sub Sys: IMS1            Messages: 3
        PSB: DFSIVP1          CPU Pct: 0.00         Active Time: 0.001101

PROGRAM  OFFSET DBDNAME  SEGNAME  TYPE   Appl. Active      IMS Active    %CPU #
-------- ------ -------- -------- ---- --------------- --------------- ------ -
DFSIVP1  000076                   GU          0.000000        0.000000   0.00
DFSIVP1  0005D2 ERIC              ISRT        0.000037        0.000059   0.00
DFSIVP1  000346 IVPDB1   A1111111 GU          0.000010        0.000993   0.00
DFSIVP1  000076 ERIC              GU          0.000000        0.000000   0.00
******************************** BOTTOM OF DATA ********************************

Command ===> _____________________________________________ Scroll ==> CSR
                                                                      24/015
```

Figure 11-21 IMS Message Call Analysis

The **>** character on the second line indicates that you can press PF11 to scroll to the left to view more data on the right.

Figure 11-22 shows the IMS Message Call Analysis display shifted to the left so that you can view the #IN-RDY and CALLS columns on the right.

```
FBIP1D00 IBM AM for z/OS - IMS Message Call Analysis -------------------------
<        2004/10/28 23:52:37 to 2004/10/28 23:55:48       Row 1 of 4

     System: ADT1             Jobname: IMS8M11          Message: IVTNO
    Program: DFSIVP1          Sub Sys: IMS1            Messages: 3
        PSB: DFSIVP1          CPU Pct: 0.00         Active Time: 0.001101

ET DBDNAME  SEGNAME  TYPE   Appl. Active      IMS Active    %CPU #IN-RDY  CALLS
-- -------- -------- ---- --------------- --------------- ------ ------- ------
76                   GU          0.000000        0.000000   0.00       0      1
D2 ERIC              ISRT        0.000037        0.000059   0.00       0      2
46 IVPDB1   A1111111 GU          0.000010        0.000993   0.00       0      2
76 ERIC              GU          0.000000        0.000000   0.00       0      2
******************************** BOTTOM OF DATA ********************

Command ===> _____________________________________________ Scroll ==> CSR
                                                                      24/015
```

Figure 11-22 IMS Message Call Analysis

Here is the relevant information about the main fields:

Program The program module that issued the call or calls identified in this line item.

Offset The offset within the program module where the call instruction is located.

DBDNAME Can be one of the following:

- The name of the Database Descriptor associated with the database if this call is accessing a database.
- The device (for example, dynamic message terminal) name for calls handled through non-database PCBs.

SEGNAME The name of the database segment that is accessed with this call. If IMS was unable to access the target segment successfully, the return code is displayed in this field (RC=xx).

TYPE An abbreviation of the type of call (GU, GN, ISRT, and so on).

Appl Active The total wall-clock time (to millionths of a second) that the application was in application code during the displayed monitoring interval.

IMS Active The total wall-clock time (to millionths of a second) that the application was in IMS code during the displayed monitoring interval.

%CPU The percentage of CPU Pct that has been used to process the calls issued from this location in the program during the displayed monitoring interval. Zero is displayed if the time is negligible.

#IN-RDY The number of samples in which the calls at this program location were ready to execute.

CALLS The number of calls to access the associated SEGNAME that have been issued from this program location during the displayed monitoring interval.

11.3.5 IMS segment search criteria analysis display

Figure 11-23 depicts the **IMS Segment Search Criteria (SSC) Analysis** display. This display appears if you cursor-select an entry in the SSC column on the IMS Message Analysis display. This display shows the segment, field, and operation performed for the target call. The field values used in the calls are not displayed.

```
FBIP1H01 IBM AM for z/OS - IMS SSC Analysis ----------------------------------
         2004/10/28 23:52:15 to 2004/10/29 01:01:00       Row 1 of 1

     System: ADT1            Jobname: IMS8M11          Sub Sys: IMS1
    Program: DFSIVP1         Message: IVTNO           Messages: 3
        PSB: DFSIVP1         CPU Pct: 0.00         Active Time: 0.001004
       PCB#: 3               DB Name: IVPDB1           Segment: A1111111
   Proc Opt: A              Call Typ: GU

CallType Search   Segment  Field    Operation
-------  -------- -------- -------- ---------
GU              1 A1111111 A1111111  =
******************************** BOTTOM OF DATA ********************************



Command ===> _____________________________________________ Scroll ==> CSR_
                                                                     24/015
```

Figure 11-23 IMS SSC Analysis

Here is the relevant information about the main fields:

PSB	The name of the Program Specification Block.
PCB	The number of the PCB within the PSB used to return the results of the call to the program.
DB Name	Can be one of the following: – The name of the Database Descriptor associated with the database if this call is accessing a database. – The device (for example, dynamic message terminal) name for calls handled through non-database PCBs.
Proc Opt	Processing options used in this call.
Call Typ	An abbreviation of the type of call (GU, GN, ISRT, and so on).
Search	An index to each component of the Segment Search Argument (SSA) used in the call.
Segment	The database segment that is accessed in the call. If the call is unsuccessful, the return code is displayed in this field (RC=xx).
Field	The segment field that is named in the call.
Operation	The operator used to locate a match for the field value. The field value is not given in the SSC analysis.

12

Application Monitor and batch reports

In many situations it is more useful to have actual written reports than to use the panels. In our work experience we have found it easiest to compare the effects of small code changes by creating before-and-after physical reports and doing a page-by-page comparison.

This chapter explains how to use the batch facility to generate reports.

12.1 Overview

On some occasions you just need to have data in your hands and see it on paper. Application Monitor's online analysis facilities and reports can provide you with a large amount of data to view. Some developers prefer to have that data on paper so that they can write their comments beside the data.

Application Monitor's batch facility provides a way to get those hard-copy reports. Any report generated will be for the whole time period in the analysis file (it is not possible to do interval reporting).

12.2 JCL - FBIBATCH

You can find the JCL for the Application Monitor batch reporter in hlq.SFBIJCL(FBIBATCH). Example 12-1 is a listing of the JCL.

Example 12-1 FBIBATCH

```
//FBIBATCH JOB <job parameters>
//*
//*-------------------------------------------------------------------*
//* This job prints Application Monitor application analysis reports *
//*-------------------------------------------------------------------*
//*
//*  CAUTION: This is neither a JCL procedure nor a complete job.
//*  Before using this job, you will have to make the following
//*  modifications:
//*
//*  1) Add the job parameters to meet your system requirements.
//*  2) Change all occurrences of "hlq" to the high-level qualifier
//*     of your Application Monitor libraries.
//*  3) Examine the ISPF DDnames ISPMLIB, ISPTLIB, and ISPSLIB; and
//*     make sure they reflect your production libraries.
//*
//*     Change "ISP.SISPTENU" to your "ISPF PRODUCT TABLES"
//*     Change "ISP.SISPMENU" to your "ISPF PRODUCT MESSAGES"
//*     Change "ISP.SISPSENU" to your "ISPF PRODUCT SKELETONS"
//*
//*  4) Examine the //STEPLIB and //ISPLLIB statements below and
//*     comment out ISP.SISPLPA and ISP.SISPLOAD if these 2 libraries
//*     are already allocated in your TSO logon procedure.
//*  5) Remove DSN610.SDSNLOAD from your //STEPLIB statement
//*     if it already resides in the LPA or the LNKLST.
//*     Modify the high-level qualifier DSN610 as appropriate.
//*     Having this DB2 load library available ensures that the
//*     "SQL Statement Completion" report will be formatted properly.
//*
//*  6) Change "userID" to a unique TSO user ID that has been logged
//*     on at least once in the past.
//*  7) Verify that the SUBSYS parameter on the ISPSTART statement
//*     correctly identifies the 4-character Application Monitor
//*     subsystem ID, which is specified in the started task
//*     FBISTART.
//*     If FBISTART uses a different subsystem value than
//*     FBIO, update the SUBSYS parameter in this job accordingly.
//*  8) Refer to the User's Guide if you want to override any
//*     defaults by coding the parameters on the ISPSTART command.
//*
```

```
//*  You need to do Steps 1 through 8 above only once.
//*  Do Steps 9 through 11 below each time before you submit this job.
//*
//*  9)  Change "historyDSN" to the name of the historical data set
//*      whose data you want to analyze.
//*  10) if you want to see reports of your COBOL statements,
//*      change "cobolDSN" to the name of the COBOL listings data set
//*      and uncomment the //FBICOBOL DD statement.
//*  11) If you want to see reports of your PL/I statements,
//*      change "pliDSN" to the name of the PL/I listings data set
//*      and uncomment the //FBIPLI DD statement.
//*
//*  IMPORTANT NOTE:
//*
//*  1) Ensure that the IBM Application Monitor started task
//*     (FBISTART) is running before you submit this job.
//*
//*********************************************************************
//*
//FBISTEP  EXEC  PGM=IKJEFT01,REGION=0M,DYNAMNBR=10
//STEPLIB  DD  DISP=SHR,DSN=hlq.SFBIAUTH
//         DD  DISP=SHR,DSN=ISP.SISPLPA          <== Verify this
//         DD  DISP=SHR,DSN=ISP.SISPLOAD         <== Verify this
//         DD  DISP=SHR,DSN=DSN610.SDSNLOAD      <== Verify this
//*
//*
//*-------------------------------------------------------------------*
//* Specify your history data set on the //FBIHIST DD                *
//*-------------------------------------------------------------------*
//FBIHIST  DD DISP=SHR,DSN=historyDSN
//*
//*-------------------------------------------------------------------*
//* The //FBIPRINT data set is RECFM=FBA, LRECL=133                  *
//*-------------------------------------------------------------------*
//FBIPRINT DD SYSOUT=*
//*
//*-------------------------------------------------------------------*
//* Use the //SYSTSIN DD statement to override processing parameters  *
//*-------------------------------------------------------------------*
//SYSTSIN  DD  *
PROFILE PREFIX(userID)
ISPSTART NEWAPPL(FBI) CMD(FBILPM00 PAGELINES=64,TRACE=80,+
SUBSYS=FBI0,+
MAXLINKS=20,MAXREPORTS=500) +
BDISPMAX(10000)
/*
//*
//*-------------------------------------------------------------------*
//* Optionally, specify your COBOL listings data set name below.      *
//* Uncomment the statement.                                          *
//*-------------------------------------------------------------------*
//*FBICOBOL DD DISP=SHR,DSN=cobolDSN
//*
//*-------------------------------------------------------------------*
//* Optionally, specify your PL/I listings data set name below.       *
//* Uncomment the statement.                                          *
//*-------------------------------------------------------------------*
//*FBIPLI   DD DISP=SHR,DSN=pliDSN
//*
//*
```

```
//SYSTSPRT DD SYSOUT=*
//SYSTERM  DD SYSOUT=*
//ISPMLIB  DD  DISP=SHR,DSN=hlq.SFBIMENU
//         DD  DISP=SHR,DSN=ISP.SISPMENU          <== Verify this
//ISPLLIB  DD  DISP=SHR,DSN=hlq.SFBIAUTH
//         DD  DISP=SHR,DSN=ISP.SISPLPA           <== Verify this
//         DD  DISP=SHR,DSN=ISP.SISPLOAD          <== Verify this
//ISPPLIB  DD  DISP=SHR,DSN=hlq.SFBIPENU
//ISPTLIB  DD  DISP=SHR,DSN=ISP.SISPTENU          <== Verify this
//ISPPROF  DD  DISP=(NEW,DELETE),DSN=&&TEMPPROF,
//         SPACE=(TRK,(1,1,1)),UNIT=SYSALLDA,
//         RECFM=FB,LRECL=80,BLKSIZE=6400
//ISPTABL  DD  DISP=(NEW,DELETE),DSN=&&TEMPTABL,
//         SPACE=(TRK,(1,1,1)),UNIT=SYSALLDA,
//         RECFM=FB,LRECL=80,BLKSIZE=6400
//ISPSLIB  DD  DISP=SHR,DSN=ISP.SISPSENU          <== Verify this
//*
//ISPLOG   DD  SYSOUT=*,
//         DCB=(RECFM=VA,LRECL=125,BLKSIZE=129)
//ISPLST0  DD  UNIT=SYSALLDA,
//         SPACE=(TRK,(5,5)),
//         DCB=(RECFM=FBA,LRECL=121,BLKSIZE=1210),
//         DISP=(,DELETE,DELETE)
//ISPLST1  DD  UNIT=SYSALLDA,
//         SPACE=(TRK,(5,5)),
//         DCB=(RECFM=FBA,LRECL=121,BLKSIZE=1210),
//         DISP=(,DELETE,DELETE)
//ISPLST2  DD  UNIT=SYSALLDA,
//         SPACE=(TRK,(5,5)),
//         DCB=(RECFM=FBA,LRECL=121,BLKSIZE=1210),
//         DISP=(,DELETE,DELETE)
//
```

Follow these steps to tailor the JCL to run the Batch Reporter:

1. Replace the JOB statement with a JOB statement that is valid for your installation.
2. Replace all occurrences of hlq with the high-level qualifier of IBM Application Monitor libraries as shown in Table 12-1.

Table 12-1 AM- Libraries

DDname	Data Set Name	Description
ISPMLIB	hlq.SFBIMENU	AM message library
ISPLLIB	hlq.SFBIAUTH	Main program library for the AM started task
ISPPLIB	hlq.SFBIPENU	AM panel library

3. Change the name of the ISPF production libraries, as shown in Table 12-2, to the names used for your installation, if necessary.

Table 12-2 ISPF Libraries

DDname	Data Set Name	Description
ISPMLIB	ISP.SISPMENU	ISPF messages library
ISPTLIB	ISP.SISPTENU	ISPF table library
ISPSLIB	ISP.SISPSENU	ISPF skeleton library

4. Specify the TSO userid for the PREFIX parameter of the inline PROFILE statement. This userid should have been logged onto at least once in the past.
5. Change the following parameters of the inline ISPSTART statement as appropriate:

   MAXLINKS Maximum number of hyperlinks to follow from any one column. This parameter limits the number of reports that are printed for any one report layout.

   You can specify a value in the range 1 to 99999.

   For example, if MAXLINKS=3 and the target report layout has two cursor-selectable columns with 25 rows of data, the batch reporter will print the reports for the top three rows (3 links times 2 cursor-selectable columns in each row = 6 reports).

   For this example, the maximum number of reports available is 50 (25 rows times 2 selectable columns in each row = 50 reports). If you wanted to print all available reports for this example, you would specify MAXLINKS=25 (25 links times 2 cursor-selectable columns in each row = 50 reports).

   The default MAXLINKS value is 20.

   MAXREPORTS Maximum number of report layouts to be printed. Use this parameter to limit the total number of report layouts in the tree of reports that are printed.

   You can specify a value in the range 1 to 99999.

   The default MAXREPORTS value is 20.

   PAGELINES Maximum number of lines to be printed per report page.

   You can specify a value in the range 1 to 99.

   The default PAGELINES value is 64.

   BDISPMAX ISPF uses this parameter to limit the number of panels to be displayed in the batch reports.

   You can specify a value from 1 to 10000.

   The ISPF default is 100.

   SUBSYS Verify that the name of the Application Monitor subsystem is correct for your installation. FBI0 is the default subsystem ID.
6. Specify the name of the target historical data set for the DSN parameter of the FBIHIST DD statement.
7. If you want to include statement analysis for a COBOL program in the batch report, uncomment the FBICOBOL DD statement and change the DSN= parameter to identify the compiler listings data set for COBOL. The listings data set must be a PDS or PDSE. Do not specify a member name. Application Monitor locates the member needed for statement analysis.
8. If you want to include statement analysis for a PL/I program in the batch report, uncomment the FBIPLI DD statement and change the DSN= parameter to identify the compiler listings data set for PL/I. The listings data set must be a PDS or PDSE. Do not specify a member name. Application Monitor locates the member needed for statement analysis.

Notes:

1. The SYSTSIN DD statement invokes FBILPM00 with the specified processing parameters.
2. The ISPPROF and ISPTABL DD statements identify temporary ISPF profile and table libraries that this job requires.
3. All messages that this job issues while it executes are listed in the JES job log.
4. If ISPF is not installed in the LNKLST or LPA, or if you require a specific release of ISPF, add the appropriate ISPF load library to the STEPLIB DD statement.
5. If you are analyzing DB2 data and the load library for the target DB2 subsystem is not installed in the LNKLST, add the appropriate DB2 load library to the STEPLIB DD statement.

12.3 Printed report layouts

The batch reporter writes the printed reports to the FBIPRINT data set. The batch reporter opens this data set with the following attributes:

- RECFM=FBA
- LRECL=133

The output listing has the following pages:

- Title Page
- Jobs, Steps, and Their Time Ranges page
- Historical Reports Available page
- The detailed analysis reports that are available in the historical data set

12.3.1 The title page

The first report page that is printed is a title page as shown in Example 12-2.

Example 12-2 Title page

```
HISTORY DATA SET: CHABERT.AM.COBOLRUN.D2004303.T1539563
    TIME SPANNED: 2004/10/29 16:13:19 TO 2004/10/29 16:13:47
         PRINTED: 2004/10/29 16:21:49 FOR CHABERT
```

The title page identifies:

- The name of the target historical data set.
- The time spanned by the reports
- Date and time the report was printed.
- The TSO userid associated with the job that printed the report.

12.3.2 Jobs, Steps, and Time Ranges page

Example 12-3 is a display of the jobs, steps, and time range page.

Example 12-3 Jobs, Steps, and Time Range page

```
IBM AM for z/OS - Jobs, Steps, and Their Time Ranges
IBM AM for z/OS - Jobs, Steps, and Their Time Ranges
History File: CHABERT.AM.COBOLRUN.D2004303.T1539563
```

```
  Start Time: 2004/10/29
              16:13:19
   Stop Time: 2004/10/29
              16:13:47
Begin    End      System   Job      Step     Pgm  CICS IMS  DB2  Dly  Dsn
-------- -------- -------- -------- -------- Rpts Rpts Rpts Rpts Rpts Rpts
16:13:19 16:13:47 STLABF6  COBOLRUN STEP01   YES                 YES  YES
```

Fields on this report page are:

History File The name of the historical data set that contains the report data.

Start Time The date and time that Application Monitor started collecting the data in the target historical data set, in the format yyyy/mm/dd hh:mm:ss.

Stop Time The date and time that Application Monitor stopped collecting the data in the target historical data set, in the format yyyy/mm/dd hh:mm:ss.

The columns on this report page contain the following data:

Begin The time that Application Monitor started to monitor the target job step or transaction.

End The time that Application Monitor stopped monitoring the target job step or transaction.

System The system on which the target job step or transaction executed.

Job The name of the job, as follows:

- For a user application submitted as a batch job: The job name
- For a started task: The name of the started task
- For a TSO user session: The TSO userid of the user
- For CICS: The name of the CICS subsystem
- For IMS: The name of the IMS dependent region

Step The name of the job step, as follows:

- For a user application submitted as a batch job: The name of the job step for which the analysis data was collected
- For a started task: The name of the job step for which the analysis data was collected
- For a TSO user session: The name of the logon procedure step
- For CICS: The name of the job step that is controlling the activity in the CICS subsystem
- For IMS: The name of the job step that is controlling the activity in the IMS dependent region

Pgm Rpts, CICS Rpts, IMS Rpts, DB2 Rpts, Dly Rpts, Dsn Rpts
These columns identify the type of analysis data that is available for the target line item, as follows:

- YES: This type of analysis is available.
- blank: This type of analysis is not available.

> **Note:** If you request that DB2 data always be collected as a default, YES appears in the DB2 Rpts column even if the target job does not access a DB2 database

12.3.3 Historical Reports available

The Historical Reports Available page, as shown in Example 12-4, follows the Jobs, Steps, and Their Time Ranges page. This page shows actual report data, including the ASA printer control characters in column 1. (The number 1 causes a new page to start. The + sign causes overtyping to highlight the text.)

Report content corresponds to the content of the Historical Reports Available display that would appear if you were viewing the display online.

Example 12-4 Historical Reports Available

```
1IBM AM for z/OS - Historical Reports Available
+IBM AM for z/OS - Historical Reports Available

   Interval Start: 2004/10/29
                   16:13:19
    Interval Stop: 2004/10/29
                   16:13:47
   Named Set: N/A

 System   Jobname  Stepname Pgm  CICS IMS  DB2  Dly  Dsn  CPU%   Sample
 -------- -------- -------- Rpts Rpts Rpts Rpts Rpts Rpts ------  Count
 STLABF6  COBOLRUN STEP01      9               YES  YES   23.49  9,470
```

12.3.4 Detailed Analysis reports

The next report pages follow the drill-down sequence of reports that would be available if you were viewing them online. Refer to the appropriate sections of this redbook for a description of report content:

- Program module analysis
- CICS analysis
- IMS analysis
- DB2 analysis
- Delay analysis
- Data set analysis

That is, the batch reporter prints a report in drill-down sequence, left to right, for each cursor-selectable field that is defined for the online report displays.

For the example given 12.3.2, “Jobs, Steps, and Time Ranges page” on page 186, the following reports are printed in the following sequence:

- CPU and Loadmod Analysis for the first load module. For each identified load module, the following reports are printed:
  - CSECT Utilization Analysis
  - Statement Analysis (if available)
  - Detail Statement Display (if available)
  - Instruction Analysis for each active CSECT (if Statement Analysis is not available)
  - Detail Instruction Display for each CSECT offset
- This sequence of reports is then repeated for each separate load module identified on the CPU and Loadmod Analysis report.
- Then, the CICS reports are printed in the following sequence:
  - CICS Transactions
  - CICS Calls in a Transaction

  - CICS Transaction Executions
  - CICS Calls in a Unit of Work
- Then, the IMS reports are printed in the following sequence:
  - IMS Messages Processed by Region
  - IMS Message Analysis
  - IMS Message Call Analysis
  - IMS Segment Search Criteria Analysis
- Then, the DB2 reports are printed in the following sequence:
  - SQL Analysis
  - SQL Statement Summary
  - SQL Statement View
  - SQL Statement Full Text
  - SQL Detailed Explain Analysis
  - SQL Explain Detail
  - SQL End Completion Statistics
  - SQL Statement Completion
  - DB2 Table Information
- Then, the Delay report is printed.
- Finally, the data set reports are printed in the following sequence for each data set the target job step accesses:
  - Data Set Analysis Report
  - VSAM Data Set Analysis Detail Report (for VSAM data sets only)
  - VSAM Local Shared Resources Report (for VSAM data sets only)

Part 4

Debug Tool and Debug Tool Utilities and Advanced Functions

In this part we describe how to customize and use Debug Tool and Debug Tool Utilities and Advanced Functions. It is divided into four chapters that deal with the following topics:

- How to debug programs using Debug Tool
- How to use Debug Tool under UNIX Systems Services
- Debug Tool Utilities and Advanced Functions
- Debug Tool and subsystems

13

Debugging programs using Debug Tool

This chapter describes how to debug programs using Debug Tool. It includes sections on debugging programs written in many languages, including:

- C/C++ programs
- PL/I programs
- OS/VS COBOL and COBOL for z/OS and OS/390
- Non-LE programs
- Assembled programs

13.1 Debugging C/C++ programs using Debug Tool

Before debugging a C/C++ program, the program must meet the following requirements:

- All the data sets required to debug your program comply with the guidelines described in this book. The source is in a single file and not a concatenation of files.
- All the libraries that your program needs are available.
- Your program is compiled with the appropriate compiler options. When a program is under development, you can get the full capability of Debug Tool by compiling your program with the TEST(ALL) compiler option.

13.1.1 Preparing the program

This section outlines the required program preparation steps.

C program preparation: Compiling with the TEST compiler option

Before you test your C program with Debug Tool, you must compile it with the C TEST compiler option:

- The suboptions BLOCK, LINE, and PATH regulate the points where the compiler inserts program hooks. When you set breakpoints, they are associated with the hooks that are used to instruct Debug Tool where to gain control of your program.
- The suboption SYM regulates the inclusion of symbol tables into the object output of the compiler. Debug Tool uses the symbol tables to obtain information about the variables in the program.

When you use the C TEST compiler option, be aware that:

- The C TEST compiler option generates hooks at entry and exit points for functions.
- The C TEST compiler option implicitly specifies the GONUMBER compiler option, which causes the compiler to generate line number tables that correspond to the input source file. You can explicitly remove this option by specifying NOGONUMBER. When the TEST and NOGONUMBER options are specified together, Debug Tool does not display the current execution line as you step through your code.
- Programs that are compiled with both the TEST compiler option and either the OPT(1) or OPT(2) compiler option do not have hooks at line, block, and path points, or generate a symbol table, regardless of the TEST suboptions specified. Only hooks for function entry and exit points are generated for optimized programs.
- You can specify any number of TEST suboptions, including conflicting suboptions (for example, both PATH and NOPATH). The last suboptions that are specified take effect. For example, if you specify TEST(BLOCK, NOBLOCK, BLOCK, NOLINE, LINE), what takes effect is TEST(BLOCK, LINE) because BLOCK and LINE are specified last.
- No duplicate hooks are generated even if two similar TEST suboptions are specified. For example, if you specify TEST(BLOCK, PATH), the BLOCK suboption causes the generation of hooks at entry and exit points. The PATH suboption also causes the generation of hooks at entry and exit points. However, only one hook is generated at each entry and exit point.

You can specify any combination of the C TEST suboptions in any order. The default suboptions are BLOCK, LINE, PATH, and SYM.

If all of the following tasks are completed, Debug Tool uses the hooks inserted by the Dynamic Debug facility instead of the hooks inserted by the compiler:

- The Dynamic Debug facility is installed on your system.
- You have not deactivated the Dynamic Debug facility by entering the SET DYNDEBUG OFF command.
- You compile your program with one of the following compilers:
  - C/C++ for MVS/ESA™ Version 3 (Program Number 5655-121)
  - C/C++ feature of OS/390 (Program Number 5647-A01)
  - C/C++ feature of z/OS (Program Number 5694-A01)

Using the hooks inserted by the Dynamic Debug facility provides better performance when you are debugging your program. However, some path breakpoints become unavailable. If you need to use those breakpoints, deactivate the Dynamic Debug facility by entering the SET DYNDEBUG OFF command.

If the Dynamic Debug facility is used to place hooks in programs that reside in read-only storage, the Authorized Debug facility must be installed and you must be authorized to use it. Contact your system administrator to get authorized.

If you build your application using the c89 or C++, perform the following steps:

1. Compile your source code as usual, but specify the –g option to generate debugging information. The –g option is equivalent to the TEST compiler option under TSO or MVS batch. For example, to compile the C source file fred.c from the u/mike/app directory, specify: `cd /u/mike/app c89 –g –o "//PROJ.LOAD(FRED)" fred.c`

   Note: The double quotes in the command line are required.

2. Set up your TSO environment, as described previously.
3. Debug the program under TSO by entering the following:

   ```
   FRED TEST ENVAR('PWD=/u/mike/app') / asis
   ```

   Note: The single quotes in the command line are required. ENVAR('PWD=/u/mike/app') sets the environment variable PWD to the path from where the source files were compiled. Debug Tool uses this information to determine from where it should read the source files

If you are debugging your application in the UNIX System Services Shell, you must debug in remote debug mode or in full-screen mode through a VTAM terminal.

The workstation component of remote debuggers is available through several products, including C and C++ Productivity Tools for OS/390 and VisualAge COBOL.

C++ program preparation: Compiling with the TEST compiler option

Before testing your C++ program with Debug Tool, you must compile it with the C++ TEST compiler option.

This causes the compiler to generate information about your program that Debug Tool needs to help you debug your program. If all of the following tasks are completed, Debug Tool uses

the hooks inserted by the Dynamic Debug facility instead of the hooks inserted by the compiler:

- The Dynamic Debug facility is installed on your system.
- You have not deactivated the Dynamic Debug facility by entering the SET DYNDEBUG OFF command.

You compile your program with one of the following compilers:

- C/C++ for MVS/ESA Version 3 (Program Number 5655-121)
- C/C++ feature of OS/390 (Program Number 5647-A01)
- C/C++ feature of z/OS (Program Number 5694-A01)

Using the hooks inserted by the Dynamic Debug facility provides better performance when you are debugging your program.

However, some path breakpoints become unavailable. If you need to use those breakpoints, deactivate the Dynamic Debug facility by entering the SET DYNDEBUG OFF command.

If the Dynamic Debug facility is used to place hooks in programs that reside in read-only storage, the Authorized Debug facility must be installed and you must be authorized to use it.

Contact your system administrator to get authorized.

A sample JCL listing for compiling a C program is shown in Figure 13-1.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------------
EDIT       GRACINE.TESTDT.JCL(CCNJIV1) - 01.08            Columns 00001 00072
Command ===> sub_                                          Scroll ===> CSR
****** ***************************** Top of Data ******************************
000001 //CCNJIV1  JOB 1,RACFUSER,TIME=1440,NOTIFY=&SYSUID,REGION=64M,
000002 //          CLASS=A,MSGCLASS=X,MSGLEVEL=(1,1)
000003 //ORDER JCLLIB ORDER=(CBC.SCCNPRC)
000004 //*-------------------------------------------------------------------
000005 //* Compile C with z1.2 compiler for use with Debug Tool
000006 //*-------------------------------------------------------------------
000007 //COMPILE EXEC EDCCB,
000008 // INFILE='CBC.SCCNSAM(CCNYIV1)',
000009 // OUTFILE='GRACINE.DEBUG.TOOL51.TESTLOAD(CIVP),DISP=SHR',
000010 // CPARM='OPTFILE(DD:OPTIONS)'
000011 //* GPARM='@@@@@@@ VERIFICATION SUCCESSFUL @@@@@@@'
000012 //COMPILE.OPTIONS DD *
000013 TEST
000014 /*
****** ***************************** Bottom of Data ***************************


 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 13-1 JCL compiling C program

C and C++ programs: Compiling on an HFS file system

If you are compiling and launching C or C++ programs on an HFS file system, you must do one of the following:

- Compile and launch the programs from the same location.
- Specify the full path name when you compile the programs.

By default, the C or C++ compiler stores the relative path and file names of the source files in the object file. When you start a debug session, if the source is not in the same location as where the program is launched, Debug Tool does not find the source. To avoid this problem, specify the full path name of the source when you compile the program.

For example, if you execute the following series of commands, Debug Tool does not find the source because it is located in another directory (/u/myid/mypgm):

1. Change to the directory where your program resides and compile the program.

   ```
   cd /u/myid/mypgm c89 -g -o "//TEST.LOAD(HELLO)" hello.c
   ```

2. Exit UNIX System Services and return to the TSO READY prompt.
3. Launch the program with the TEST run-time option.

   ```
   Call TEST.LOAD(HELLO) 'test/'
   ```

Debug Tool finds the source if you change the compile command to:

```
c89 -g -o "//TEST.LOAD(HELLO)" /u/myid/mypgm/hello.c
```

This example applies to C++ as well.

The same restriction applies to programs that you compile to run in a CICS environment.

C program: Compiling with the #pragma statement

The TEST/NOTEST compiler option can be specified either when you compile your program or directly in your program, using a #pragma.

This #pragma must appear before any executable code in your program.

The program generates the following:

- Symbol table information
- Symbol information for nested blocks
- Hooks at line numbers:
  - #pragma options (test(SYM,BLOCK,LINE))
  - This is equivalent to TEST(SYM,BLOCK,LINE,PATH).
  - You can also use a #pragma to specify run-time options.

C and C++ programs: Rules for the placement of hooks in functions and nested blocks

For C and C++ programs, the following rules apply to the placement of hooks for getting in and out of functions and nested blocks:

- The hook for function entry is placed before any initialization or statements for the function.
- The hook for function exit is placed just before actual function return.
- The hook for nested block entry is placed before any statements or initialization for the block.
- The hook for nested block exit is placed after all statements for the block.

C and C++ programs: Rules for the placement of hooks in statements and path points

For C and C++ programs, the following rules apply to the placement of hooks for statements and path points:

- Label hooks are placed before the code and all other statement or path point hooks for the statement.
- The statement hook is placed before the code and path point hook for the statement.
- A path point hook for a statement is placed before the code for the statement.

13.1.2 Using mainframe interface debugger (3270 interface debugger)

Perform the following steps to use the mainframe interface debugger:

1. Invoke the program from batch using the appropriate test run-time option.

   In this example, we are debugging the C IVP that was already compiled as shown in Example 13-1. The C IVP program can run with any number of command line parameters. For our initial run, we used no parameters.

   It may be to your advantage to allocate an INSPLOG data set. This will allow the setup commands from this run to be captured, edited, and used as a preferences file for the next run of this program. Remember that log data sets must be sequential and FB 72. We also have had success with using the PC's cut and paste feature to cut the commands we like from the log screen of the debugger in order to create a preferences file.

   For our purposes, the default LE values for the first three parameters are sufficient; we are not using an INSPPREF file. Figure 13-2 shows the resultant command.

Example 13-1 IVP C program

```
Debugging a C IVP program, already compiled:
```

```
  Menu  List  Mode  Functions  Utilities  Help
 ------------------------------------------------------------------------
                              ISPF Command Shell            1 Member processed
Enter TSO or Workstation commands below:

===> call 'GRACINE.DEBUG.TOOL51.TESTLOAD(CIVP)' 'test(,,,*) /'_
 ------------------------------------------------------------------------
 ------------------------------------------------------------------------

Place cursor on choice and press enter to Retrieve command
```

Figure 13-2 ISPF TSO call command against compiled C IVP program

2. The MFI screen looks as shown in Figure 13-3.

```
C        LOCATION: "CBC.SCCNSAM(CCNYIV1)" initialization
Command ===>                                                  Scroll ===> PAGE
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 0 OF
******************************** TOP OF MONITOR ********************************
******************************** BOTTOM OF MONITOR *****************************

SOURCE: CBC.SCCNSAM(CCNYIV1) ----+----3----+----4----+----5----+- LINE: 1 OF 2
       1  /* ECHO ARGUMENTS TO STDOUT
       2   */
       3  #include <stdio.h>
       4  #define NUM_CHARS 36
       5  int i;
       6  int main(ac, av)
LOG 0----+----1----+----2----+----3----+----4----+----5----+----6- LINE: 2 OF
0002 or is not accessible.
0003 IBM Debug Tool Version 5 Release 1 Mod 0
0004 11/23/2004 11:15:45 AM
0005 5655-M18 and 5655-M19: (C) Copyright IBM Corp. 1992, 2004
PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEAR
PF  7:UP         8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEVE
```

Figure 13-3 MFI screen display on entry to C IVP

3. To stop at the call to printf and at the exit, issue the commands:

   ```
   AT CALL printf;
   AT EXIT main;
   ```

   To monitor <i> from the beginning and stop at main to monitor the parameters to main:

   ```
   MONITOR LIST i;
   AT ENTRY main;
   ```

   The results are shown in Figure 13-4.

```
C        LOCATION: "CBC.SCCNSAM(CCNYIV1)" initialization
Command ===>                                                  Scroll ===> PAGE
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF
******************************** TOP OF MONITOR ********************************
0001  1 i   0
******************************** BOTTOM OF MONITOR *****************************

SOURCE: CBC.SCCNSAM(CCNYIV1) ----+----3----+----4----+----5----+- LINE: 1 OF 2
       1  /* ECHO ARGUMENTS TO STDOUT
       2   */
       3  #include <stdio.h>
       4  #define NUM_CHARS 36
       5  int i;
       6  int main(ac, av)
LOG 0----+----1----+----2----+----3----+----4----+----5----+---- LINE: 10 OF 1
0010 The command element listi is invalid.
0011  MONITOR
0012    LIST i ;
0013  AT ENTRY main ;
PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEAR
PF  7:UP         8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEVE
                                                                      02/01
```

Figure 13-4 Debug Tool MFI screen after initial commands

4. Press PF 11 to zoom to the log file; the panel in Figure 13-5 is displayed.

```
C        LOCATION: "CBC.SCCNSAM(CCNYIV1)" initialization
Command ===>                                                    Scroll ===> PAGI
LOG 0----+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 1
******************************** TOP OF LOG ********************************
0001 An error occurred while opening file: INSPLOG . The file may not exist,
0002 or is not accessible.
0003 IBM Debug Tool Version 5 Release 1 Mod 0
0004 11/23/2004 11:58:58 AM
0005 5655-M18 and 5655-M19: (C) Copyright IBM Corp. 1992, 2004
0006  AT CALL printf ;
0007  AT EXIT main ;
0008 The partially parsed command is:
0009  MONITOR
0010 The command element listi is invalid.
0011  MONITOR
0012    LIST i ;
0013  AT ENTRY main ;
******************************* BOTTOM OF LOG ******************************

PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEAI
PF  7:UP         8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEVI
```

Figure 13-5 A zoom look at the log file after our initial commands

5. Press PF11 to go back to the original view. At this point, run the program and wait for the first breakpoint to be reached, as shown in Figure 13-6.

```
C        LOCATION: "CBC.SCCNSAM(CCNYIV1)" :> main ENTRY
Command ===>                                                    Scroll ===> PAGI
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF
****************************** TOP OF MONITOR ******************************
0001  1 i   0
***************************** BOTTOM OF MONITOR ****************************

SOURCE: CBC.SCCNSAM(CCNYIV1) ----+----3----+----4----+----5----+- LINE: 1 OF 2
      1  /* ECHO ARGUMENTS TO STDOUT
      2   */
      3  #include <stdio.h>
      4  #define NUM_CHARS 36
      5  int i;
      6  int main(ac, av)
LOG 0----+----1----+----2----+----3----+----4----+----5----+---- LINE: 11 OF 1
0011  MONITOR
0012    LIST i ;
0013  AT ENTRY main ;
0014  RUN ;
PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEAI
PF  7:UP         8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEVI
```

Figure 13-6 At the first breakpoint

In Figure 13-6, note that the location where Debug Tool has stopped appears at the top line of the screen.

6. At this point you want to add additional monitor statements for ac, av, and *av. Ideally, one should be able to put these monitor statements in one entry, but some experimentation has found that they are not always accepted when the variable is out of scope. For this

reason, we recommend putting a breakpoint at the appropriate scope and turning on the monitoring at that point. The new monitor commands are:

```
MONITOR LIST ac;
MONITOR LIST av;
MONITOR LIST *av;
```

The monitor section at the top of Figure 13-7 now has entries for i, ac, av, and *av.

```
Command ---> _                                                  Scroll ---> PAGE
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF
****************************** TOP OF MONITOR ******************************
0001  1 i    0
0002  2 ac   1
0003  3 av   0x0023704B
0004  4 * av   "civp"
***************************** BOTTOM OF MONITOR ****************************

SOURCE: CBC.SCCNSAM(CCNYIV1) ----+----3----+----4----+----5----+- LINE: 1 OF 2
       1  /* ECHO ARGUMENTS TO STDOUT
       2   */
       3  #include <stdio.h>
       4  #define NUM_CHARS 36
       5  int i;
       6  int main(ac, av)
LOG 0----+----1----+----2----+----3----+----4----+----5----+---- LINE: 14 OF 1
0014  MONITOR
0015    LIST av ;
0016  MONITOR
0017    LIST * av ;
PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEAR
```

Figure 13-7 At entry of main monitors showing initial values

7. Zooming to the log, you now find a summary of all your setup commands, as shown in Figure 13-8.

```
Command ===> _                                                  Scroll ===> PAGE
LOG 0----+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 18
****************************** TOP OF LOG ******************************
0001 An error occurred while opening file: INSPLOG . The file may not exist,
0002 or is not accessible.
0003 IBM Debug Tool Version 4 Release 1 Mod 0
0004 10/08/2003  5:15:19 PM
0005 5655-L24 and 5655-L23: (C) Copyright IBM Corp. 1992, 2003
0006  AT CALL printf ;
0007  AT EXIT main ;
0008  MONITOR
0009    LIST i ;
0010  AT ENTRY main ;
0011  RUN ;
0012  MONITOR
0013    LIST ac ;
0014  MONITOR
0015    LIST av ;
0016  SET SCREEN MONITOR SOURCE LOG ;
0017  MONITOR
0018    LIST * av ;
PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEAR
PF  7:UP         8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEVE
```

Figure 13-8 Current log file after all commands are set up

If you did not set up an INSPLOG this would be a good time to cut the commands from the log file to later create a preference file.

8. Press PF11 to get back to the original view.
9. Run again and the panel in Figure 13-9 is displayed.

```
Command ---> _                                                  Scroll ---> PAGE
MONITOR ----+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 4
****************************** TOP OF MONITOR ******************************
0001  1 i   36
0002  2 ac   0
0003  3 av   0x0D23704C
0004  4 * av   0x0
****************************** BOTTOM OF MONITOR ***************************

SOURCE: CBC.SCCNSAM(CCNYIV1) ----+----3----+----4----+----5----+ LINE: 22 OF 23
      22        exit(i);
      23        }
****************************** BOTTOM OF SOURCE ****************************



LOG 0----+----1----+----2----+----3----+----4----+----5----+---- LINE: 22 OF 25
0022 The operating system has generated the following message:
0023    CEE0199W The termination of a thread was signaled due to a STOP
0024 statement
0025 The current location is CIVP ::> "CBC.SCCNSAM(CCNYIV1)" :> 22.
PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEAR
```

Figure 13-9 At the exit of the program

Note that the printf breakpoint was never triggered because the printf was never executed. Type qq to quit this session.

10. Edit your INSPLOG to create a preference file or paste the value cut from the log. Remember that a preference file should be FB 80 without sequence numbers. Our preference file looks as shown in Figure 13-10.

```
Command ===> _                                                  Scroll ===> CSR
****** ***************************** Top of Data ******************************
==MSG> -Warning- The UNDO command is not available until you change
==MSG>           your edit profile using the command RECOVERY ON.
000001 AT CALL printf ;
000002 AT EXIT main ;
000003 MONITOR
000004    LIST i ;
000005 AT ENTRY main ;
000006 RUN ;
000007 MONITOR
000008    LIST ac ;
000009 MONITOR
000010    LIST av ;
000011 MONITOR
000012    LIST *av ;
****** **************************** Bottom of Data ****************************


 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
```

Figure 13-10 Our new preferences file

11. Allocate the INSPPREF file:

```
alloc FI(INSPPREF) DA('redbk1.dev.pref(civp)')
```

12. Rerun the program with more parameters. Note that this time you can use the default TEST runopt because you are using a INSPPREF file:

```
call 'redbk1.dev.load(civp)' 'test / Testing one two'
```

When the MFI screen comes up, you are at the entry of main with all the monitor variables set as shown in Figure 13-11.

```
Command ===>                                                    Scroll ===> PAGE
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 4
****************************** TOP OF MONITOR ******************************
0001  1 i   0
0002  2 ac   4
0003  3 av   0x0D237230
0004  4 * av   "civp"
***************************** BOTTOM OF MONITOR ****************************

SOURCE: CBC.SCCNSAM(CCNYIV1) ----+----3----+----4----+----5----+- LINE: 6 OF 23
       6  int main(ac, av)
       7      int  ac;
       8      char *avÝ";
       9      {
      10      i= 0;
      11      ++av, --ac;
LOG 0----+----1----+----2----+----3----+----4----+----5----+---- LINE: 14 OF 17
0014  MONITOR
0015    LIST av ;
0016  MONITOR
0017    LIST * av ;
PF  1:?          2:STEP      3:QUIT      4:LIST      5:FIND      6:AT/CLEAR
PF  7:UP         8:DOWN      9:GO       10:ZOOM     11:ZOOM LOG  12:RETRIEVE
```

Figure 13-11 Stopped at entry with all monitor variables set

13. Run the program. This time it stops at the call to printf for the first parameter as shown in Figure 13-12.

```
Command ---> _                                                  Scroll ---> PAGE
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 4
****************************** TOP OF MONITOR ******************************
0001  1 i   0
0002  2 ac  3
0003  3 av  0x0D237234
0004  4 * av   "testing"
***************************** BOTTOM OF MONITOR ****************************

SOURCE: CBC.SCCNSAM(CCNYIV1) ----+----3----+----4----+----5----+ LINE: 12 OF 23
      12        for (; 0 < ac; ++av, --ac)
      13            {
      14            i= i + printf("%s", *av);
      15            if (1 < ac)
      16                printf(" ");
      17            else
LOG 0----+----1----+----2----+----3----+----4----+----5----+---- LINE: 15 OF 18
0015    LIST av ;
0016  MONITOR
0017    LIST * av ;
0018  RUN ;
PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEAR
PF  7:UP         8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEVE
```

Figure 13-12 Stopped before printf call

13.1.3 Using IBM Distributed Debugger

The IBM Distributed Debugger is a client/server application that enables you to detect and diagnose errors in your programs. This client/server design makes it possible to debug programs running on systems accessible through a network connection as well as those programs running on your workstation.

The debugger server, also known as a debug engine, runs on the same system where the program you want to debug runs. This system can be your workstation or a system accessible through a network. If you debug a program running on your workstation, you are performing local debugging. If you debug a program running on a system accessible through a network connection, you are performing remote debugging.

The debug engine can either start the program you want to debug or attach to an already running program. Once started, the debug engine controls the execution of your program.

The debugger client is a graphical user interface where you can issue commands used by the debug engine to control the execution of your program. For example, you can set breakpoints, step through your code, and examine the contents of variables.

To use the remote debugging capability for stored procedures on the Intel® and UNIX platforms, you need to install the IBM Distributed Debugger. The IBM Distributed Debugger is included on the VisualAge for Java Professional Edition CD. The debugger client runs only on the Windows platform. Supported server platforms include Windows, AIX®, and Solaris. At this time, only Java and C stored procedures can be debugged remotely. At a later date, the Debugger will be integrated into DB2. Support for SQL procedures will then be available.

To debug SQL procedures on the OS/390 platform, you must also have the IBM C/C++ Productivity Tools for OS/390 R1 product. For more information on this product, go to the following Web site:

http://www.ibm.com/software/ad/c390/pt/

Distributed Debugger for the workstation

The Distributed Debugger can be uploaded from the following Web site:

http://www-1.ibm.com/support/docview.wss?uid=swg24007187

Follow these steps to install the Distributed Debugger:

1. Download and unzip the file.
2. Select **Start** → **Run**, and type:

   ```
   x:\setup -ParentProduct=IBM VisualAge COBOL
   ```

   (*x* is the location of the unzipped files)
3. When prompted for type of install, select **Typical**.
4. When prompted for where to install, click Browse and locate where the Debugger was originally installed. It will probably be in:

   c:\Program Files\IBM\IBMDebug
5. Complete the installation and reboot.
6. If you have VisualAge PL/I installed, follow the previous instructions, but using the command:

   ```
   x:\setup -ParentProduct=IBM VisualAge PLI
   ```

   (*x* is the location of the unzipped files)

Installing the Distributed Debugger on the workstation

Complete the following steps to install the distributed debugger on the workstation:

1. Unzip the file idebug_929_windows.zip. If you accept the defaults a directory will be created and in all files will be copied into the directory.
2. Find the program setup.exe and run it to launch the install.
3. We recommend that you accept all defaults. A window is displayed informing you the install has been completed successfully.
4. Optionally, create a shortcut icon to the desktop which will speed up the process of starting the Remote Debugger. To do this, right-click on your Windows desktop to activate the Windows menu; select **New** → **Shortcut**. Figure 13-13 displays the result.

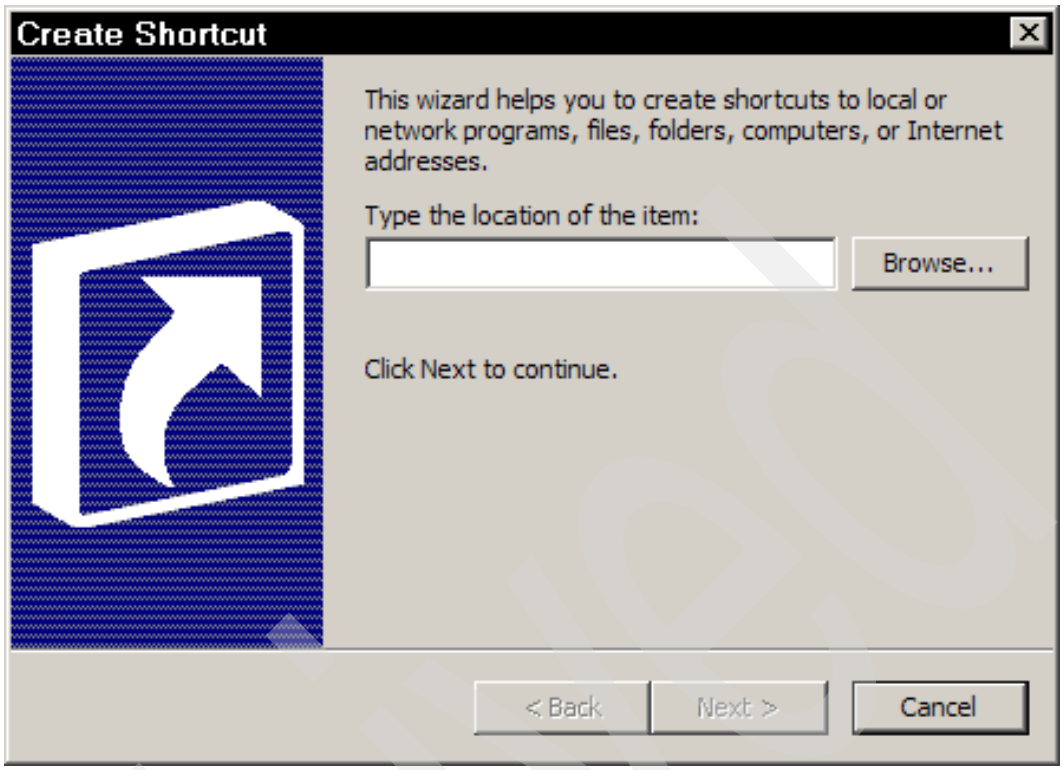


Figure 13-13 Create desktop shortcut to activate Remote Debugger

5. Enter the location of the Remote Debugger in the Type the location of the item field, as shown in Figure 13-14, and click Next.

> **Important:** It is very important that you include `-qdeamon` after `C:\IBMDebug\bin\idebug.exe` in the location field as shown in the figure. Otherwise the installation cannot complete successfully.

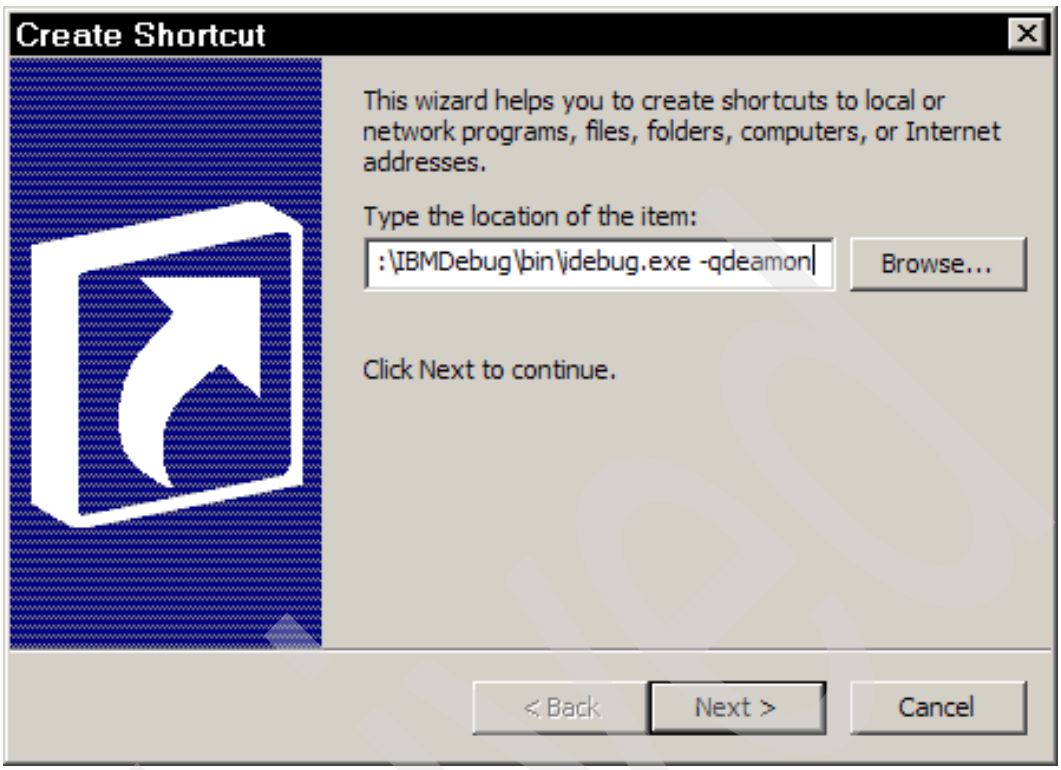


Figure 13-14 Location where remote debugger will reside

6. The window shown in Figure 13-15 appears. Type a name for the shortcut created and click Next.

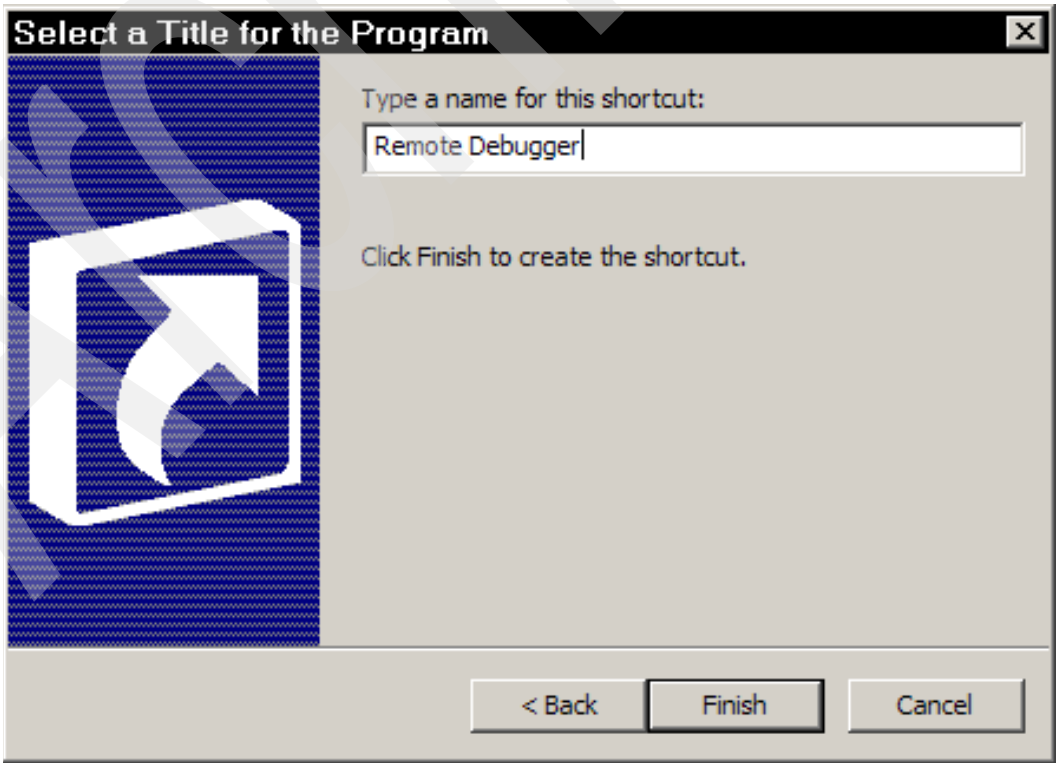


Figure 13-15 Name given to the created shortcut

7. An icon is created on the desktop. Double-click the icon to start the IBM Distributed Debugger, shown in Figure 13-16.

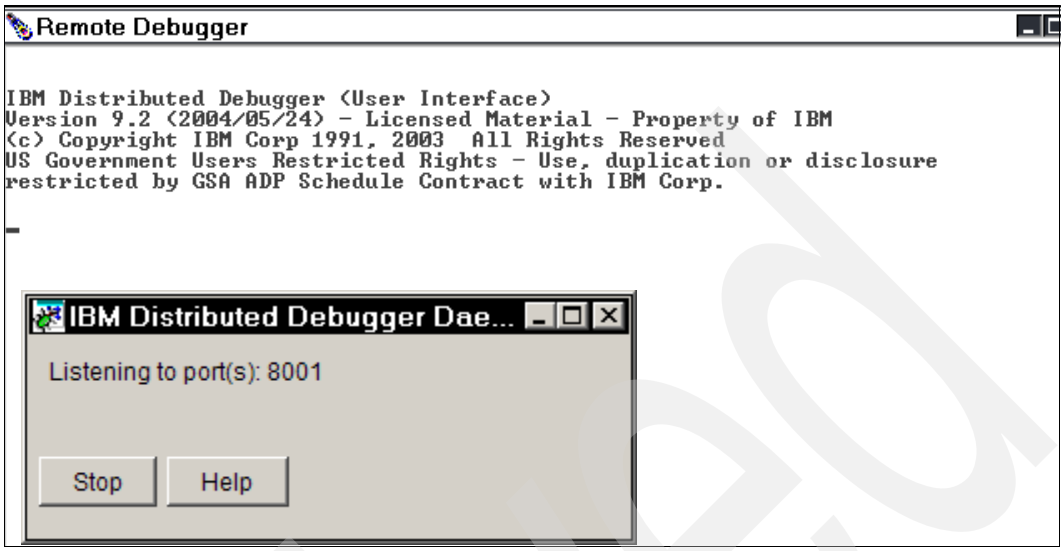


Figure 13-16 Remote Debugger active on workstation

8. Installation is complete. To stop the Remote Debugger, click Stop.
9. For the purposes of this example, the IBM Distributed Debugger is watching port 8001 on a machine at TCP/IP address 9.28.205.246. Therefore, the TSO command to call the C++ IVP program we compiled previously is:

```
call 'gracine.debug.tool51.testload(civp)' 'test(,,,vadtcpip&9.28.205.246%8001:*) /'
```

As with the MFI debugger, on entry to the debugger, one can only monitor variables that are at global scope, as shown in Figure 13-17.

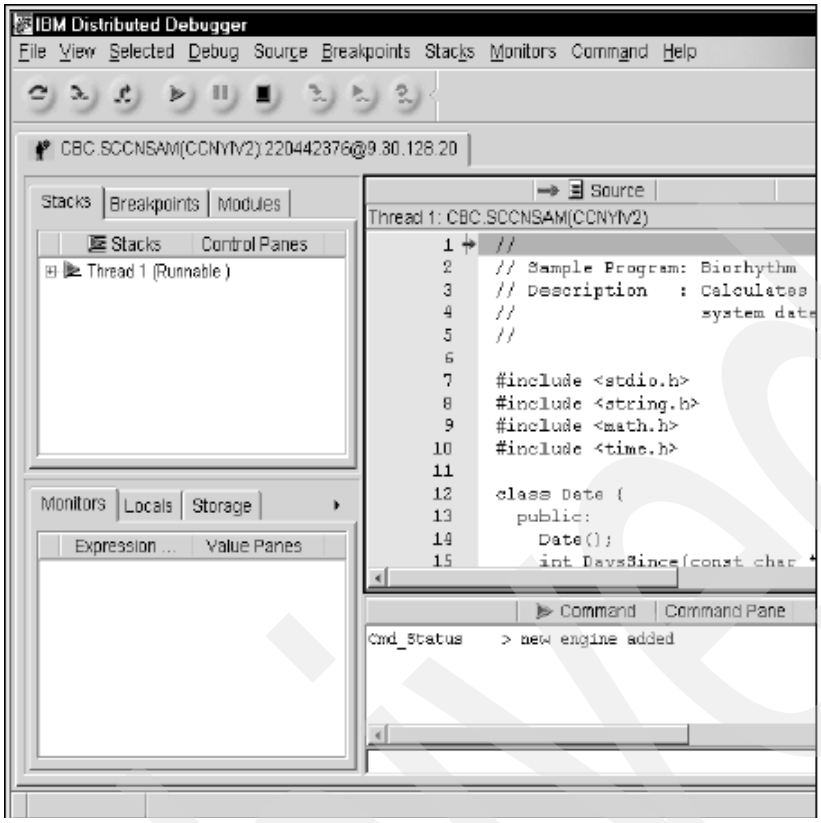


Figure 13-17 Remote Debugger display

Since there are no global scope variables in the C IVP, use the step button in the upper left hand corner to step into main. Then choose to monitor the variable *i* by doing the following:

a. Select **Monitors**.

b. Select **Monitor Expression** (or press Shift F9).

c. Fill in the variable name.

This will put bio in the monitor window with a small + beside it. Click **+** to expand the content of bio as shown in Figure 13-18.

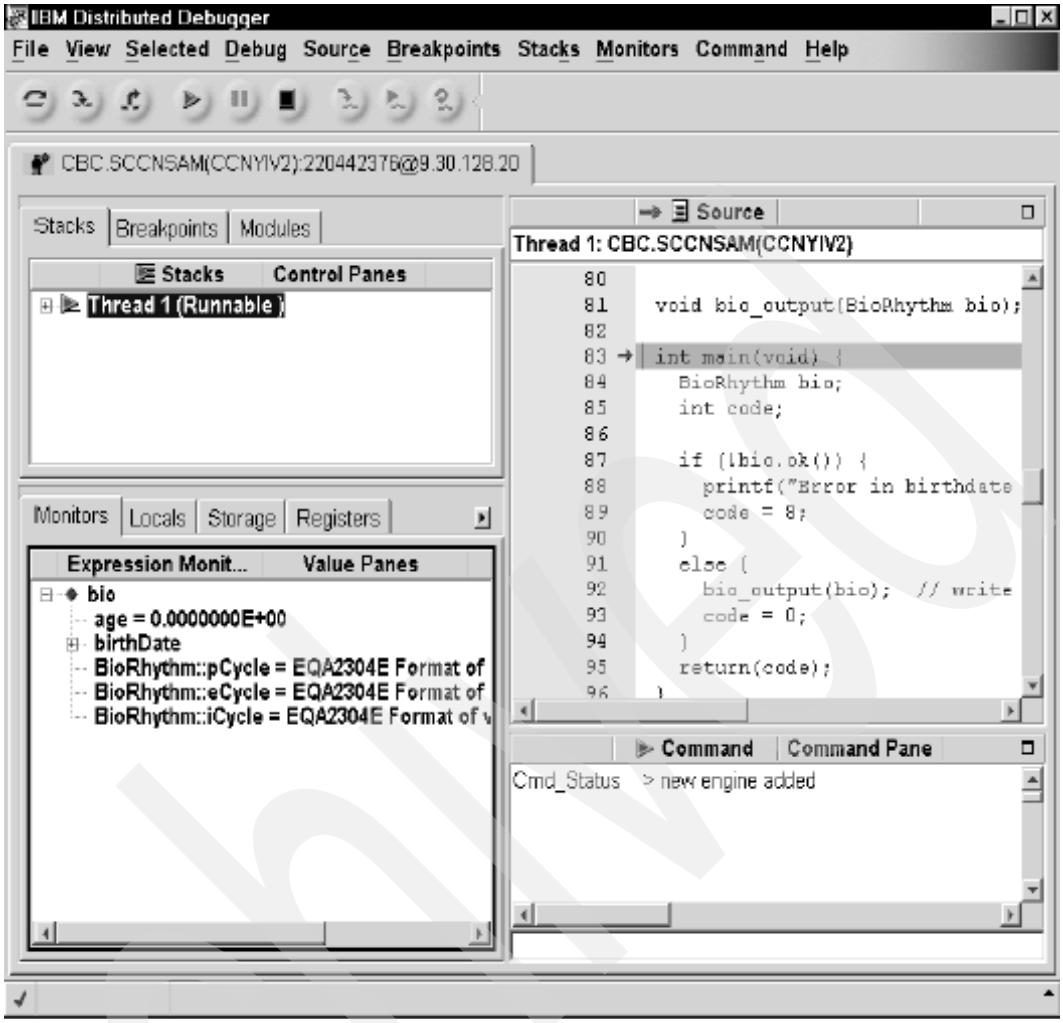


Figure 13-18 The monitor screen showing the initial bio values

Alternatively, if you only want to monitor bio while you are in function Main, simply choose the Locals tab in the bottom left-hand window to view all current locals, as shown in Figure 13-19.

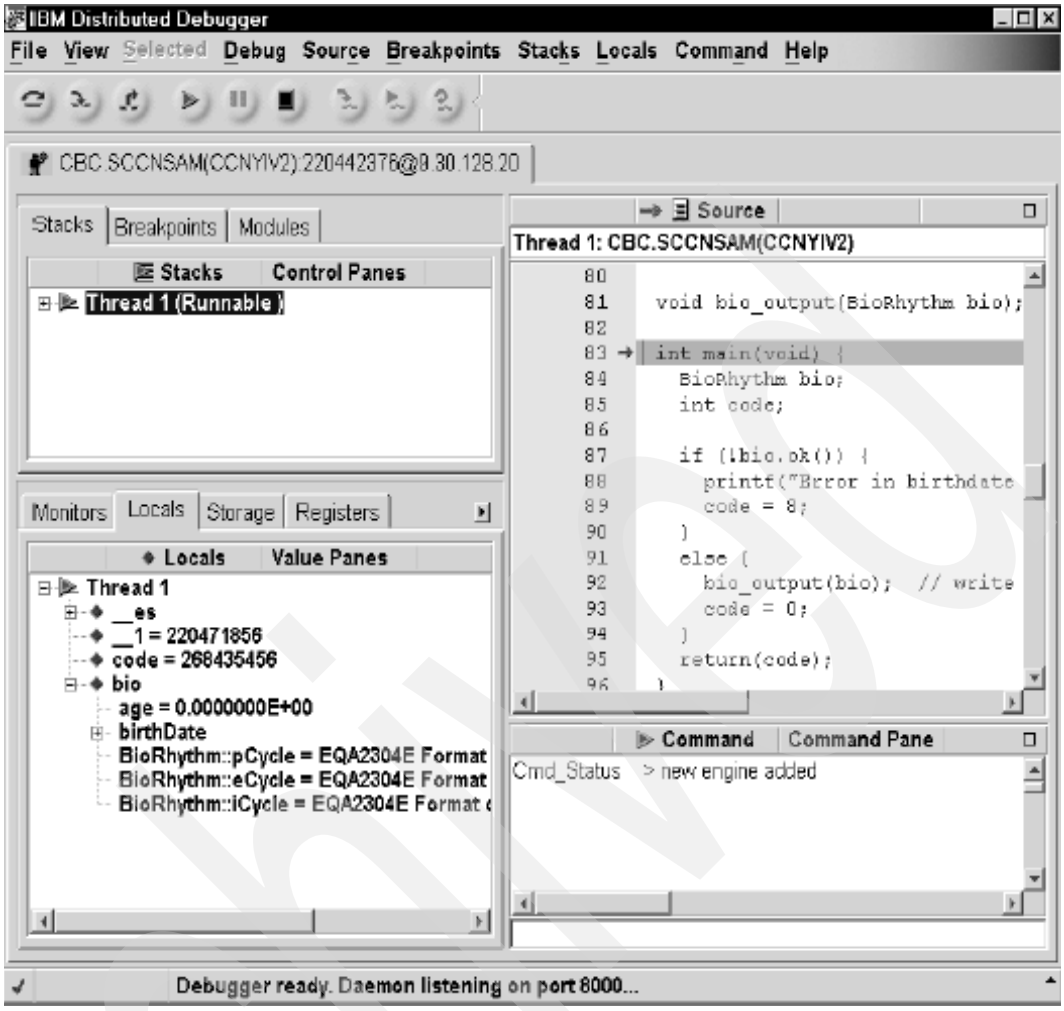


Figure 13-19 Looking at the local variables

10. You can then set a function-level breakpoint at the entry to BioRythm::ok() using the following steps:
    a. Select the **Breakpoint** menu.
    b. Select **Set Function**.
    c. Click the arrow under Function to get a list of all available functions.
    d. Click OK; the panel shown in Figure 13-20 is returned.

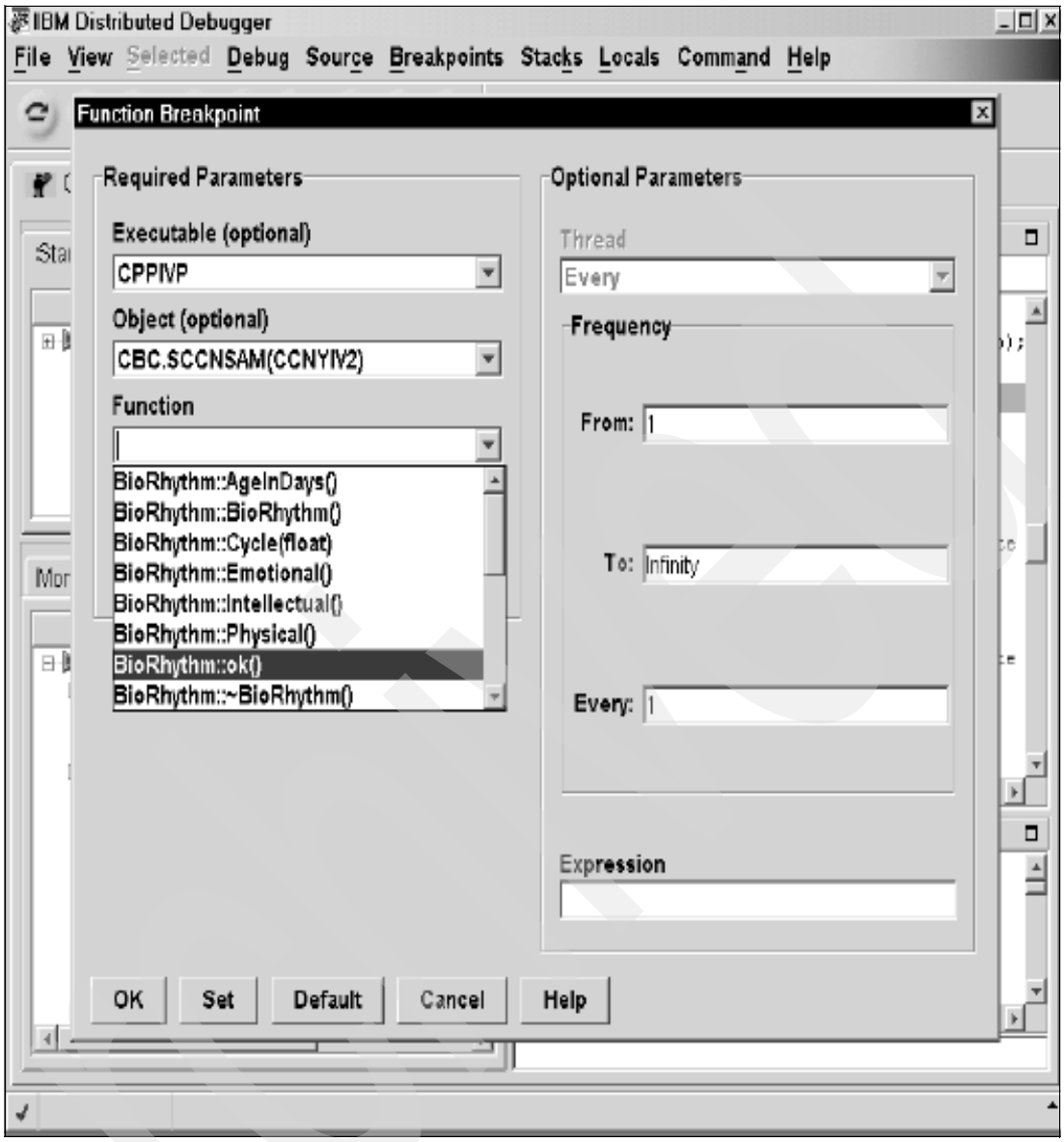


Figure 13-20 Setting a breakpoint on Biorythm::ok

11. Run to the breakpoint by pressing the small green triangle on the command icon line. Remember to switch back to the TSO screen to fill in the information when prompted. Control will return to the remote debugger once the birth date has been entered.
12. Click to the monitor window and expand bio so you can see the age field as shown in Figure 13-21.

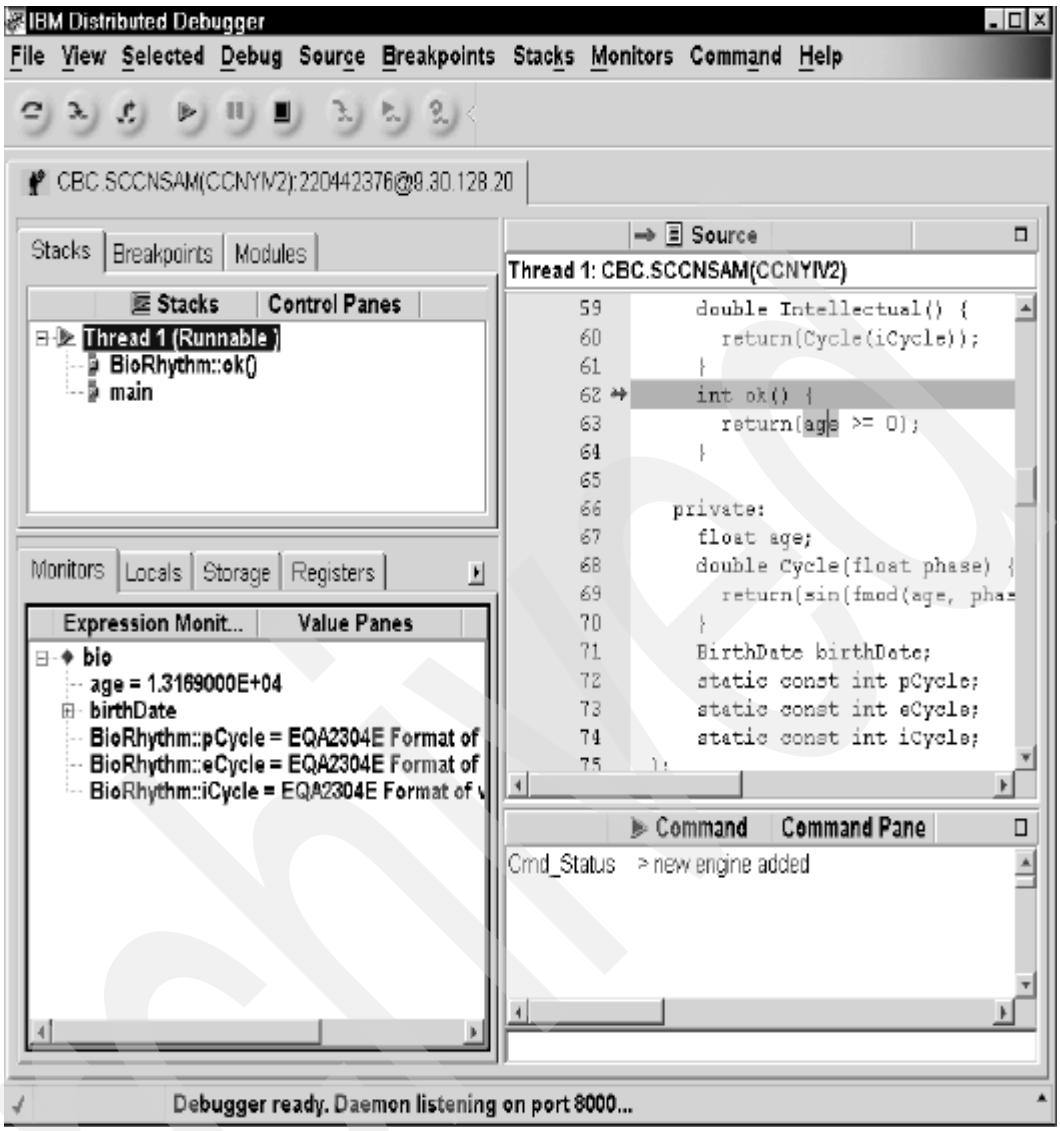


Figure 13-21 Bio expanded

13. Alternatively, if you did not already have the monitor of the variable bio available, you could go to the source window, highlight the word **age**, and then use the left mouse button to get options and select a program monitor as shown in Figure 13-22.

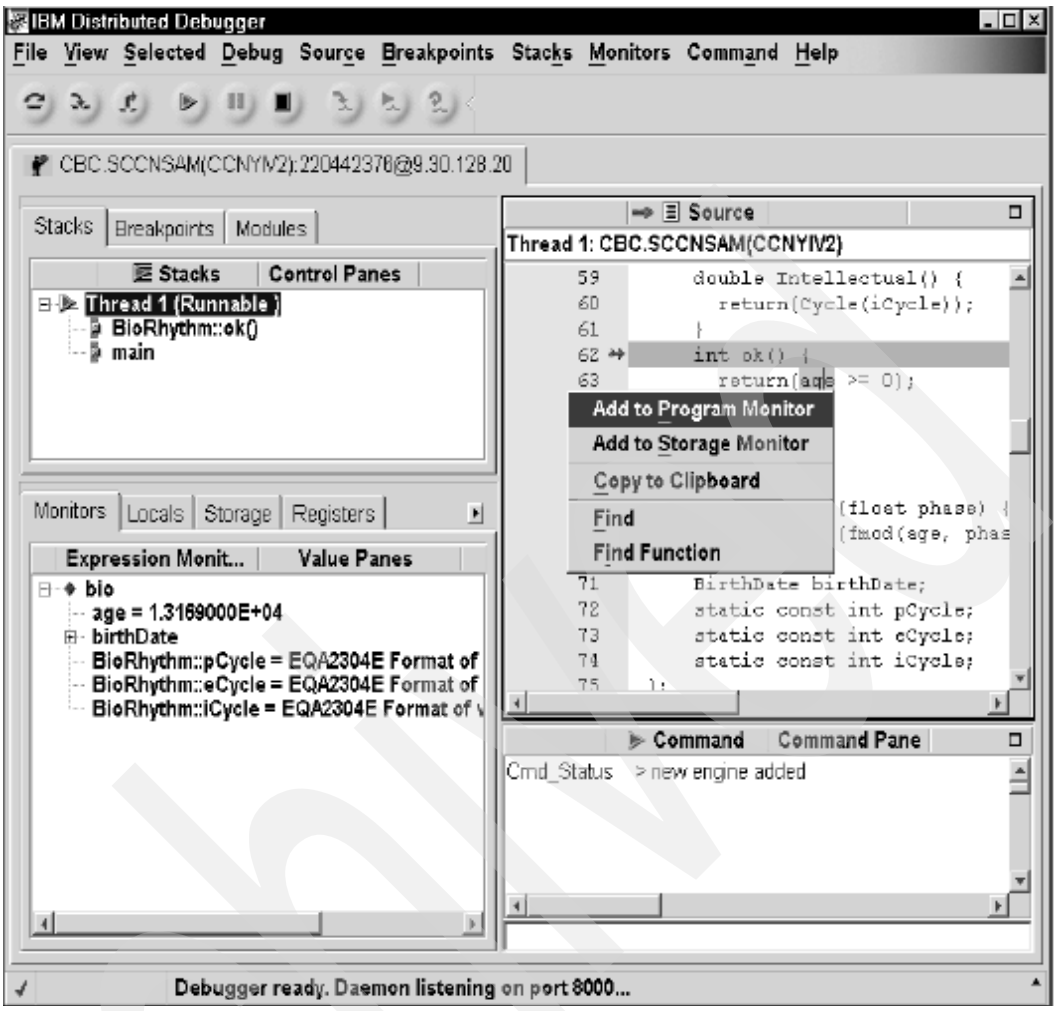


Figure 13-22 Adding a program monitor for age

14. Monitored age is displayed as in Figure 13-23.

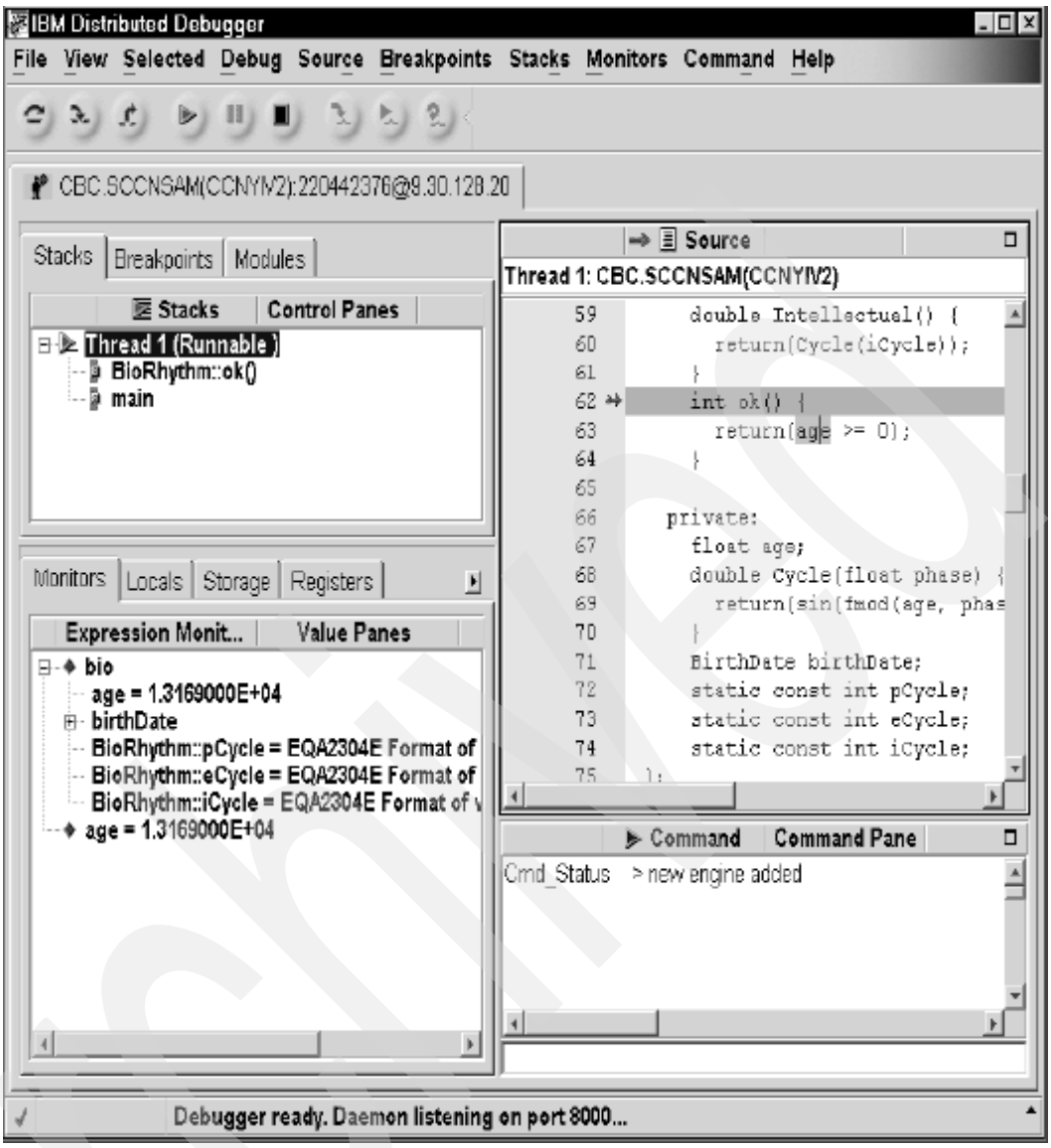


Figure 13-23 Remote debugger now monitors the age variable

15. Continue to debug as desired.

Remote Debugger requirements, advantages, and restrictions are summarized in Table 13-1.

Table 13-1 Remote Debugger requirements, advantages, and restrictions

Hardware requirements: Pentium®-based 100MHz processor (minimum) or higher (recommended). 128MB memory or higher, and 100MB hard drive space.	
Software requirements: Microsoft® Windows NT Version 4.0 (Service Pack 4 or later required) or Microsoft Windows 2000 Professional or Microsoft Windows XP Professional	
Install requirements: Administrator authority	
Alternative to the MFI “dual VTAM” Debug Tool install definitions	Yes
Ease of use	Yes, over the MFI to migrate from vendor tools

Single view of all program Working Storage	Yes, not available on the MFI
Save of all Monitor and Breakpoint definitions across DT invocations for the same program	Yes, not available on the MFI
Ease of use for HEX data monitor display and update	Yes
Ease of use for HEX storage display and update	Yes
Assembler Support	No
COMMANDS File	No
Preferences File	Yes
SESSION Logging	No
Large Working Storage and Linkage Display Window	Yes (limited by the workstation memory)
DYNAMIC Patching	No
PSUEDO CODE	No
ALLOCATE	No
AT CALL	Yes
AT CHANGE	Yes
AT CURSOR	Yes
AT DELETE	No
AT ENTRY	Yes
AT EXIT	No
AT GLOBAL	No
AT LABEL	Yes
AT OCCURANCE	Yes
AT PATH	No
AT TERMINATION	No
CALL %DUMP	No
CALL %FA	No
CALL entry-name	Yes
CALL procedure	Yes
COMMENT	No
COMPUTE	No
DESCRIBE	Yes
EVALUATE	No
FREE	No
GO BYPASS	Yes
LIST	Yes
MOVE	Yes
PERFORM	No
AUTOMONITOR	Yes

PLAYBACK	No
QUALIFY RESET	Yes
QUERY	No
RETRIEVE	No
SET FREQUENCY	No
SET PACE	No
TRIGGER	No
TSO	No

13.2 Debugging PL/I programs using Debug Tool

This section describes how to:

- Prepare a PL/I application for debugging
- Debug a PL/I application with the mainframe interface (MFI)
- Debug a PL/I application with the remote interface

This section is based on an example given in the manual, *OS PL/I Version 2 Programming: Using PLITEST*.

13.2.1 Preparing a PL/I program for debugging

For the example described in this section we used Enterprise PL/I for z/OS and OS/390 V3 R3 and Debug Tool 5.1.

The TEST option is used by the PL/I compiler to enable debugging for a batch compile, a TSO compile, or a compile with the Debug Tool Utilities (DTU). The TEST option has many suboptions, but for optimal debugging, we recommend always taking the default. A sample compile of our test program is shown in Example 13-2.

Example 13-2 Compile job

```
//EPLICOMP   JOB ,
//  MSGLEVEL=(2,0),CLASS=A,
//  NOTIFY=&SYSUID,REGION=1024M
//ORDER JCLLIB ORDER=(ENPLI330.SIBMZPRC)
//*-----------------------------------------------------------------
//* Compile PLI with the Enterprise PLI V3R3 compiler
//*-----------------------------------------------------------------
//COMPILE   EXEC IBMZCB,
//          LNGPRFX=ENPLI330,
//          PARM='OBJECT,OPTIONS,TEST(ALL)'
//PLI.SYSIN DD DSN=CHABERT.PLI.SOURCE(KNIGHT),DISP=SHR
//BIND.SYSLMOD DD DSN=CHABERT.PLI.LOAD(KNIGHT),DISP=SHR
```

The debug information for PL/I is currently stored inside the executable and can increase the size of the executable.

In general we find that it is easiest to add the appropriate extra option to your standard development build method rather than using the DTU compile facility. However, the examples used in this chapter are all small enough that any compile method will work.

Note: Debug Tool will require the exact source files that were compiled, for use when debugging Enterprise PL/I. It is best to store them in a permanent location and then compile. If Debug Tool has trouble finding the source files, the SET SOURCE command can be used to tell Debug Tool where to find them.

Notes on preparing a PL/I V3R3 program for Debug Tool

The main considerations for preparing the program are the following:

1. Compile with the TEST(ALL) and SOURCE compiler options.

   If all of the following tasks are completed, Debug Tool uses the hooks inserted by the Dynamic Debug facility instead of the hooks inserted by the compiler:

   – The Dynamic Debug facility is installed on your system.

   – You have not deactivated the Dynamic Debug facility by entering the SET DYNDEBUG OFF command.

   – You compile your program with one of the following compilers:

     - PL/I for MVS and VM Version 1 Release 1 (Program Number 5688-235)
     - VisualAge PL/I for OS/390 Version 2 Release 2 (Program Number 5655-B22)
     - Enterprise PL/I for z/OS and OS/390 Version 3 (Program Number 5655-H31)

   Using the hooks inserted by the Dynamic Debug facility provides better performance when you are debugging your program. However, some path breakpoints become unavailable. If you need to use those breakpoints, deactivate the Dynamic Debug facility by entering the SET DYNDEBUG OFF command.

   The SOURCE compiler option is required to generate a listing file. Keep the listings in a file of the name userid.pgmname.LIST or use the SET SOURCE command to find it.

   During a debug session, Debug Tool displays the first file it finds named userid.pgmname.list in the Source window. If Debug Tool cannot find the listing at this location, use one of the following methods to associate your source listing with the program you are debugging:

   – Enter the SET SOURCE command with the name of the location or file.

   – Enter the SET DEFAULT LISTINGS command with the name of the location or file.

   – Specify the EQADEBUG DD statement with the name of the location or file.

   – Code the EQAUEDAT user exit with the new location.

2. You must link with the Language Environment SCEELKED library. The OS/PLI PLIBASE or SIBMBASE libraries are not acceptable.

3. You must run the program on the Language Environment run time library (SCEERUN, with SCEERUN2 if we are OS/390 V2R10 or higher).

Quick run through the program

A sample job to run our test program is shown in Example 13-3.

Example 13-3 JCL to run the test program

```
//PLIRUN   JOB 'PLI RUN',
// CLASS=A,MSGCLASS=X,MSGLEVEL=(1,1),
// NOTIFY=&SYSUID,
// REGION=32M
//*
//KNIGHT   EXEC PGM=KNIGHT,REGION=2048K
//STEPLIB  DD  DSN=CHABERT.PLI.LOAD,DISP=SHR
```

```
//         DD  DSN=CEE.SCEERUN,DISP=SHR
//SYSPRINT DD  SYSOUT=*
//CEEDUMP  DD  SYSOUT=*
//SYSUDUMP DD  SYSOUT=*
```

A quick execution of the program generates the results shown in Figure 13-24.

```
  Display  Filter  View  Print  Options  Help
 ------------------------------------------------------------------------------
 SDSF OUTPUT DISPLAY PLIRUN   JOB03138  DSID   101 LINE 0       COLUMNS 02- 81
 COMMAND INPUT ===> _                                          SCROLL ===> CSR
********************************* TOP OF DATA ***********************************
    3   10   39    0   27   16   43    0
    0    0   26   17   40    0   34    0
    7   20    0   36    0   44    0    0
    0   25    0    0    0   35    0   51
   21    0    0   56    0   52    0   64
    0    0   22    0    0    0   50    0
    0    0   61    0   53    0   63    0
   58    0    0    0   62    0    0    0
******************************** BOTTOM OF DATA *********************************
                                                                        04/021
```

Figure 13-24 Debug Tool - PLI - Original program output

Notice the following things about this output:

- There is no 1 indicating the starting point.
- There are lots of 0s, indicating that the knight didn't make it to that point.

The correct output is shown in Figure 13-25. Notice the lack of 0s, and the 1 in the upper left-hand corner.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------------
EDIT       CHABERT.DEBUG.PRINT                                Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
****** ***************************** Top of Data ******************************
000001 1     1   16   27   22    3   18   47   56
000002      26   23    2   17   46   57    4   19
000003      15   28   25   64   21   48   55   60
000004      24   35   30   45   58   61   20    5
000005      29   14   63   34   49   44   59   54
000006      36   31   38   41   62   53    6    9
000007      13   40   33   50   11    8   43   52
000008      32   37   12   39   42   51   10    7
****** **************************** Bottom of Data ****************************
```

Figure 13-25 Debug Tool - PLI - Correct program output

13.2.2 Debugging a PL/I application with the mainframe interface (MFI)

The KNIGHT program used as our example determines a knight's tour—the moves a knight makes—as it lands on each square once and only once when it starts in the upper left-hand corner of the board. This program intentionally contains some errors; in this section we demonstrate how to use Debug Tool to find those errors.

For your convenience, the source for the program is as shown in Example 13-4. Take a moment to read through the program comments to familiarize yourself with the structure and function of this program.

Example 13-4 The Knight program

```
*PROCESS STMT SOURCE LANGLVL(SAA2);
 knight: proc options(main);
   /*************************************************************/
   /*  this program determines a knight's tour starting from    */
   /*  a square in the corner of the board.                     */
   /*************************************************************/
   /*  the size of the board is established here.              */
   /*************************************************************/
   dcl size            fixed bin(31) value(8);
   /*************************************************************/
   /*  rowjump and coljump define the numbers of rows and       */
   /*  columns respectively in each of the 8 possible moves     */
   /*  that a knight can make.                                  */
   /*  thus rowjump(1) and coljump(1) represent the move        */
   /*  from "a" to "b" in the following diagram                 */
   /*        - - - - -                                          */
   /*       |a| | | | |                                         */
   /*        - - - - -                                          */
   /*       | | | | | |                                         */
   /*        - - - - -                                          */
   /*       | |b| | | |                                         */
   /*        - - - - -                                          */
   /*************************************************************/
```

```
dcl rowjump(8)          fixed bin(15)
                        static init(-2,-1,+2,-1,+1,+1,+2,-2);
dcl coljump(8)          fixed bin(15)
                        static init(+1,+2,-1,-2,+2,-2,-1,-1);
/*************************************************************/
/*  the land array holds for each square the number of the    */
/*  move when the knight landed on that square.               */
/*************************************************************/
dcl land(size,size)           pic'ZZZ9';
/*************************************************************/
/*  the rank array holds a ranking for each square to which   */
/*  the knight could move - this includes squares in the      */
/*  two imaginary rows and columns that surround the board.   */
/*  the knight will move to the square with the lowest        */
/*  positive rank.                                            */
/*  the rank of all squares off the board is zero, and        */
/*  consequently the knight will never move off the board.    */
/*  the rank of all other squares is equal to the number      */
/*  of moves possible from that square. thus, initially       */
/*  the rank of the four corner squares is 2.                 */
/*************************************************************/
dcl rank(-1:2+size,-1:2+size)    fixed bin(7);
/*************************************************************/
/*  the following variables are used for temporary purposes.  */
/*************************************************************/
dcl (minrank,minmove)   fixed bin(15);
dcl testrank            fixed bin(15);
dcl (m,r,c,k)           fixed bin(15);
/*************************************************************/
/*  note that rankings are kept for two rows or columns       */
/*  beyond the real edges of the board.                       */
/*  initialize all those rankings                             */
/*  the rank of a square to which the knight should not move  */
/*  will be zero. hence the initial rank of all the squares   */
/*  off the board is zero.                                    */
/*************************************************************/
call initialize_Rankings;
/*************************************************************/
/*  initialize all those moves to zero.                       */
/*************************************************************/
land = 0;
/*************************************************************/
/*  make the first move in the tour                           */
/*************************************************************/
m = 1;
r = 1; c = 1;
land(r,c) = m;
/*************************************************************/
/*  determine the moves in the tour                           */
/*************************************************************/
do while ( m < size*size ) until ( minrank = 0 );
  /*********************************************************/
  /*  determine the best move to make                      */
  /*********************************************************/
  minrank = 0;
```

```
  do k = 1 to 8;
    testrank = rank(r+rowjump(k),c+coljump(k));
    if testrank > 0 then
      if ( minrank = 0 ) | ( minrank > testrank) then
        do;
          minrank = testrank;
          minmove = k;
        end;
      else;
    else;
  end;
  /*********************************************************/
  /*  make the move if one was found                       */
  /*********************************************************/
  if minrank > 0 then
    do;
      /*******************************************************/
      /*  lessen rank of squares not moved to                */
      /*******************************************************/
      do k = 1 to 8;
        if rank(r+rowjump(k),c+coljump(k)) > 0 then
          rank(r+rowjump(k),c+coljump(k))
            = rank(r+rowjump(k),c+coljump(k)) - 1;
        else;
      end;
      /*******************************************************/
      /*  make the move                                      */
      /*******************************************************/
      r = r + rowjump(minmove);
      c = c + coljump(minmove);
      m = m + 1;
      land(r,c) = m;
    end;
  else;
end;
/*************************************************************/
/*  print out the tour                                       */
/*************************************************************/
do r = 1 to size;
  put skip(1);
  put edit( (land(r,c) do c = 1 to size)) ( x(1),p'ZZZ9' );
end;
/*************************************************************/
/*  initialize rankings for all squares                      */
/*************************************************************/
initialize_Rankings: proc;
 dcl (r,c,k)           fixed bin(15);
  rank = 0;
  do r = 1 to size;
    do c = 1 to size;
      rank(r,c) = 8;
    end;
  end;
  do r = 1 to size;
    do c = 1 to size;
```

```
      do k = 1 to 8;
        if rank(r+rowjump(k),c+coljump(k)) = 0 then
          rank(r,c) = rank(r,c) - 1;
        else;
      end;
    end;
  end;
end;
end knight;
```

To begin debugging the KNIGHT program, perform the following steps:

1. Invoke the program from batch using the appropriate test runtime option. In this example, we are debugging the KNIGHT module that was already compiled.
2. It may be to your advantage to allocate an INSPLOG data set, since the rank and land arrays are quite large and you may wish to review the debug session after it has completed.
3. Start up the MFI debugger using the CLIST shown in Example 13-5.

Example 13-5 CLIST to run KNIGHT and start the MFI debugger

```
ALLOCATE FILE(INSPPREF) DATASET('CHABERT.DEBUG.PREF(KNIGHT)')  REUSE
ALLOCATE FILE(INSPLOG)  DATASET('CHABERT.DEBUG.LOG(KNIGHT)')   REUSE
ALLOCATE FILE(SYSPRINT) DATASET('CHABERT.DEBUG.PRINT')         REUSE
CALL 'CHABERT.PLI.LOAD(KNIGHT)' 'TRAP(ON) TEST(,,,*) / 8'
FREE DATASET('CHABERT.DEBUG.PREF(KNIGHT)')
FREE DATASET('CHABERT.DEBUG.LOG(KNIGHT)')
FREE DATASET('CHABERT.DEBUG.PRINT')
```

4. The resultant MFI debugging screen is shown in Figure 13-26.

```
PL/I     LOCATION: 'CHABERT.PLI.SOURCE(KNIGHT)' initialization
Command ===> _                                          Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 0 OF 0
****************************** TOP OF MONITOR ******************************
****************************** BOTTOM OF MONITOR ***************************




SOURCE: CHABERT.PLI.SOURCE(KNIGHT) ---3----+----4----+----5----+ LINE: 1 OF 214
****************************** TOP OF SOURCE *******************************
       1  *PROCESS STMT SOURCE LANGLVL(SAA2);                                .
       2   knight: proc options(main);                                       .
       3     /*************************************************************/ .
       4     /*                                                           */ .
       5     /*  this program determines a knight's tour starting from    */ .
       6     /*  a square in the corner of the board.                     */ .
       7     /*                                                           */ .
       8     /*************************************************************/ .
LOG 0----+----1----+----2----+----3----+----4----+----5----+---- LINE: 19 OF 34
PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEAR
PF  7:UP         8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEVE
                                                                    02/015
```

Figure 13-26 Debug Tool - PLI - MFI screen on entry

5. Take a couple of steps into the program and set a break point at the exit of the initialize_Rankings routine using the command:

   ```
   AT EXIT initialize_Rankings
   ```

6. Now run the program to that breakpoint by typing GO or pressing F9.

> **Note:** The DESCRIBE ATTRIBUTES command is not supported for Enterprise PL/I. If you are using another version of PL/I, you could verify the attributes of rank at this point using the DESCRIBE ATTRIBUTES rank.

7. Verify the value of the variable *rank* by using the command:

   ```
   LIST rank
   ```

   Figure 13-27 shows a portion of the log (obtained by pressing PF 11) after this command has been run.

```
PL/I     LOCATION: 'CHABERT.PLI.SOURCE(KNIGHT)' :> INITIALIZE_RANKINGS EXIT
Command ===> _                                                Scroll ===> CSR
LOG 0----+----1----+----2----+----3----+----4----+----5----+-- LINE: 166 OF 184
0166 EQA1140I RANK( 9, 4) =       0
0167 EQA1140I RANK( 9, 5) =       0
0168 EQA1140I RANK( 9, 6) =       0
0169 EQA1140I RANK( 9, 7) =       0
0170 EQA1140I RANK( 9, 8) =       0
0171 EQA1140I RANK( 9, 9) =       0
0172 EQA1140I RANK( 9,10) =       0
0173 EQA1140I RANK(10,-1) =       0
0174 EQA1140I RANK(10, 0) =       0
0175 EQA1140I RANK(10, 1) =       0
0176 EQA1140I RANK(10, 2) =       0
0177 EQA1140I RANK(10, 3) =       0
0178 EQA1140I RANK(10, 4) =       0
0179 EQA1140I RANK(10, 5) =       0
0180 EQA1140I RANK(10, 6) =       0
0181 EQA1140I RANK(10, 7) =       0
0182 EQA1140I RANK(10, 8) =       0
0183 EQA1140I RANK(10, 9) =       0
0184 EQA1140I RANK(10,10) =       0
PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEAR
PF  7:UP         8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEVE
                                                                     02/015
```

Figure 13-27 Debug Tool - PLI - MFI Zoomed log

   You can now use the F7 key to move back up and view all the values of rank.

8. On line 63 of the source there is a comment pointing out that the initial rank of the four corners is 2. If you move back in this list, you see that not all corners have a value of 2 (the screen shows corner (8,1) and corner (8,8) with values of 2).

9. Issue the following commands so you can see the corner values on a single screen as shown in Figure 13-28:

   ```
   monitor list (rank(1,1),rank(1,8))
   monitor list (rank(8,1),rank(8,8))
   ```

```
PL/I     LOCATION: 'CHABERT.PLI.SOURCE(KNIGHT)' :> INITIALIZE_RANKINGS EXIT
Command ===> _                                                  Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 4
******************************** TOP OF MONITOR ********************************
0001  1 RANK ( 1, 1 )        1
0002    RANK ( 1, 8 )        3
0003  2 RANK ( 8, 1 )        2
0004    RANK ( 8, 8 )        2
****************************** BOTTOM OF MONITOR ******************************


SOURCE: CHABERT.PLI.SOURCE(KNIGHT) ---3----+----4----+----5--- LINE: 210 OF 214
     210      end;                                                          8
     211                                                                    .
     212    end;                                                            1
     213                                                                    .
     214   end knight;                                                      0
****************************** BOTTOM OF SOURCE *******************************



LOG 0----+----1----+----2----+----3----+----4----+----5----+-- LINE: 188 OF 188
PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEAR
PF  7:UP         8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEVE
                                                                     02/015
```

Figure 13-28 Debug Tool - PLI - MFI Rank values for the 4 corners

10. Re-create the rankings (by calling initialize_Rankings explicitly) and verify that the rankings are correct after the first group of do statements execute. But first you need to set a breakpoint at the first statement following the first group of do statements in initialize_Rankings. The command to do this is:

    `At 202 list (rank(1,1),rank(1,8))`

    This sets a breakpoint at line (or statement) number and displays the value of rank(1,1) and rank(1,8).

    Now type:

    `GOTO 196`

    This command resumes execution at statement 196, where statement 196 is in a currently active block. If there is no breakpoint at statement 196, Debug Tool will run from statement 196 until a breakpoint (statement 202) is hit.

    This is done to run from the start of the first group of do statements, as shown in Figure 13-29.

```
PL/I     LOCATION: 'CHABERT.PLI.SOURCE(KNIGHT)' :> 202
Command ===> _                                                Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 4
******************************** TOP OF MONITOR ********************************
0001  1 RANK ( 1, 1 )        8
0002    RANK ( 1, 8 )        8
0003  2 RANK ( 8, 1 )        8
0004    RANK ( 8, 8 )        8
****************************** BOTTOM OF MONITOR *******************************


SOURCE: CHABERT.PLI.SOURCE(KNIGHT) ---3----+----4----+----5--- LINE: 200 OF 214
     200      end;                                                        016
     201                                                                  ...
     202      do r = 1 to size;                                           001
     203        do c = 1 to size;                                         008
     204          do k = 1 to 8;                                          064
     205            if rank(r+rowjump(k),c+coljump(k)) = 0 then           512
     206              rank(r,c) = rank(r,c) - 1;                          176
     207            else;                                                 336
     208          end;                                                    512
LOG 0----+----1----+----2----+----3----+----4----+----5----+-- LINE: 189 OF 189
PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEAR
PF  7:UP         8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEVE
                                                                      02/015
```

Figure 13-29 Debug Tool - PLI - MFI Rank at the end of the first DO loops in initialize_Rankingd

11. If you examine the full array with a list rank, you can see that these values are correct. So, as shown in Figure 13-30, you can clear this breakpoint with the command:

```
CLEAR AT 202
```

```
PL/I     LOCATION: 'CHABERT.PLI.SOURCE(KNIGHT)' :> 202
Command ===> _                                                Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 4
******************************** TOP OF MONITOR ********************************
0001  1 RANK ( 1, 1 )        8
0002    RANK ( 1, 8 )        8
0003  2 RANK ( 8, 1 )        8
0004    RANK ( 8, 8 )        8
****************************** BOTTOM OF MONITOR *******************************


SOURCE: CHABERT.PLI.SOURCE(KNIGHT) ---3----+----4----+----5--- LINE: 200 OF 214
     200      end;                                                        016
     201                                                                  ...
     202      do r = 1 to size;                                           001
     203        do c = 1 to size;                                         008
     204          do k = 1 to 8;                                          064
     205            if rank(r+rowjump(k),c+coljump(k)) = 0 then           512
     206              rank(r,c) = rank(r,c) - 1;                          176
     207            else;                                                 336
     208          end;                                                    512
LOG 0----+----1----+----2----+----3----+----4----+----5----+-- LINE: 190 OF 190
PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEAR
PF  7:UP         8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEVE
                                                                      02/015
```

Figure 13-30 Debug Tool - PLI - MFI After removing the breakpoint

12. This means the problem must be in the second group of do statements. This group of do statements sets the rankings for each square in the board. Consider the loop shown in Example 13-6.

Example 13-6 DO statements to set the ranking of each square in the board

```
do r = 1 to size;
  do c = 1 to size;
    do k = 1 to 8;
      if rank(r+rowjump(k),c+coljump(k)) = 0 then
        rank(r,c) = rank(r,c) - 1;
      else;
    end;
  end;
end;
```

a. You need to do two things. The first is to stop once the value for rank(1,1) has been computed. To do this, put a breakpoint at line 209 and list rank(1,1) at that point. The command for that is:

```
AT 209 list rank(1,1);
```

b. Next you must get control when the if statement at line 205 is true. This involves an AT PATH breakpoint. Whenever a path point has been reached, the value of %PATHCODE contains an integer value that identifies the kind of change occurring when the path of program execution has reached a point of discontinuity and the path condition is raised.

   Table 13-2 shows the possible values for the Debug Tool variable %PATHCODE when the current programming language is PL/I.

Table 13-2 %PATHCODE values for PL/I

PATHCODE value	Meaning
0	An attention interrupt occurred.
1	A block has been entered.
2	A block is about to be exited.
3	Control has reached a label constant.
4	Control is being sent somewhere else as the result of a CALL or a function reference.
5	Control is returning from a CALL invocation or a function reference. Register 15, if it contains a return code, has not yet been stored.
6	Some logic contained in a complex DO statement is about to be executed.
7	The logic following an IF..THEN is about to be executed.
8	The logic following an ELSE is about to be executed.
9	The logic following a WHEN within a select-group is about to be executed.
10	The logic following an OTHERWISE within a select-group is about to be executed.

c. You only want to stop if it is a true if clause, so the command is:

```
AT PATH if %pathcode ^=7 then go
```

d. Press (PF9), **go**, and the program stops at line 205 as shown in Figure 13-31.

```
PL/I     LOCATION: 'CHABERT.PLI.SOURCE(KNIGHT)' :> 206
Command ===> _                                                Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 0 OF 0
******************************** TOP OF MONITOR ********************************
****************************** BOTTOM OF MONITOR *******************************


SOURCE: CHABERT.PLI.SOURCE(KNIGHT) ---3----+----4----+----5--- LINE: 202 OF 214
     202      do r = 1 to size;                                          002
     203        do c = 1 to size;                                        009
     204          do k = 1 to 8;                                         065
     205            if rank(r+rowjump(k),c+coljump(k)) = 0 then          513
     206              rank(r,c) = rank(r,c) - 1;                         176
     207            else;                                                336
     208          end;                                                   512
     209        end;                                                     064
     210      end;                                                       008
LOG 0----+----1----+----2----+----3----+----4----+----5----+-- LINE: 336 OF 356
0336  AT PATH
0337    IF %PATHCODE ^= 7 THEN
0338      GO ;
0339  GO ;
PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEAR
PF  7:UP         8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEVE
                                                                     02/015
```

Figure 13-31 Debug Tool - PLI - MFI At the path breakpoint

13. Every time the program stops at 205 you want to list the decrements. You need to monitor (1+coljump(K) and 1+rowjump(K), so issue the following commands:

    ```
    clear monitor 1
    clear monitor 2
    ```

    To clear all the corners from the monitoring window:

    ```
    monitor list (1+coljump(k), 1+rowjump(k))
    ```

    Press PF9 to go until you hit the breakpoint at 209.

14. If you look at row and column information in the LOG window, you can see that it was decremented twice for the move to rank(3,0). The rowjump or coljump arrays must be wrong:

    ```
    dcl rowjump(8)          fixed bin(15)
                            static init(-2,-1,+2,-1,+1,+1,+2,-2);
    dcl coljump(8)          fixed bin(15)
                            static init(+1,+2,-1,-2,+2,-2,-1,-1);
    ```

    You can see that the 3rd and 7th elements are the same. Looking at the number pattern you can see that you can correct this discrepancy by setting coljump(7) to +1.

    a. Do this in Debug Tool by issuing the command:

       ```
       COLJUMP(7)=1;
       ```

    b. Clear the PATH and 209 breakpoints:

       ```
       CLEAR AT PATH;
       CLEAR AT 209;
       ```

    c. Rerun the body of initialize_Rankings again with:

       ```
       GOTO 196;
       ```

    d. It should stop at the EXIT breakpoint of initialize_Rankings. At this point, check the corner values as shown in Figure 13-32:

       ```
       LIST (rank(1,1),rank(1,8),rank(8,1),rank(8,8))
       ```

```
PL/I     LOCATION: 'CHABERT.PLI.SOURCE(KNIGHT)' :> INITIALIZE_RANKINGS EXIT
Command ===> _                                                Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 0 OF 0
****************************** TOP OF MONITOR ******************************
****************************** BOTTOM OF MONITOR ***************************


SOURCE: CHABERT.PLI.SOURCE(KNIGHT) ---3----+----4----+----5--- LINE: 210 OF 214
     210      end;                                                          24
     211                                                                    ..
     212    end;                                                            03
     213                                                                    ..
     214  end knight;                                                       00
****************************** BOTTOM OF SOURCE ****************************
LOG 0----+----1----+----2----+----3----+----4----+----5----+ LINE: 1079 OF 1085
1079          breakpoint has not been established.
1080  GOTO 196 ;
1081  LIST ( RANK ( 1, 1 ), RANK ( 1, 8 ), RANK ( 8, 1 ), RANK ( 8, 8 ) ) ;
1082 EQA1141I RANK ( 1, 1 ) =       2
1083 EQA1141I RANK ( 1, 8 ) =       2
1084 EQA1141I RANK ( 8, 1 ) =       2
1085 EQA1141I RANK ( 8, 8 ) =       2
PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEAR
PF  7:UP         8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEVE
                                                                      02/015
```

Figure 13-32 Debug Tool - PLI - MFI The corners after the fix

15. Quit debugging and let the program continue until it is finished using the new command:

    `QUIT DEBUG;`

    As you can see in Figure 13-33, the results are better, but there is still another bug.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ---------------------------------------------------------------------------
EDIT       CHABERT.DEBUG.PRINT                          Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR_
****** ***************************** Top of Data ******************************
==MSG> -Warning- The UNDO command is not available until you change
==MSG>           your edit profile using the command RECOVERY ON.
000001 1    5   60   21   62    9   54    0    0
000002     16   23   12   59    0    0   28   53
000003      0   20   63   22   27    0    0    0
000004      0   13    0    0    0   39   50   33
000005      0    0    0   64   49   42    0    0
000006      0    0    0    0    0    0   40   47
000007      0    0    0    0   41    0    0    0
000008      0    0    0    0    0   37   46    0
****** ***************************** Bottom of Data ***************************
                                                                      04/015
```

Figure 13-33 Debug Tool - PLI - MFI The results after the first fix

13.2.3 Debugging a PL/I application with the remote interface

To show off the remote debugger, we describe how to finish debugging this problem with the IBM Distributed Debugger.

1. Start the distributed debugger listening using the command:

```
C:\IBMDebug\bin\idebug.exe -qdaemon -quiport=8001
```

This results in the screen shown in Figure 13-34.

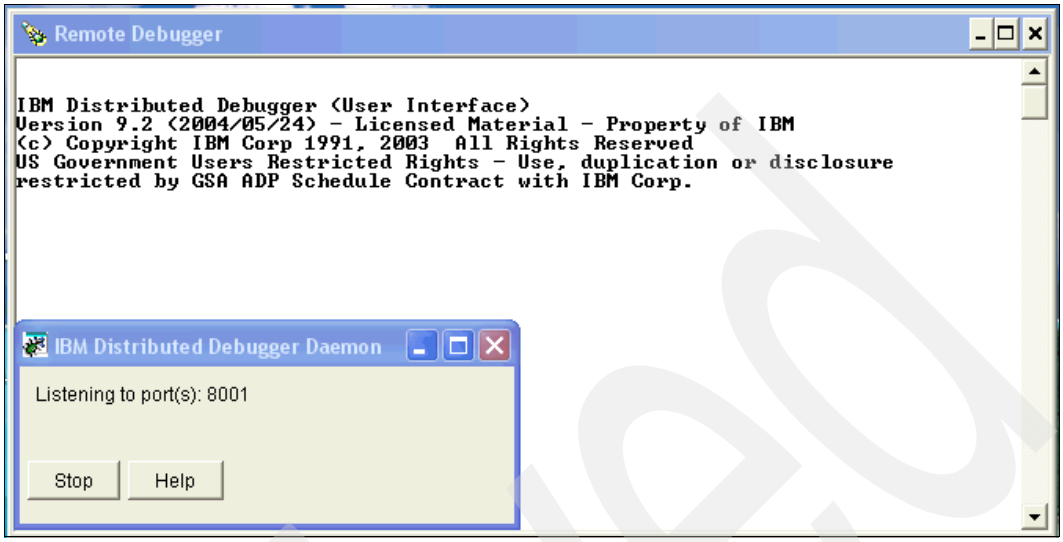


Figure 13-34 Remote debugger initialization

2. Using the **Ipconfig** command from a Windows Command prompt window, display your IP-address as shown in Figure 13-35.

```
Ethernet adapter Local Area Connection (LAN):

        Connection-specific DNS Suffix  . : svl.ibm.com
        IP Address. . . . . . . . . . . . : 9.30.60.207
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 9.30.60.1
```

Figure 13-35 IP Address

3. The new intermediate PL/I program is called KNIGHTM. Invoke the remote debugger to debug it using the job shown in Example 13-7.

Example 13-7 JCL to start KNIGHT using the IBM Distributed Debugger

```
//DTBATCH  JOB 'REM DEBUG',
// CLASS=A,MSGCLASS=X,MSGLEVEL=(1,1),
// NOTIFY=&SYSUID,
// REGION=32M
//*
// EXEC PGM=KNIGHT,
// PARM='TEST(,,,''VADTCPIP&9.30.60.207%8001:*'') / 8'
//INSPPREF DD DSN=CHABERT.DEBUG.PREF(KNIGHT),DISP=(SHR,,)
//REPOUT DD SYSOUT=*
//STEPLIB DD DSN=CHABERT.PLI.LOAD,DISP=(SHR,,)
// DD DSN=ADTOOLS.DEBUG.TOOL51.SEQAMOD,DISP=(SHR,,)
//SYSABEND DD SYSOUT=*
//SYSPRINT DD DSN=CHABERT.DEBUG.PRINT,DISP=(SHR,,)
//TRANREP DD SYSOUT=*
```

At this point the remote debugger is displayed as shown in Figure 13-36.

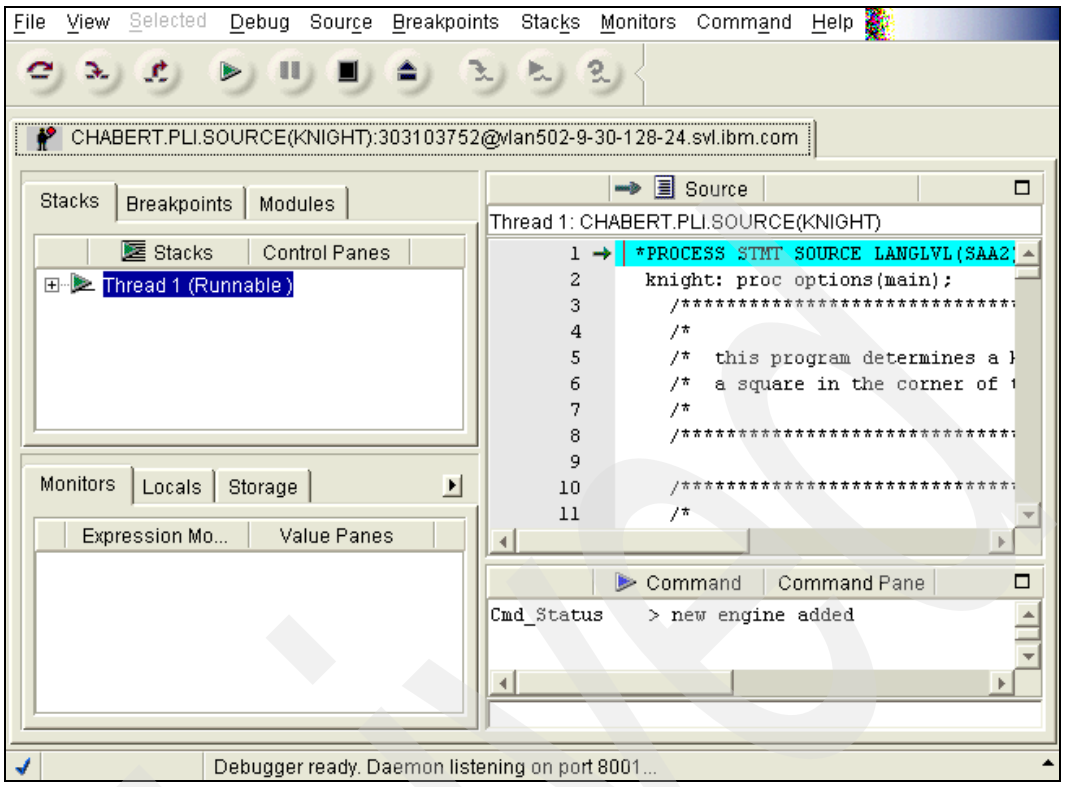


Figure 13-36 Debug Tool - PLI - Remote debugger

4. Place a breakpoint at the beginning of the actual program, as shown in Figure 13-37.

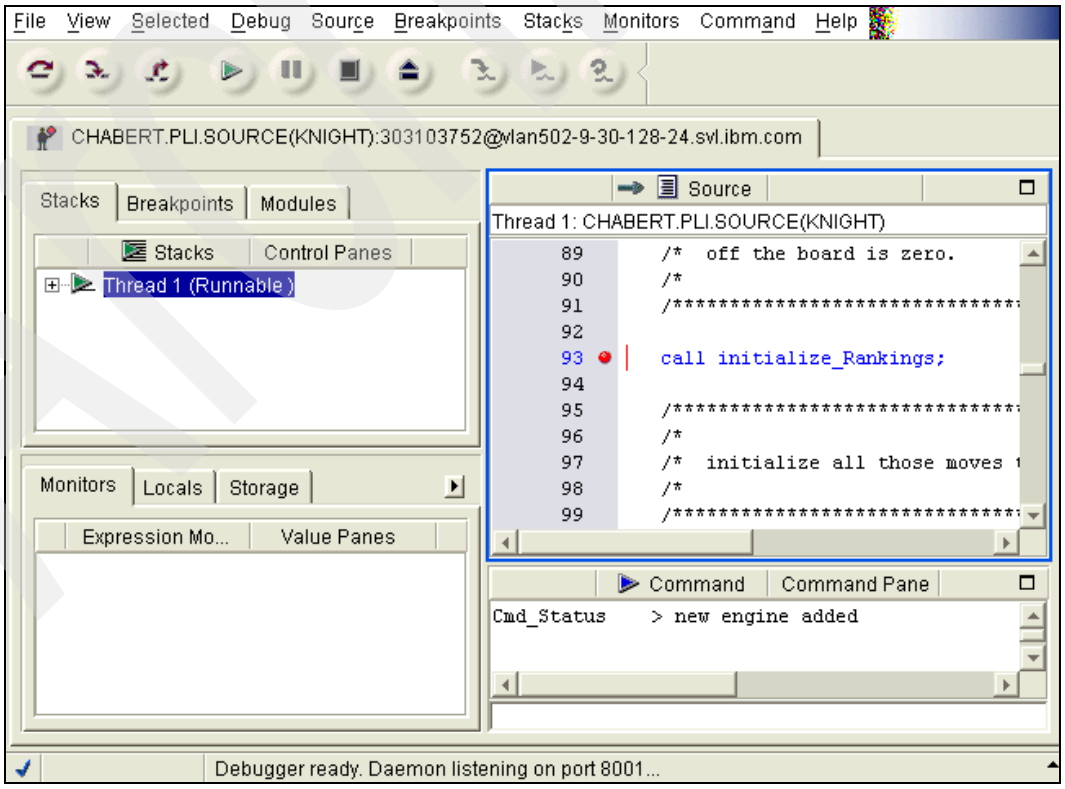


Figure 13-37 Debug Tool - PLI - Remote debugger - After setting breakpoint

5. Click the button pointed to by the cursor in the following figure; the program runs to the breakpoint that you set previously, as shown in Figure 13-38.

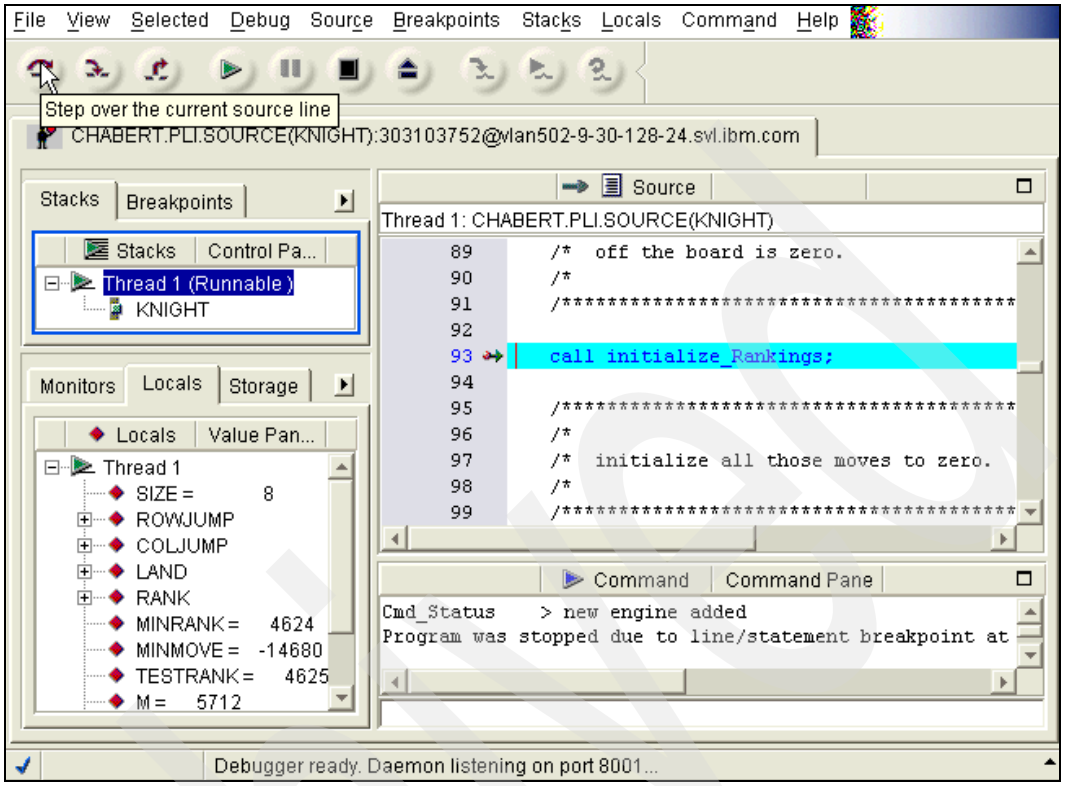


Figure 13-38 Debug Tool - PLI - Remote debugger display at our breakpoint

6. At the top of the window are controls that allow you to manage the execution of your procedure, step-by-step if you want, then the screen is divided into four parts:

   a. The top left part is shown in Figure 13-39. This area displays Stacks, Breakpoints, and Modules. A sample of each is shown in the figure.

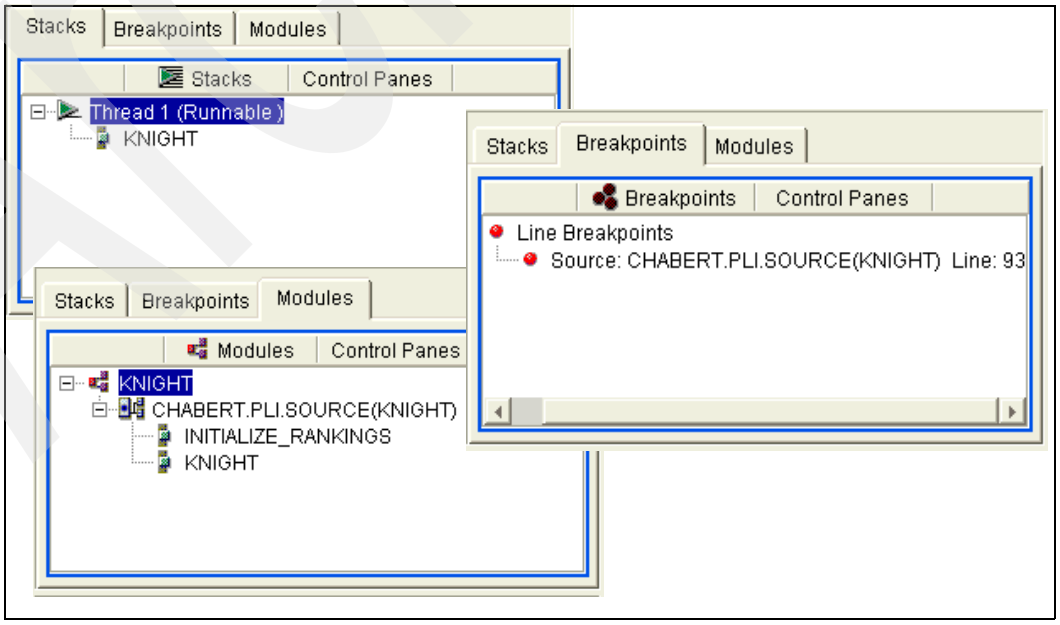


Figure 13-39 Debug Tool - PLI - Remote debugger - Top left part of the screen

b. In the source code part of the main window, an arrow shows the current statement being executed as shown in Figure 13-40.

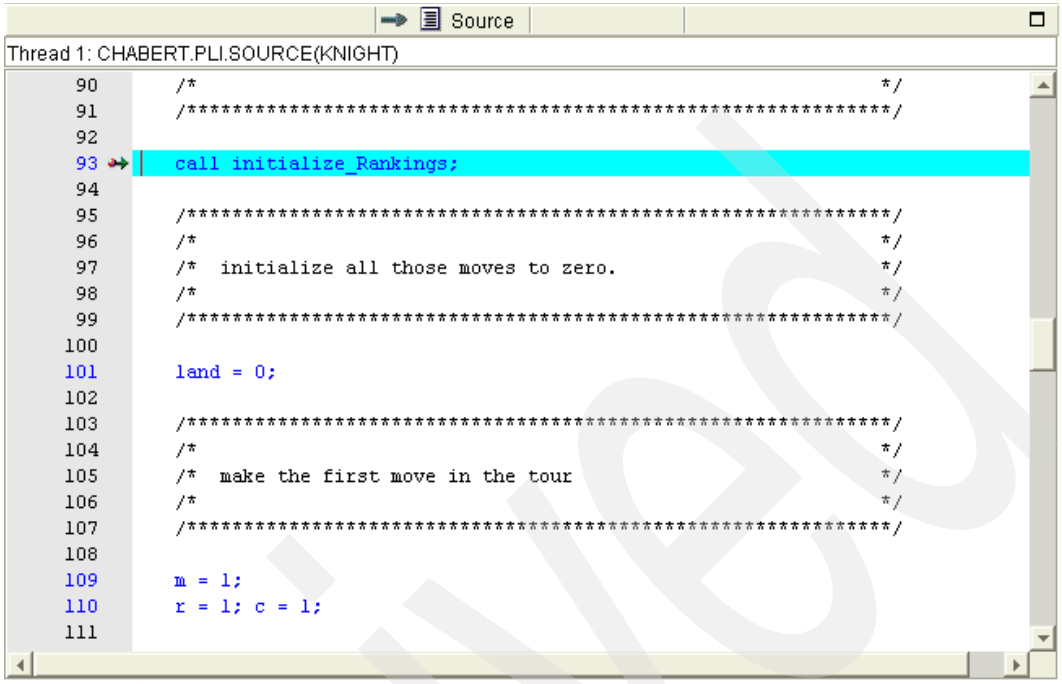


Figure 13-40 Debug Tool - PLI - Remote debugger - Source part

You can also set breakpoints in your source code, indicated by a red dot next to the line number. To set breakpoints, all you have to do is double-click next to the line number and the breakpoint is set. You can only set breakpoints in lines that actually execute some code, so you will not be able to set breakpoints in lines with comments, for example.

c. In the monitor part of the main window, you can monitor and change values of variables and parameters. To start monitoring the values of a variable, click **Monitor** → **Monitor Expression** as shown in Figure 13-41.

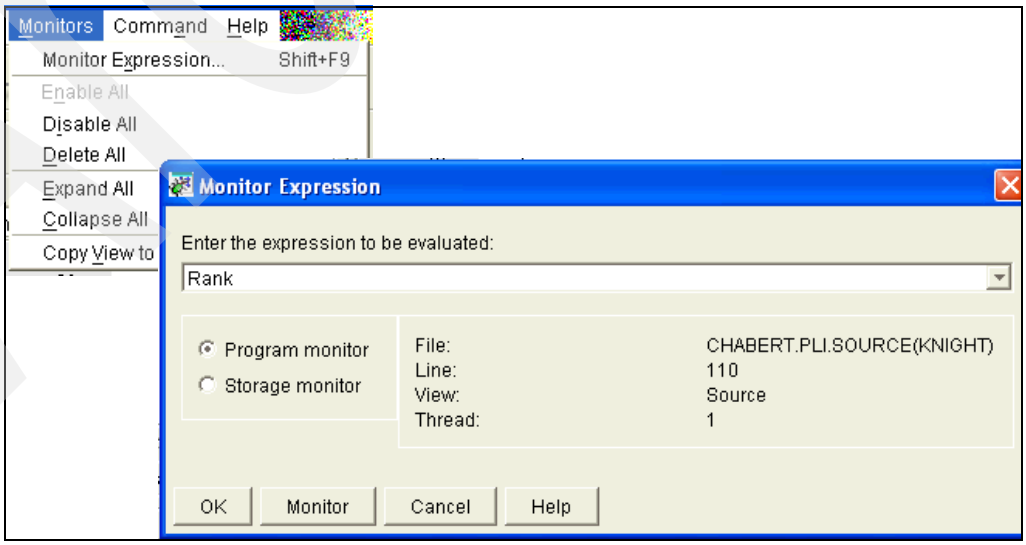


Figure 13-41 Debug Tool - PLI - Remote debugger - Monitor definition

The monitor part displays the Rank expression as well as other information. Figure 13-42 shows several of the windows included in the monitor portion of the screen.

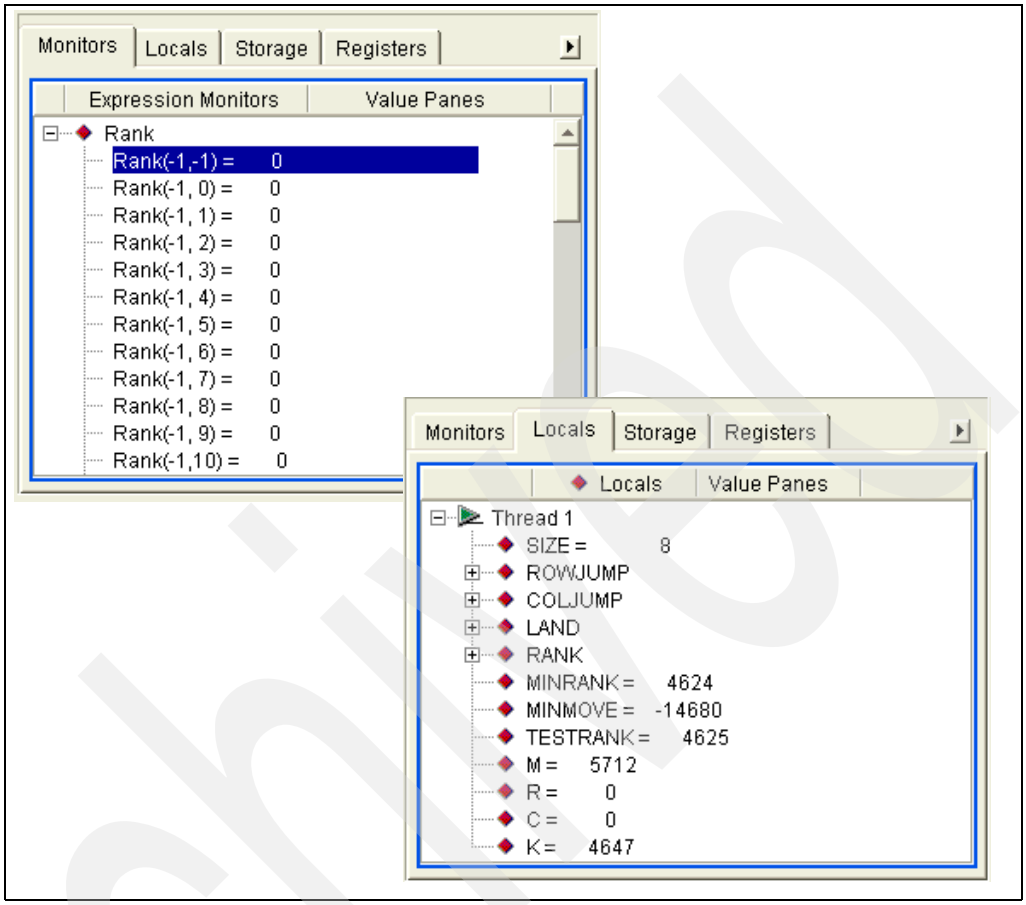


Figure 13-42 Debug Tool - PLI - Remote debugger - Monitor and Locals windows

d. The last part is a log window as shown in Figure 13-43.

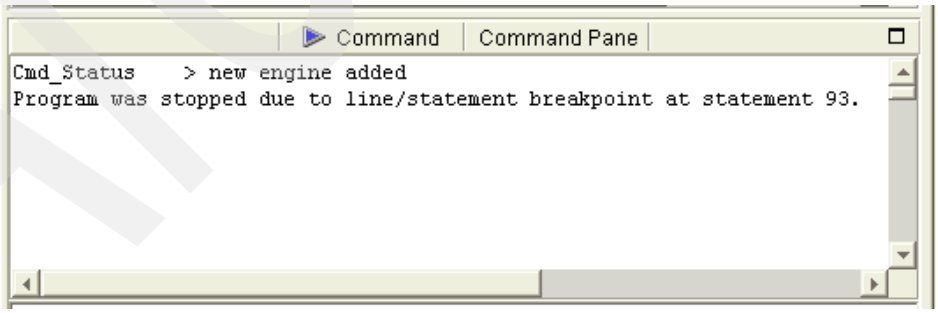


Figure 13-43 Debug Tool -PLI - Remote debugger - Log window

7. The land array is the array that will hold the results (the final path across the chess board), and you would like to monitor the changes in the land array. Go to the **Breakpoints** menu and select the **Set Storage Change** entry as shown in Figure 13-44.

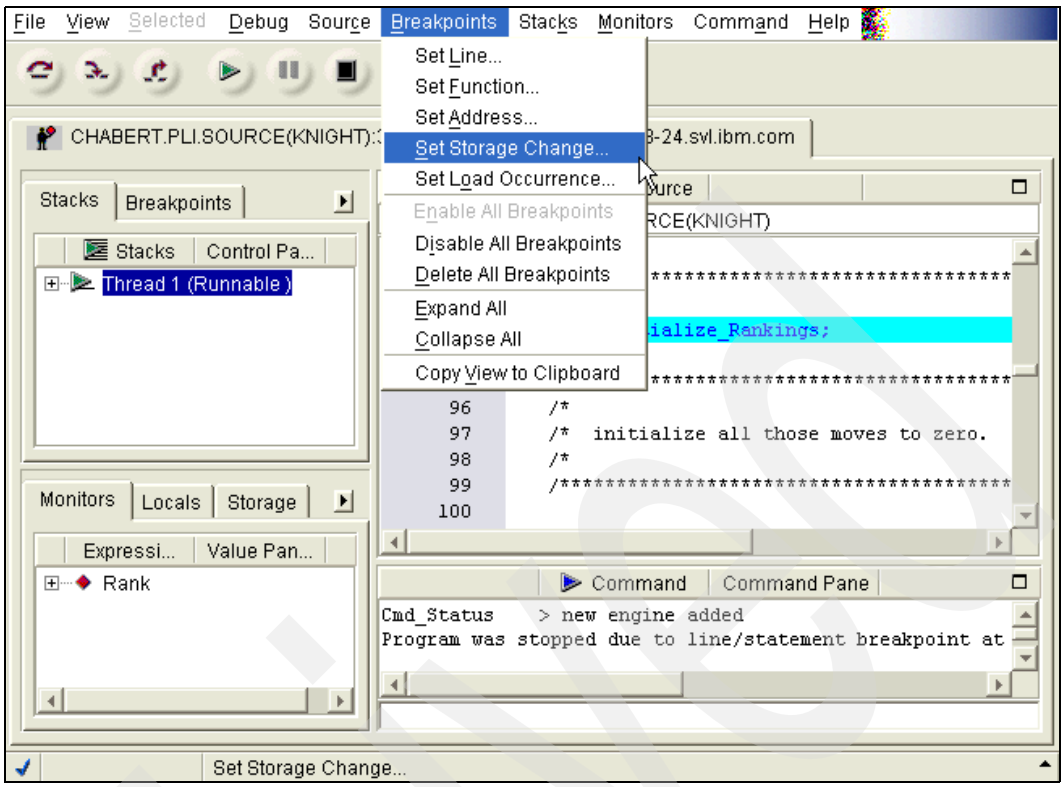


Figure 13-44 Debug Tool - PLI - Remote debugger - Setting a storage change breakpoint

8. Set a storage change breakpoint on the first 128 bytes of land as shown in Figure 13-45.

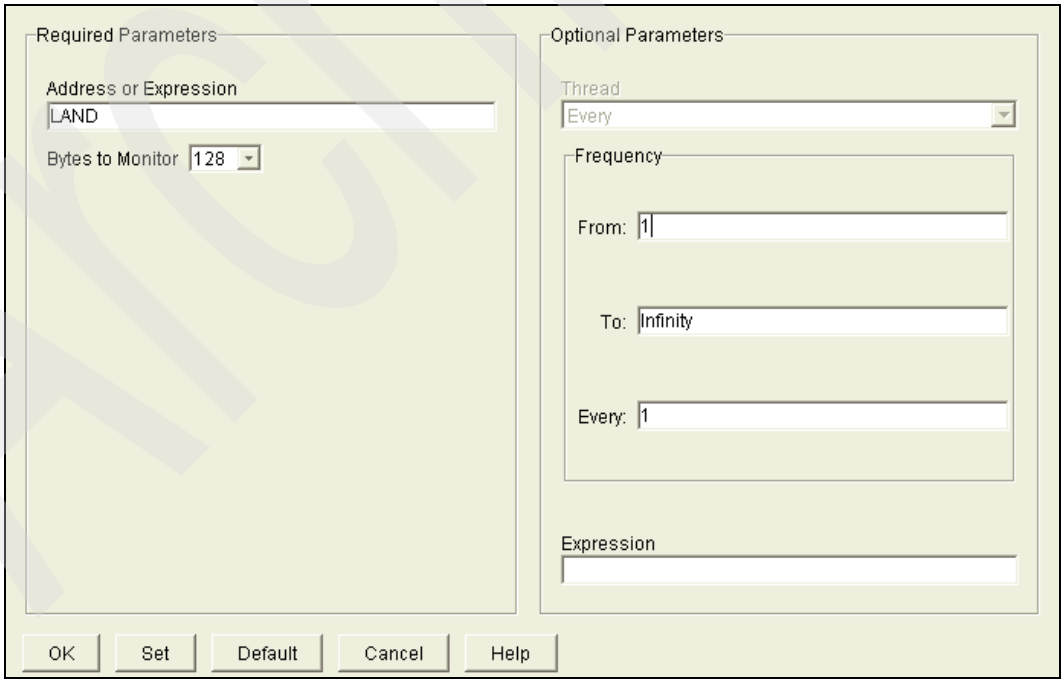


Figure 13-45 Debug Tool - PLI - Remote debugger - Setting breakpoint on first 128 bytes of land

9. You could enter an expression to allow you to monitor the last 128 bytes. But at this point that may not be necessary. Click the green play arrow to see what is going to happen, as shown in Figure 13-46.

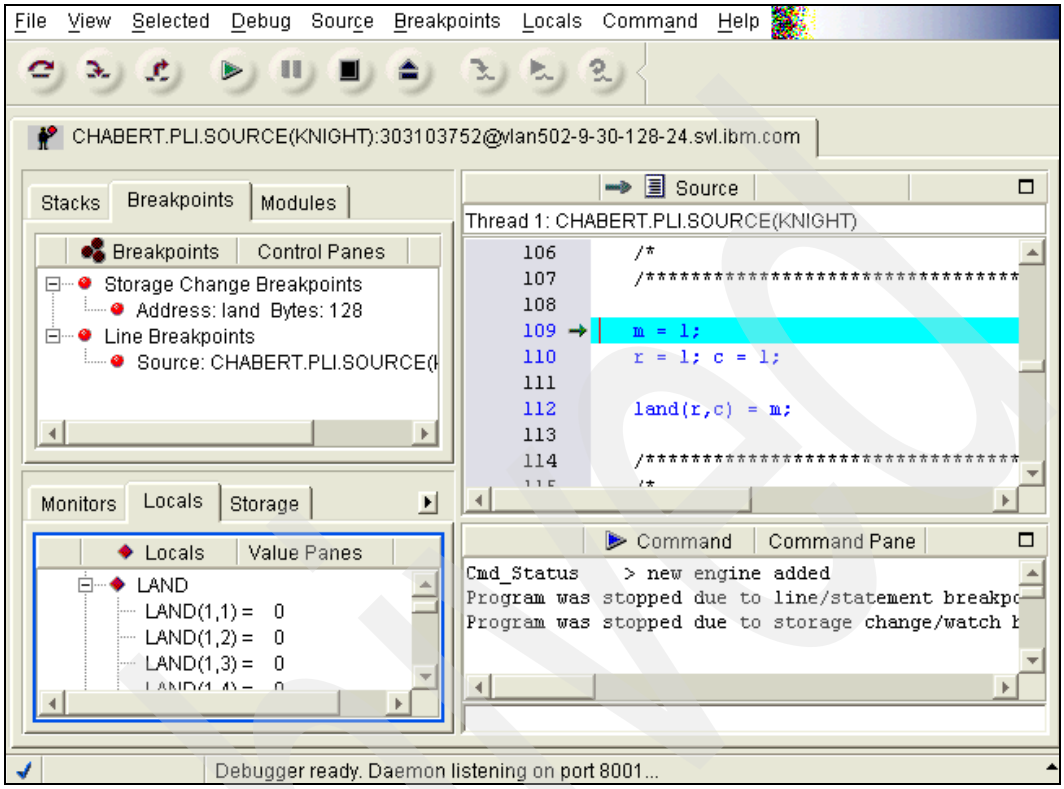


Figure 13-46 Debug Tool - PLI - Remote debugger - First storage change stop, after the allocation

The breakpoint window shows your two breakpoints. The log window indicates that the program was stopped due to a storage change/watch breakpoint on land for 128 bytes.

10. Click play again as shown in Figure 13-47.

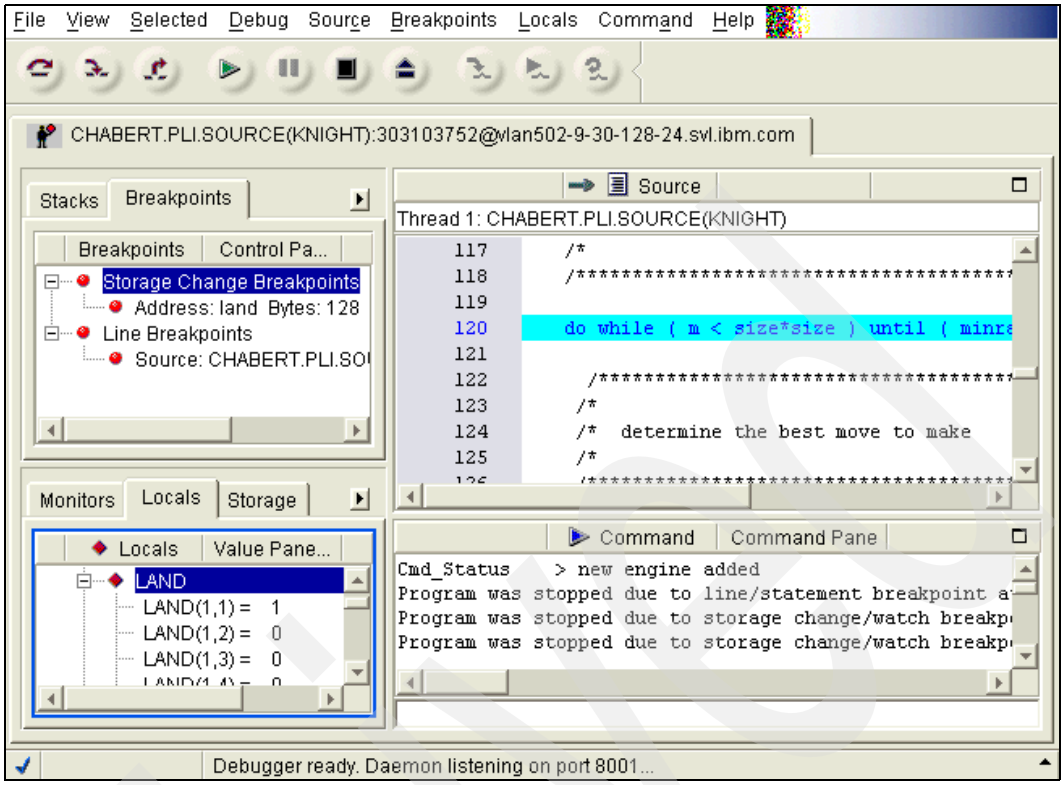


Figure 13-47 Debug Tool - PLI - Remote debugger - At the 2nd storage change breakpoint

11. The program stops when land(1,1) is modified and set to 1.

    Click play again and see that land(2,3) has been set to 2.

    Click play a third time; the result is shown in Figure 13-48.

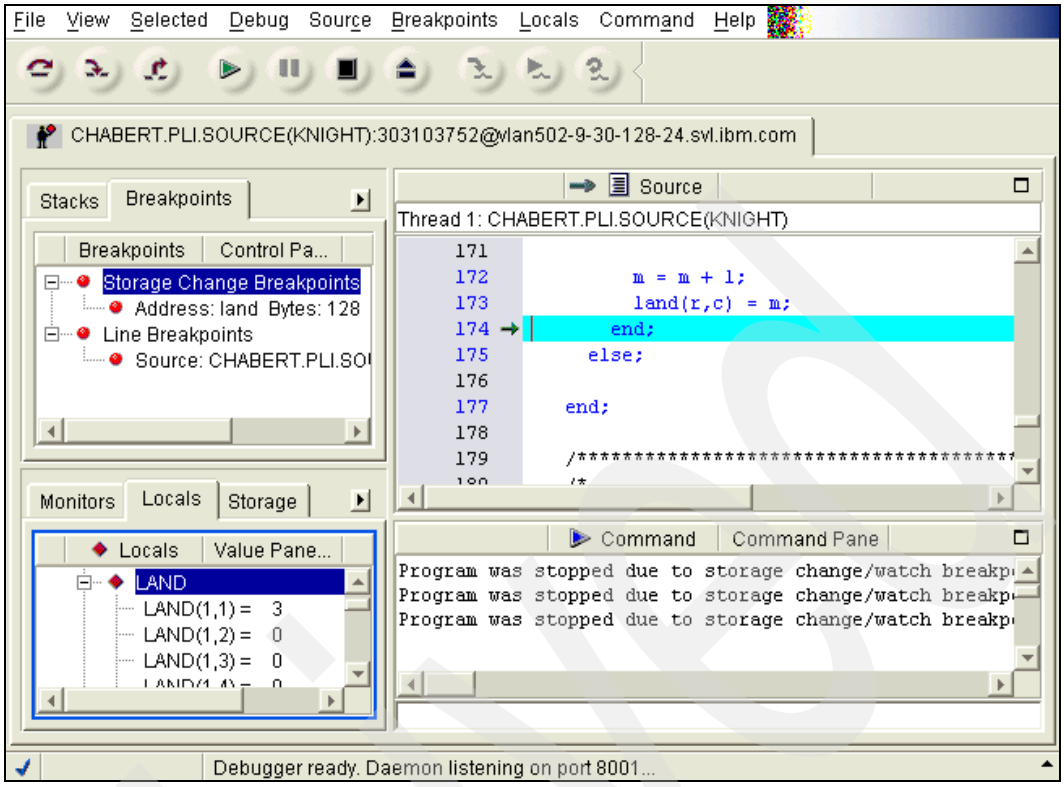


Figure 13-48 Debug Tool - PLI - Remote debugger - At the 3rd storage overwrite

Note that the land(1,1) entry has been rewritten. Obviously there must be something wrong with rank(1,1). Take a look by compressing land in the locals window, and selecting rank as shown in Figure 13-49.

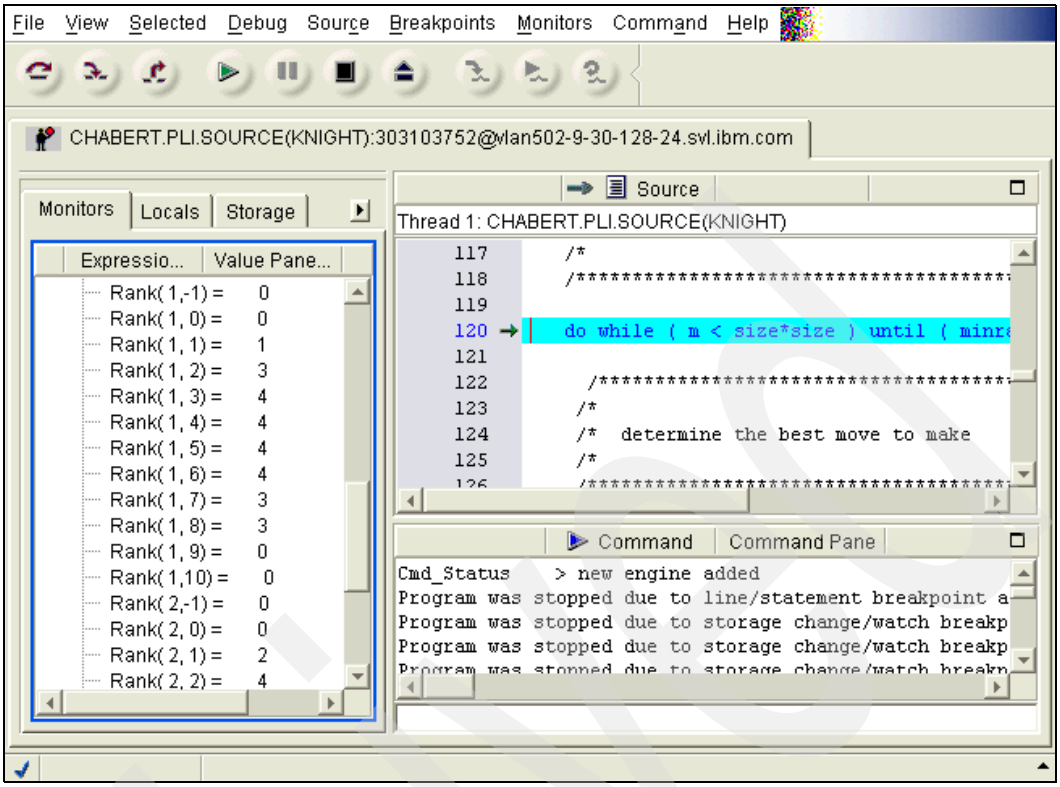


Figure 13-49 Debug Tool - PLI - Remote debugger - Looking at the value of rank(1,1)

12. Remember that a 0 value at a rank position indicates a position the knight should not go. However, rank(1,1) has a value of 1. The problem is that the value of rank is not being set to 0 once it has been used as a landing point in the path. Modify the code to add the following statement at line 169:

```
rank(r,c) = 0;
```

This means that used squares are marked out of bounds and will not be selected, as shown in Example 13-8.

Example 13-8 Final code change to fix the knight code

```
        else;
      end;
      /*********************************************************/
      /*  make the move                                        */
      /*                                                       */
      /*********************************************************/

      rank(r,c) = 0;
      r = r + rowjump(minmove);
      c = c + coljump(minmove);

      m = m + 1;
      land(r,c) = m;
    end;
  else;

end;
```

A run of the fixed program gives you the result shown in Figure 13-50. Note how easy it is to follow the knight's path from the upper left-hand square all over the board.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------------
EDIT       CHABERT.DEBUG.PRINT                                Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
****** ***************************** Top of Data ******************************
000001 1     1   16   27   22    3   18   47   56
000002      26   23    2   17   46   57    4   19
000003      15   28   25   64   21   48   55   60
000004      24   35   30   45   58   61   20    5
000005      29   14   63   34   49   44   59   54
000006      36   31   38   41   62   53    6    9
000007      13   40   33   50   11    8   43   52
000008      32   37   12   39   42   51   10    7
****** **************************** Bottom of Data ****************************
                                                                    04/015
```

Figure 13-50 Debug Tool - PLI - Output from the working program

13.3 Debugging COBOL programs using Debug Tool

This section describes how to:

- Prepare an OS/VS COBOL application for debugging using JCL
- Prepare a COBOL for z/OS and OS/390 application for debugging using Debug Tool Utilities and Advanced Functions (DTU&AF).
- Debug a COBOL for z/OS and OS/390 application using Debug Tool Set Utility (DTSU) to start the mainframe interface (MFI)
- Debug a COBOL for z/OS and OS/390 application using Debug Tool Set Utility (DTSU) to start the remote debugger interface.

13.3.1 Preparing an OS/VS COBOL application for debugging using JCL

When you debug an OS/VS program, you can use most of the Debug Tool commands. Debugging an OS/VS COBOL program is much like debugging an assembler program. However, there are differences between debugging an OS/VS COBOL or assembler program and debugging programs written in other programming languages supported by Debug Tool. The main considerations and differences are the following:

- After you compile your program, you must create a debug information file, also called the EQALANGX file. Debug Tool uses this file to obtain information about your OS/VS COBOL program.
- You must inform Debug Tool that a compile unit is an OS/VS COBOL compile unit and instruct Debug Tool to load the OS/VS COBOL compile unit's debug information. Do this by entering the LOADDEBUGDATA (or LDD) command.

- OS/VS COBOL expressions must be enclosed in single quotes. In addition, only a subset of the OS/VS COBOL operators and figurative constants are supported by Debug Tool.

Note: OS/VS COBOL is part of the Debug Tool Utilities and Advanced Functions.

Program preparation

The following compiler options are required:

- SOURCE
- DMAP
- PMAP
- VERB
- XREF

You cannot have LSTONLY or LSTCOMP.

EQALANGX generation

The process of generating the EQALANGX file for OS/VS COBOL is the same as generating the EQALANGX for assembler with two exceptions:

1. The PARMs are different. The required PARMs for OS/VS COBOL are:

   ```
   PARM='(COBOL LOUD ERROR 64K CREF'
   ```

2. Instead of using the SYSADATA output of the assembler, the input to EQALANGX is the OS/VS COBOL listing.

Example 13-9 is sample JCL that can be used to generate an EQALANGX file for OS/VS COBOL.

Example 13-9 EQALANGX generation

```
//IDILANGX EXEC PGM=EQALANGX,REGION=32M,
// PARM='(COBOL LOUD ERROR 64K CREF'
//STEPLIB  DD  DISP=SHR,DSN=ADTOOLS.DEBUG.TOOL51.SEQAMOD
//LISTING  DD  DISP=SHR,DSN=ADTOOLS.COBOSVS.LISTING(&MEM.)
//IDILANGX DD  DISP=SHR,DSN=ADTOOLS.DEBUG.TOOL51.OSVS.EQALANX
```

The variables used in this example have the following meanings:

(COBOL Indicates that an OS/VS COBOL module is being processed.

LOUD This parameter is required.

ERROR This parameter is optional. If you specify it, additional information is displayed when an error is detected.

64K This parameter is required.

CREF This parameter is required.

hlq.EQALANGX The name of the data set where the EQALANGX debug file is to be placed. This data set must have variable block record format (RECFM=VB) and a logical record length of 1562 (LRECL=1562).

Debug Tool searches for the EQALANGX debug file in a partitioned data set with the name hlq.EQALANGX and a member name that matches the name of the OS/VS COBOL program. If you want the member name of the EQALANGX debug file to match the name of the OS/VS COBOL program, you do not need to specify a member name on the DD statement.

EQALANGX usage

Before you can debug an OS/VS COBOL program, the compilation unit (CU) must meet the following requirements:

- If the CU is known to Debug Tool, it must be a disassembly CU.
- If the CU is not known to Debug Tool, it must appear later as a disassembly CU.

If the CU is part of a load module that has not yet been loaded when you issue the LDD command, Debug Tool will issue a message indicating that the CU is not found and that the LDD has been deferred. If the CU later appears as a disassembly CU, the LDD will be done at that time.

In either case, you must define the CU as an OS/VS COBOL CU and load the debug data that is associated with that program. To accomplish these objectives, use the LOADDEBUGDATA command (abbreviated as LDD) as follows:

- If your debug data is in a partitioned data set where the high-level qualifier is the current user ID, the low-level qualifier is EQALANGX, and the member name is the same as the name of the CU that you want to debug, enter the following command:

  ```
  LDD membername
  ```

- If your debug data is in a different partitioned data set than userid.EQALANGX but the member name is the same as the name of the CU that you want to debug, enter the following command before or after you enter LDD membername:

  ```
  SET DEFAULT LISTINGS
  ```

- If your debug data is in a sequential data set or is a member of a partitioned data set but the member name is different from the CU name, enter the following command before or after you enter LDD membername:

  ```
  SET SOURCE
  ```

LinkEdit

Example 13-10 is sample JCL that link-edits an OS/VS COBOL program with the OS/VS COBOL run-time.

Example 13-10 JCL to link-edit an OS/VS COBOL program with OS/VS COBOL run-time

```
//OSCOBLNK JOB (USER01),'USER01',MSGCLASS=H,MSGLEVEL=(1,1),
//            NOTIFY=&SYSUID,REGION=32M,CLASS=A,TIME=1
//*
//* JCL to link edit Cobol testcase COBO2O. COBO2O was compiled with
//* the OS/VS Cobol compiler.
//*
//* Link edit library   : OS/VS Cobol
//* Input  object       : ADTOOLS.DEBUG.TOOL51.OSVS.OBJ
//* Output RUNLIB module: ADTOOLS.DEBUG.TOOL51.OSVS.LOAD
//*
//LKED   EXEC PGM=IEWL,PARM=(LET,MAP,LIST)
//SYSLIB   DD DISP=SHR,DSN=ESFLINT.OSVSCOB.VSCLLIB
//SYSPRINT DD SYSOUT=*
//SYSLMOD  DD DISP=SHR,DSN=ADTOOLS.DEBUG.TOOL51.OSVS.LOAD
//SYSUT1   DD UNIT=SYSDA,SPACE=(TRK,(10,10))
//OBJECT   DD DISP=OLD,DSN=ADTOOLS.DEBUG.TOOL51.OSVS.OBJ
//SYSLIN   DD *
 INCLUDE OBJECT(COBO2O)
 NAME COBO2O(R)
/*
//
```

Example 13-11 is sample JCL that link-edits an OS/VS COBOL program with the LE run-time.

Example 13-11 JCL to link-edit an OS/VS COBOL program with the LE run-time

```
//OSCOBLNK JOB (USER01),'USER01',MSGCLASS=H,MSGLEVEL=(1,1),
//             NOTIFY=&SYSUID,REGION=32M,CLASS=A,TIME=1
//*
//* JCL to link edit Cobol testcase COB020. COB020 was compiled with
//* the OS/VS Cobol compiler.
//*
//* Link edit library   : OS/VS Cobol
//* Input  object       : ADTOOLS.DEBUG.TOOL51.OSVS.OBJ
//* Output RUNLIB module: ADTOOLS.DEBUG.TOOL51.OSVS.LOAD
//*
//LKED   EXEC PGM=IEWL,PARM=(LET,MAP,LIST)
//SYSLIB   DD DISP=SHR,DSN=ESFLINT.CEEV2R10.SCEELKED
//SYSPRINT DD SYSOUT=*
//SYSLMOD  DD DISP=SHR,DSN=ADTOOLS.DEBUG.TOOL51.OSVS.LOAD
//SYSUT1   DD UNIT=SYSDA,SPACE=(TRK,(10,10))
//OBJECT   DD DISP=OLD,DSN=ADTOOLS.DEBUG.TOOL51.OSVS.OBJ
//SYSLIN   DD *
 INCLUDE OBJECT(COB020)
 NAME COB020L(R)
/*
//
```

Execution

- If you link with the OS/VS COBOL run-time, your execution JCL should STEPLIB to ESFLINT.OSVSCOB.VSCLLIB.
- If you link with the LE run-time, your execution JCL should STEPLIB to ESFLINT.CEEV2R10.SCEERUN.

Syntax for OS/VS COBOL expressions

The syntax used for debugging OS/VS COBOL programs is a subset of OS/VS COBOL syntax and a subset of the syntax used for debugging assembler programs.

In general, whenever you enter an OS/VS COBOL expression (such as the operand of LIST expression, Assignment, IF, and so forth), you must enclose the OS/VS COBOL expression in single-quotes. For example:

```
LIST 'A-B IN C';
'A' = 'B';
IF 'A = 22' THEN …
```

There are some Debug Tool commands that can be used for debugging OS/VS COBOL programs that use the assembler syntax for OS/VS COBOL programs. For example, while debugging an OS/VS COBOL program you might use the following command:

```
STORAGE(X"1B4C0",3) = X"0102FC";
```

Restrictions on OS/VS COBOL expressions

In addition to the requirement that OS/VS COBOL expressions be enclosed in single quotes, the following restrictions apply to OS/VS COBOL expressions:

- The following operators are supported by Debug Tool in OS/VS COBOL expressions:
  - IN or OF

- Subscript/index
  In a subscript or index list, the subscript or index expressions must be separated by a comma. A space is not sufficient for separating subscript or index expressions.
- LENGTH OF
  Returns the length of a symbol.
- +, -, *, /
  Addition, Subtraction or prefix minus, Multiplication and Division
- //
  Remainder
- ||
  Concatenation
- (...)
  Parenthesis to control the order of operation, specify the subscript of an array, or select a substring.

  symbol(subscript,subscript,…) Parenthesis to specify a subscript or index for an array.

  symbol(substrstart:substrend) Parenthesis to select a substring of the bytes from substrstart to substrend from a character variable.

  symbol(substrstart::substrlen) Parenthesis to select a substring of substrlen bytes.
- &
  Logical “and” operation.
- |

  Logical “or” operation.
- =, <, >, <=, >=

- Lower-case letters are accepted in contexts other that non-numeric literals as a substitute for (and equivalent to) upper-case letters.
- The use of COBOL special registers such as DAY, DATE, TIME, and so forth is not supported in Debug Tool OS/VS COBOL expressions.
- All non-numeric literals must be enclosed in double quotes. Single quotes cannot be used.
- Only the following subset of figurative constants are supported in Debug Tool OS/VS COBOL expressions:
  - HIGH-VALUE, HIGH-VALUES
  - LOW-VALUE, LOW-VALUES
  - QUOTE, QUOTES
  - SPACE, SPACES
  - ZERO, ZEROES, ZEROS

Common symbol

Debug Tool implicitly defines the following symbols in all OS/VS COBOL compilation units:

- _STORAGE
  This symbol is implicitly defined as a symbol representing all of main memory.
- %symbol
  A valid Debug Tool variable or built-in function, for example, %ADDRESS or %HEX(expression).

Restrictions

- Limited Path Table, %PATHCODE, or AT PATH support (Entry and Label path points only)
- No AT CALL support
- No AT EXIT support
- No STEP RETURN support

Assignment command

The Assignment command assigns the value of an expression to a specified variable. It is the equivalent of the OS/VS COBOL COMPUTE statement. An example is shown in Example 13-12.

Example 13-12 Assignment

```
>>-- 'receiver' -- = -- 'sourceexpr'--; ><
```

receiver A valid Debug Tool OS/VS COBOL reference enclosed in single quotes.

sourceexpr A valid Debug Tool OS/VS COBOL expression enclosed in single quotes.

Usage:

1. When the assignment receiver is an arithmetic variable, the source can be a hexadecimal string of the same length as the receiver.
2. When the assignment receiver is a non-numeric string, the source can be a hexadecimal string of any length.
3. When the assignment receiver is a COBOL INDEX variable, the source is assumed to be a subscript value and is converted to the proper offset before the value is stored into the receiver.

Note: The Assignment command cannot be used while you replay recorded statements by using the PLAYBACK commands.

13.3.2 Preparing a COBOL for z/OS and OS/390 application using DTU&AF

Debug Tool Utilities is only available if you have installed Debug Tool Utilities and Advanced Functions. It has the features shown in Figure 13-51.

```
---------------------------- Debug Tool Utilities ----------------------------
Option ===> _____________________________________________________________________

                                                                   More:     +
0  Manage Job Card
   For Program Preparation and Setup File Management

1  Program Preparation
   Compile old or new COBOL programs with newer compilers, convert old COBOL
   source into new COBOL source, use other compilers, and link edit.

2  Manage and Use Debug Tool Setup Files
   You can manage setup files and use them to run your program interactively
   with Debug Tool in TSO Foreground or submit your program to run in
   the background using MVS batch.

3  Code Coverage
   Measure code coverage in programs written in COBOL, PL/I, C/C++ and
   Assembler when compiled with specific IBM compilers and HLASM.

4  Manage IMS Programs

                                                                     02/014
```

Figure 13-51 Debug Tool Utilities

0 - Manage Job Card

This feature is used for program preparation and to set up file management.

1 - Program Preparation

This feature is used to compile old or new COBOL programs with newer compilers, and convert old COBOL source into new COBOL source.

2 - Manage and Use Debug Tool Setup Files

This feature is used to manage setup files and use them to run your program interactively with Debug Tool in TSO foreground or submit your program to run in the background using MVS batch.

3 - Code Coverage

This feature is used to measure code coverage in programs written in COBOL, PL/I, C/C++ and Assembler when compiled with specific IBM compilers and HLASM.

Perform the following steps to compile and link with program preparation.

1. Select **1 - Program Preparation** from the main DTU panel; the screen shown in Figure 13-52 is displayed.

```
----------------------- Debug Tool Program Preparation -----------------------
Option ===> ____________________________________________________________________

                                                                   More:     +
1  COBOL Compile
   Using 5655-G53 IBM Enterprise COBOL for z/OS and OS/390

2  COBOL Convert and Compile
   Using 5648-B05 COBOL and CICS Command Level Conversion Aid
   and   5655-G53 IBM Enterprise COBOL for z/OS and OS/390

3  PL/I  Compile
   Using 5655-H31  IBM(R) Enterprise PL/I for z/OS

4  C and C++ Compile
   Using 5694A01 z/OS C/C++

5  Assemble
   Using High Level Assembler

L  Link Edit
   Using z/OS Binder

                                                                   02/014
```

Figure 13-52 Debug Tool Program Preparation

2. Select option **2 - COBOL Convert and Compile** from the Debug Tool Program Preparation panel; the screen shown in Figure 13-53 is displayed.

```
----------------Debug Tool Program Preparation - COBOL Compile----------------
Command ===> ___________________________________________________________________

Compile using 5655-G53 IBM Enterprise COBOL for z/OS and OS/390
F Foreground or Batch Processing  (F B)
                                                                   More:     +
Specify primary input data set for compilation.

Source Library:
   Project . . . CHABERT
   Group . . . . TRADER  . . . ________ . . . ________ . . . ________
   Type  . . . . SOURCE
   Member  . . . TR*_____        (Blank or pattern for member selection list)

Other Partitioned or Sequential Data Set:
   Data Set Name . . . ____________________________________________________
   Volume Serial . . . ______    (If not cataloged)

Test options, data set name patterns, CICS and SQL processing options:
_ Enter / to edit options and data set name patterns
N CICS    I Integrated CICS translator, S Separate CICS translator, N None
N DB2/SQL I Integrated SQL coprocessor, S Separate DB2 precompiler, N None

Compiler options (TEST compiler option is automatically added):
                                                                   13/022
```

Figure 13-53 Debug Tool Program Preparation - COBOL Compile

This compile utility lets you select either for a Foreground (F) () or Background (B) compile. In foreground compile the processing is run in TSO, and the return codes are displayed on a panel; in background compile a job is created and submitted in batch. You must specify (using FP8) the SYSLIB data set used for COPY statement processing to search for COPY members if required.

3. Specify your source library. If you do not specify the source member or if it is a generic value, a member selection list will be presented as shown in Figure 13-54.

```
  Menu  Functions  Confirm  Utilities  Help
 ------------------------------------------------------------------------
LIBRARY          CHABERT.TRADER.SOURCE                    Row 00001 of 00005
Command ===> _____________________________________________ Scroll ===> PAGE
   Name     Prompt          Size    Created          Changed            ID
_ TRADERB                    771   2002/10/17   2004/10/19 22:41:15    CHABERT
_ TRADERD                   1108   2002/09/26   2004/10/19 22:41:25    CHABERT
_ TRADERD3                  1108   2002/10/08   2004/10/19 22:41:32    CHABERT
_ TRADERI                   1033   2002/10/04   2004/10/19 22:41:41    CHABERT
_ TRADERI3                  1033   2002/10/08   2004/10/19 22:41:50    CHABERT
  **End**
                                                                  04/015
```

Figure 13-54 Library member

4. Select the member TRADERB and press Enter. The screen shown in Figure 13-55 is displayed listing all the input data sets (your source), final options, data set names objects, sidedeck, DB2 compiler, and CICS translator options and data sets. You may overwrite any or all of these fields.

```
-----------------------COBOL Compile - Verify Selections----------------------
Command ===> ____________________________________________________________________

Compile using 5655-G53 IBM Enterprise COBOL for z/OS and OS/390
                                                                  More:     +
Input data set: 'CHABERT.TRADER.SOURCE(TRADERB)'

Compiler options:
TEST(NONE,SYM,SEPARATE)
____________________________________________________________________________

These data set names were generated using patterns that were specified
in the "Edit data set name patterns" panel.  Override these names by
overtyping. These data sets will be created, if they do not exist.

  Compilation output data sets:
    Listing . 'CHABERT.TRADER.LISTING(TRADERB)'
    Object. . 'CHABERT.TRADER.OBJECT(TRADERB)'
    SYSDEBUG. 'CHABERT.TRADER.SYSDEBUG(TRADERB)'
    SYSTERM . *

  CICS Translator and DB2 SQL Processor options (Integrated and Separate):
N   CICS. . . COBOL3 CICS
N   DB2 . . . HOST(COBOL)
                                                                  02/015
```

Figure 13-55 COBOL Compile - Verify Selections

TEST compiler option

The suboptions you specify when you compile your COBOL program with the TEST compiler option affect the size and performance of your program and the debugging capabilities available. Depending on the suboptions you specify, the compiler does the following:

- Creates the symbol tables
- Creates debugging information
- Inserts hooks at selected points in your program

Debug Tool uses the symbol tables to obtain information about program variables. Programs compiled with one of the following compilers and with the SEPARATE suboption store debugging information and symbol tables in a separate file.

- Enterprise COBOL for z/OS and OS/390, Version 3
- COBOL for OS/390 & VM, Version 2 Release 2
- COBOL for OS/390 & VM, Version 2 Release 1 with APAR PQ40298

The file, called a separate debug file, must be a non-temporary file and must also be available during the debug session. If you move or rename the separate debug file, specify the new location using one of the following methods:

- Enter the SET SOURCE command with the name of the new location.
- Enter the SET DEFAULT LISTINGS command with the name of the new location.
- Specify the EQADEBUG DD statement with the name of the new location.
- Code the EQAUEDAT user exit with the new location.

Debug Tool uses hooks to gain control of your program at selected points during its execution. The hooks do not modify your source. The hooks are inserted into your program at one of the following times:

- At compile time, when you specify the TEST compiler option with any suboption except NONE.
- At run time, if the Dynamic Debug facility is activated, which is the default. To use the Dynamic Debug facility while you debug programs that do not have compiled-in hooks, compile your programs with the TEST(NONE) compiler option.

If you want to use the DATA suboption of the PLAYBACK ENABLE command, you must specify the SYM suboption of the TEST compiler option when you compile your program.

For VS COBOL II programs, in addition to the TEST compiler option, you must specify:

- The SOURCE compiler option. This option is required to generate a listing file.
- The RESIDENT compiler option. This option is required by Language Environment to ensure that the necessary Debug Tool routines are loaded dynamically at run time.

If a data set (such as Listing, Object, or SYSDEBUG) is missing when you press Enter from the Verify Selections screen, in either foreground or background mode, the Create data set screen shown in Figure 13-56 will be displayed. You can overwrite any values on this screen.

```
------------------------------- Create data set -------------------------------
Command ===> _____________________________________________________________________


Specify parameters for Listing
   'CHABERT.TRADER.LISTING(TRADERB)'

Data set Allocation Parameters:
 Data set name type. . . . PDSE___       (PDSE, PDS, or SEQ)

 Allocate command string:
CYLINDER SPACE(1 1) LRECL(133) RECFM(F B A)

 Press  ENTER to create the data set
 Press  EXIT or CANCEL to cancel creation

                                                                        02/015
```

Figure 13-56 Create data set

Once the data sets are defined, the compile is started. If you defined the compile to be run in background, you will first get a panel showing the JCL (where you also may want to overwrite any statements) as shown in Figure 13-57.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ---------------------------------------------------------------------------
EDIT       CHABERT.DTPPTEMP.C1162984                     Columns 00001 00072
Command ===> ___________________________________________ Scroll ===> CSR_
000008 //*****************************************************************
000009 //*  Debug Tool Utilities - COBOL Compile
000010 //*****************************************************************
000011 //EQASPBC1 EXEC PGM=IGYCRCTL,
000012 // PARM='TEST(NONE,SYM,SEPARATE)'
000013 //STEPLIB    DD  DISP=SHR,
000014 //       DSNAME=IGY.V3R4M0.SIGYCOMP
000015 //SYSIN    DD  DISP=SHR,
000016 //       DSNAME=CHABERT.TRADER.SOURCE(TRADERB)
000017 //SYSPRINT DD  DISP=OLD,
000018 //       DSNAME=CHABERT.TRADER.LISTING(TRADERB)
000019 //SYSTERM  DD  SYSOUT=*
000020 //SYSDEBUG DD  DISP=OLD,
000021 //       DSNAME=CHABERT.TRADER.SYSDEBUG(TRADERB)
000022 //SYSLIN   DD  DISP=OLD,
000023 //       DSNAME=CHABERT.TRADER.OBJECT(TRADERB)
000024 //*
000025 //SYSUT1   DD  SPACE=(CYL,(1,1)),UNIT=SYSALLDA
000026 //SYSUT2   DD  SPACE=(CYL,(1,1)),UNIT=SYSALLDA
000027 //SYSUT3   DD  SPACE=(CYL,(1,1)),UNIT=SYSALLDA
                                                                        05/027
```

Figure 13-57 Background JCL

You can save this as a model to be reused, after customization, for other COBOL compilations.

After the compile is completed and if you are using foreground mode, you will get a slightly different panel as shown in Figure 13-58.

```
-------------------------COBOL Compile - View Outputs-------------------------
Command ===> ____________________________________________________________________
                                                                   More:     +
Compile using 5655-G53 IBM Enterprise COBOL for z/OS and OS/390

DB2 Precompiler return code = NOT RUN
CICS Translator return code = NOT RUN
COBOL Compiler  return code = 12

The following data sets were used for this compilation.
Enter E to Edit, V to View, or B to Browse these data sets.

  _ Input . : 'CHABERT.TRADER.SOURCE(TRADERB)'

  Compilation output data sets:
  B Listing : 'CHABERT.TRADER.LISTING(TRADERB)'
  _ Object. : 'CHABERT.TRADER.OBJECT(TRADERB)'
  _ SYSDEBUG: 'CHABERT.TRADER.SYSDEBUG(TRADERB)'
  _ SYSTERM : *

N DB2 SQL output DBRM data set:
  _ DBRMLIB :

N CICS Translator data sets:
                                                                  16/006
```

Figure 13-58 COBOL Compile - View Outputs

If the compile did not work, you can browse the listings data set directly from here by entering `B` just before the data set name. If you find that the compile did not work because of a syntax problem, then edit the source from here by entering `E` just before the input data set name.

As you can see in Figure 13-58, the return code from the compiler was 12. Enter `B` to browse the Listing data set to find the reason for the return code of 12, as shown in Figure 13-59.

```
  Menu  Utilities  Compilers  Help
 ____________________________________________________________________________
 BROWSE    CHABERT.TRADER.LISTING(TRADERB)          Line 00001409 Col 001 080
 Command ===> ____________________________________________ Scroll ===> CSR

    748   WRITE-TRANSACTION-REPORT . . .  P442 P499
    770   WRITE-TRANSACTION-REPORT-EXIT
PP 5655-G53 IBM Enterprise COBOL for z/OS  3.4.0              TRADERB   Date 11
 Defined   Cross-reference of programs     References

     25   TRADERB
PP 5655-G53 IBM Enterprise COBOL for z/OS  3.4.0              TRADERB   Date 11
LineID  Message code  Message text

    71  IGYGR1216-I   A "RECORDING MODE" of "F" was assumed for file "REPORT-FIL

    74  IGYGR1216-I   A "RECORDING MODE" of "F" was assumed for file "TRAN-REPOR

   111  IGYDS1159-E   A "PICTURE" clause was not found for elementary item "CUST

   112  IGYDS0010-S   A "COPY" statement was found but the "LIB" compiler option
                      item following the next period.

                      Same message on line:    114    116
                                                                  24/029
```

Figure 13-59 COBOL Compile - Listing data set

After correcting all errors, you can link edit the program.

This link utility lets you select either for a foreground or background link. You need to specify your object library; you can define the object member, but are not required to. If you do not specify the object member, you will be presented a member selection list later.

The panels shown for the link edit are quite similar to those used for compiling the program and are not included here.

Manage and use Debug Tool setup file

Setup files can save you time when you are debugging a program that needs to be restarted multiple times. Setup files store information needed to run your program and start Debug Tool.

You can create several setup files for each program; each setup file can store information about starting and running your program in different circumstances. To create and manage files, use Debug Tool Setup Utility (DTSU), which is part of Debug Tool. You do not need Debug Tool Utilities and Advanced Functions to use this tool.

From the Debug Tool Utilities panel shown in Figure 13-60, select option **2 - Manage and Use Debug Tool Setup Files**.

```
---------------------------- Debug Tool Utilities ----------------------------
Option ===> 2_____________________________________________________________________

                                                                More:     +
0  Manage Job Card
   For Program Preparation and Setup File Management

1  Program Preparation
   Compile old or new COBOL programs with newer compilers, convert old COBOL
   source into new COBOL source, use other compilers, and link edit.

2  Manage and Use Debug Tool Setup Files
   You can manage setup files and use them to run your program interactively
   with Debug Tool in TSO Foreground or submit your program to run in
   the background using MVS batch.

3  Code Coverage
   Measure code coverage in programs written in COBOL, PL/I, C/C++ and
   Assembler when compiled with specific IBM compilers and HLASM.

4  Manage IMS Programs

                                                                    02/015
```

Figure 13-60 Debug Tool Utilities

In this example we describe how to create a new setup file member.

1. Enter the new member name DEBUGTR as shown in Figure 13-61 and press Enter.

```
------------------- Debug Tool Foreground - Edit Setup File ------------------
Command ===> ____________________________________________________________________


Setup File Library:
   Project . . . CHABERT
   Group . . . . DEBUG   . . . ________ . . . ________ . . . ________
   Type  . . . . SETUPFIL
   Member  . . . DEBUGTR        (Blank or pattern for member selection list)
                                (or existing or new member name)

Other Data Set Name:
   Data Set Name . . . ____________________________________________________
   Volume Serial . . . ______    (If not cataloged)


_ Initialize New setup file for DB2 (/)

                                                                        09/026
```

Figure 13-61 Debug Tool Foreground - Edit Setup File

2. Figure 13-62 is displayed.

```
EDIT - Edit Setup File 'CHABERT.DEBUG.SETUPFIL(DEBUGTR)'      Row 1 to 1 of 1
Command ===> COPY_____________________________________ Scroll ===> PAGE

Modify information and use the  Run command to execute,
        or the  Submit command to submit to Batch.
     Press HELP for a list of all available commands.

Load Module Name ________
Choose the format of your parameter string:
1 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs    - Run-time Options  / Program Arguments

_ Enter / to modify parameters ____________________________________________

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)            DISP
___ ________ ___ _ ****************** Top of Data ******************** _____
******************************** Bottom of data ********************************

                                                                        02/019
```

Figure 13-62 Debug Tool - Edit Setup File

– The top part of the Edit Setup File panel contains the name of the program (load module) that you want to run, and the run-time parameter string. If the setup file is for a DB2 program, the panel also contains fields for the DB2 System identifier and the DB2 plan.

– The bottom part of the Edit Setup File panel contains the file allocation statements. This part of the panel is similar to an ISPF edit panel. You can insert new lines, copy (repeat) a line, delete a line, and type over information on a line.

3. You can enter the COPY command to copy information from another setup file or JCL data set into the current setup file. The JCL to execute the program we want to test resides at CHABERT.TRADER.JCL(TRADERB).

   Enter the COPY command on the command line in Figure 13-62 and press Enter. The panel shown in Figure 13-63 is displayed. Fill in the data set name where the JCL resides and press Enter.

```
------------- Debug Tool Foreground - Copy from Setup File or JCL ------------
Command ===> ____________________________________________________________________

Select data to copy into 'CHABERT.DEBUG.SETUPFIL(DEBUGTR)'

Setup File or JCL Library:
   Project . . . CHABERT
   Group . . . . TRADER  . . . ________ . . . ________ . . . ________
   Type  . . . . JCL
   Member  . . . T*_____        (Blank or pattern for member selection list)
                                (or existing or new member name)

Other Data Set Name:
   Data Set Name . . . ____________________________________________________________
   Volume Serial . . . ______    (If not cataloged)


Note: When you copy from another setup file the entire contents are copied.
      When copying from JCL you can select the information you want to copy.

                                                                          10/021
```

Figure 13-63 Debug Tool - Copy from Setup File or JCL

4. Figure 13-64 is displayed. Select the member TRADERB and press Enter.

```
  Menu  Functions  Confirm  Utilities  Help
 ____________________________________________________________________________
LIBRARY          CHABERT.TRADER.JCL                        Row 00001 of 00006
Command ===> ____________________________________________ Scroll ===> CSR
   Name     Prompt          Size    Created          Changed            ID
 _ TABLES                    48   1999/08/09   2004/11/04 16:32:35    CHABERT
 S TRADERB                   13   2001/06/21   2004/11/21 13:48:27    CHABERT
 _ TRADERBS                  20   2001/06/21   2004/11/22 19:02:00    CHABER2
 _ TRADERD                   23   2002/09/25   2004/11/22 19:02:05    CHABER2
 _ TRADERI                   17   2002/10/07   2004/11/22 19:02:09    CHABER2
 _ TRANFILE                   3   2004/11/21   2004/11/21 13:59:26    CHABERT
   **End**
                                                                        07/013
```

Figure 13-64 Debug Tool - Library member

5. Figure 13-65 is displayed. Type `S*` on the command line and press Enter.

```
---------------- Debug Tool Foreground - Copy from JCL Datas Row 1 to 12 of 12
Command ===> S*_____________________________________________ Scroll ===> PAGE

Enter S* on the command line or on a Sel line to select all JCL statements.
Enter S on a Sel line to select that JCL statement.
Enter RESET to deselect all JCL statements.

Sel   JCL Image
____ //BATCHTRA JOB
____ //* from 'CHABERT.TRADER.JCL (TRADERB)'
____ //GOTRAD EXEC PGM=TRADERB
____ //STEPLIB  DD  DISP=SHR,DSN=CHABERT.TRADER.LOAD
____ //         DD  DISP=SHR,DSN=ADTOOLS.DEBUG.TOOL51.SEQAMOD
____ //SYSPRINT DD  SYSOUT=*
____ //SYSABEND DD  SYSOUT=*
____ //COMPFILE DD  DISP=SHR,DSN=CHABERT.TRADER.COMPFILE
____ //CUSTFILE DD  DISP=SHR,DSN=CHABERT.TRADER.CUSTFILE
____ //TRANSACT DD  DISP=SHR,DSN=CHABERT.TRADER.SAMPLES(TRANFILE)
____ //REPOUT   DD  SYSOUT=*
____ //TRANREP  DD  SYSOUT=*
****************************** Bottom of data ******************************

                                                                    02/017
```

Figure 13-65 Debug Tool - Copy from JCL Data Rows

6. All JCL statements will be selected as shown in Figure 13-66.

```
---------------- Debug Tool Foreground - Copy from JCL Datas Row 1 to 13 of 13
Command ===> _____________________________________________ Scroll ===> PAGE

Enter S* on the command line or on a Sel line to select all JCL statements.
Enter S on a Sel line to select that JCL statement.
Enter RESET to deselect all JCL statements.

Sel   JCL Image
S___ //BATCHTRA JOB
____ //* from 'CHABERT.TRADER.JCL (TRADERB)'
S___ //GOTRAD EXEC PGM=TRADERB,
S___ //  PARM=RPTOPTS(YES)
S___ //STEPLIB  DD  DISP=SHR,DSN=CHABERT.TRADER.LOAD
S___ //         DD  DISP=SHR,DSN=ADTOOLS.DEBUG.TOOL51.SEQAMOD
S___ //SYSPRINT DD  SYSOUT=*
S___ //SYSABEND DD  SYSOUT=*
S___ //COMPFILE DD  DISP=SHR,DSN=CHABERT.TRADER.COMPFILE
S___ //CUSTFILE DD  DISP=SHR,DSN=CHABERT.TRADER.CUSTFILE
S___ //TRANSACT DD  DISP=SHR,DSN=CHABERT.TRADER.SAMPLES(TRANFILE)
S___ //REPOUT   DD  SYSOUT=*
S___ //TRANREP  DD  SYSOUT=*
****************************** Bottom of data ******************************

                                                                    21/003
```

Figure 13-66 Debug Tool - JCL statements

7. All JCL statements have now been selected. Press PF3 and the panel for editing the setup is displayed as shown in Figure 13-67.
8. You can now set parameters for debugging. Put a / in front of Enter / to modify parameters, and press Enter.

```
EDIT - Edit Setup File 'CHABERT.DEBUG.SETUPFIL(DEBUGTR)'      Row 1 to 7 of 10
Command ===> _____________________________________________ Scroll ===> PAGE

Modify information and use the  Run command to execute,
        or the  Submit command to submit to Batch.
     Press HELP for a list of all available commands.

Load Module Name TRADERB
Choose the format of your parameter string:
1 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs    - Run-time Options  / Program Arguments

/ Enter / to modify parameters _____________________________________________

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)            DISP
___ ________ ___ _ ***************** Top of Data ********************  ____
___ COMPFILE 1__ _ 'CHABERT.TRADER.COMPFILE'                          SHR
___ CUSTFILE 1__ _ 'CHABERT.TRADER.CUSTFILE'                          SHR
___ REPOUT   1__ _ SYSOUT=*                                           ____
___ STEPLIB  1__ _ 'CHABERT.TRADER.LOAD'                              SHR
___ ________ 2__ _ 'ADTOOLS.DEBUG.TOOL51.SEQAMOD'                     SHR
___ SYSABEND 1__ _ SYSOUT=*                                           ____
                                                                    14/004
```

Figure 13-67 Debug Tool - Edit

9. Figure 13-68 shows the default settings.

```
--------------- Debug Tool Foreground - Modify Parameter String --------------
Command ===> ____________________________________________________________
                                                              More:     +
Modify Test options, other run-time options, and Program arguments

Select Test Options:

Test Option      ==> TEST          Test/Notest
Test Level       ==> ALL           All/Error/None
Commands File                      *, DDname, or Data Set Name
                 ==> *
Prompt Level     ==> PROMPT        Prompt/NoPrompt
Preference File                    *, DDname, or Data Set Name
                 ==> INSPPREF

Select (/) a session type and provide parameters:

 / Full-screen mode
   Terminal LU     ==> ____________ blank or MFI VTAM Terminal LU

 _ Remote debug mode
   Connection type ==> ________     SINGLE/MULTIPLE socket
   Address         ==> ____________________________________________
   Port            ==> 8000________
                                                                    02/015
```

Figure 13-68 Debug Tool - Modify Parameter String (for MFI)

10. Press **PF3** to accept the defaults. This brings you back to the previous screen, which has been updated as shown in Figure 13-69.

```
EDIT - Edit Setup File 'CHABERT.DEBUG.SETUPFIL(DEBUGTR)'      Row 1 to 7 of 10
Command ===> _____________________________________________ Scroll ===> PAGE

Modify information and use the  Run command to execute,
        or the  Submit command to submit to Batch.
     Press HELP for a list of all available commands.

Load Module Name TRADERB
Choose the format of your parameter string:
1 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs    - Run-time Options  / Program Arguments

_ Enter / to modify parameters '/TEST(ALL,'*',PROMPT,'MFI:INSPPREF') '

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)            DISP
 I  ________ ___ _ ****************** Top of Data ********************
___ COMPFILE 1__ _ 'CHABERT.TRADER.COMPFILE'                          SHR
___ CUSTFILE 1__ _ 'CHABERT.TRADER.CUSTFILE'                          SHR
___ REPOUT   1__ _ SYSOUT=*                                           ____
___ STEPLIB  1__ _ 'CHABERT.TRADER.LOAD'                              SHR
___ ________ 2__ _ 'ADTOOLS.DEBUG.TOOL51.SEQAMOD'                     SHR
___ SYSABEND 1__ _ SYSOUT=*                                           ____
                                                                  18/004
```

Figure 13-69 Debug Tool - Edit

11. As shown in Figure 13-69, you are using a preference file, INSPPREF, a file that gives initial commands to Debug Tool when debugging starts. This file was not included in the JCL that was copied, so you need to add this file. Put an `I` on the Cmd field to the left as shown in Figure 13-69 and press Enter; Figure 13-70 is displayed.

```
EDIT - Edit Setup File 'CHABERT.DEBUG.SETUPFIL(DEBUGTR)'      Row 1 to 7 of 11
Command ===> _____________________________________________ Scroll ===> PAGE

Modify information and use the  Run command to execute,
        or the  Submit command to submit to Batch.
     Press HELP for a list of all available commands.

Load Module Name TRADERB
Choose the format of your parameter string:
1 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs    - Run-time Options  / Program Arguments

_ Enter / to modify parameters '/TEST(ALL,'*',PROMPT,'MFI:INSPPREF') '

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)            DISP
___ ________ ___ _ ****************** Top of Data ********************
___ ________ 1__ _ ** Select Detail DD Information **                 ____
___ COMPFILE 1__ _ 'CHABERT.TRADER.COMPFILE'                          SHR
___ CUSTFILE 1__ _ 'CHABERT.TRADER.CUSTFILE'                          SHR
___ REPOUT   1__ _ SYSOUT=*                                           ____
___ STEPLIB  1__ _ 'CHABERT.TRADER.LOAD'                              SHR
___ ________ 2__ _ 'ADTOOLS.DEBUG.TOOL51.SEQAMOD'                     SHR
                                                                  02/015
```

Figure 13-70 Debug Tool - Edit

12. As shown in Figure 13-70, a line has been added with the text **Select Detail DD Information.** On this line add the data set and member that contains the initial commands for the debug session, as shown in Figure 13-71.

```
EDIT - Edit Setup File 'CHABERT.DEBUG.SETUPFIL(DEBUGTR)'      Row 1 to 7 of 11
Command ===> _____________________________________________ Scroll ===> PAGE

Modify information and use the  Run command to execute,
        or the  Submit command to submit to Batch.
     Press HELP for a list of all available commands.

Load Module Name TRADERB
Choose the format of your parameter string:
1 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs    - Run-time Options  / Program Arguments

_ Enter / to modify parameters '/TEST(ALL,'*',PROMPT,'MFI:INSPPREF') '

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)            DISP
___ ________ ___ _ ***************** Top of Data ********************  ____
___ INSPPREF 1   _ 'CHABERT.DEBUG.PREF(TRAD1)'                        SHR
___ COMPFILE 1   _ 'CHABERT.TRADER.COMPFILE'                          SHR
___ CUSTFILE 1   _ 'CHABERT.TRADER.CUSTFILE'                          SHR
___ REPOUT   1   _ SYSOUT=*                                           ____
___ STEPLIB  1   _ 'CHABERT.TRADER.LOAD'                              SHR
___ ________ 2   _ 'ADTOOLS.DEBUG.TOOL51.SEQAMOD'                     SHR
                                                                  02/015
```

Figure 13-71 Debug Tool - Edit

13. Issue the **RUN** command on the command line as shown in Figure 13-72; debugging is now started.

```
EDIT - Edit Setup File 'CHABERT.DEBUG.SETUPFIL(DEBUGTR)'      Row 1 to 7 of 11
Command ===> RUN_____________________________________________ Scroll ===> PAGE

Modify information and use the  Run command to execute,
        or the  Submit command to submit to Batch.
     Press HELP for a list of all available commands.

Load Module Name TRADERB
Choose the format of your parameter string:
1 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs    - Run-time Options  / Program Arguments

_ Enter / to modify parameters '/TEST(ALL,'*',PROMPT,'MFI:INSPPREF') '

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)            DISP
___ ________ ___ _ ***************** Top of Data ********************  ____
___ INSPPREF 1   _ 'CHABERT.DEBUG.PREF(TRAD1)'                        SHR
___ COMPFILE 1   _ 'CHABERT.TRADER.COMPFILE'                          SHR
___ CUSTFILE 1   _ 'CHABERT.TRADER.CUSTFILE'                          SHR
___ REPOUT   1   _ SYSOUT=*                                           ____
___ STEPLIB  1   _ 'CHABERT.TRADER.LOAD'                              SHR
___ ________ 2   _ 'ADTOOLS.DEBUG.TOOL51.SEQAMOD'                     SHR
                                                                  02/018
```

Figure 13-72 Debug Tool - Run command

The preference file in Example 13-13 is used during our FMI session.

Example 13-13 Preference file

```
SET FREQUENCY ON                                ;
SET SUFFIX ON                                   ;
SET MSGID ON                                    ;
SET AUTOMONITOR ON LOG                          ;
```

```
SET DEFAULT SCROLL CSR                          ;
SET COLOR BLEU SOURCE AREA                      ;
SET COLOR BLEU MONITOR AREA                     ;
SET PF1  "PK13-24"  = SET KEYS 24               ;
SET PF2  "KEEP"     = MON LOCAL %CU LIST CURSOR ;
SET PF3  "QUIT"     = QUIT                      ;
SET PF4  "CURRLINE" = QUALIFY RESET             ;
SET PF5  "FIND"     = FIND                      ;
SET PF6  "SETBREAK" = AT TOGGLE CURSOR          ;
SET PF7  "UP"       = SCROLL UP                 ;
SET PF8  "DOWN"     = SCROLL DOWN               ;
SET PF9  "GO 1"     = STEP                      ;
SET PF10 "ZOOM WIN" = IMMEDIATE ZOOM            ;
SET PF11 "ZOOM LOG" = WINDOW ZOOM LOG           ;
SET PF12 "GO"       = GO                        ;
SET PF13 "PK 1-12"  = SET KEYS 12               ;
SET PF14 "CLR KEEP" = CLEAR MONITOR             ;
SET PF15 "QUIT"     = QUIT                      ;
SET PF16 "LIST CSR" = LIST CURSOR               ;
SET PF17 "FIND"     = IMMEDIATE FIND            ;
SET PF18 "AT/CLEAR" = AT TOGGLE                 ;
SET PF19 "TOP"      = TOP                       ;
SET PF20 "BOTTOM"   = BOT                       ;
SET PF21 "GO 1"     = STEP                      ;
SET PF22 "ZOOM WIN" = IMMEDIATE ZOOM            ;
SET PF23 "ZOOM LOG" = WINDOW ZOOM LOG           ;
SET PF24 "RETRIEVE" = IMMEDIATE RETRIEVE        ;
WINDOW SIZE 10 MONITOR                          ;
WINDOW SIZE 10 SOURCE                           ;
```

13.3.3 Using MainFrame Interface debugger

The RUN command starts the MFI debugger. The resultant MFI debugging screen is shown in Figure 13-73. This section describes how to use the MFI debugger.

```
COBOL    LOCATION: TRADERB initialization
Command ===> CUSTOMER-NAME_                                    Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 0 OF 0
****************************** TOP OF MONITOR ******************************
****************************** BOTTOM OF MONITOR ***************************




SOURCE: TRADERB --1----+----2----+----3----+----4----+----5----+ LINE: 1 OF 811
       1        ***************************************************************.
       2        * LICENSED MATERIALS - PROPERTY OF IBM                        .
       3        * 5655-ADS (C) COPYRIGHT IBM CORP. 2004                       .
       4        * ALL RIGHTS RESERVED                                         .
       5        ***************************************************************.
       6        * PROGRAM:  TRADERB                                            .
       7        *                                                             .
       8        * AUTHOR :  DAVE BARFIELD                                      .
       9        * MODIFIED: LARRY KAHM                                         .
LOG 0----+----1----+----2----+----3----+----4----+----5----+-- LINE: 140 OF 140
PF  1:PK13-24    2:KEEP       3:QUIT       4:CURRLINE   5:FIND       6:SETBREAK
PF  7:UP         8:DOWN       9:GO 1      10:ZOOM WIN  11:ZOOM LOG  12:GO
                                                                     02/028
```

Figure 13-73 MFI - Screen on entry

1. The MFI screen has been customized according to your preference file. To monitor the TR-CUSTOMER-NAME, enter CUSTOMER-NAME on the command line and press PF5: Find; this screen returned is shown in Figure 13-74.

```
COBOL    LOCATION: TRADERB initialization
Command ===>                                                   Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 0 OF 0
****************************** TOP OF MONITOR ******************************
****************************** BOTTOM OF MONITOR ***************************




SOURCE: TRADERB --1----+----2----+----3----+----4----+----5--- LINE: 150 OF 811
     150            05 TR-CUSTOMER-NAME         PIC X(30).                  .
     151            05 FILLER                    PIC X(1) VALUE '.'.         .
     152            05 TR-COMPANY-NAME           PIC X(20).                  .
     153            05 TR-REQUEST-TYPE           PIC X(15).                  .
     154            05 TR-NO-OF-SHARES           PIC 9(4).                   .
     155            05 TR-SUBTYPE                PIC X(1).                   .
     156            05 FILLER                    PIC X(4).                   .
     157                                                                     .
     158        ***************************************************************.
LOG 0----+----1----+----2----+----3----+----4----+----5----+-- LINE: 140 OF 140
PF  1:PK13-24    2:KEEP       3:QUIT       4:CURRLINE   5:FIND       6:SETBREAK
PF  7:UP         8:DOWN       9:GO 1      10:ZOOM WIN  11:ZOOM LOG  12:GO
                                                                     13/030
```

Figure 13-74 MFI - Find command

2. With your cursor on the TR-CUSTOMER-NAME field, press PF2:Keep to monitor it. The resulting screen is shown in Figure 13-75.

```
COBOL    LOCATION: TRADERB initialization
Command ===> STEP 9_                                          Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 2
****************************** TOP OF MONITOR ******************************
0001  1 TR-CUSTOMER-NAME
0002        EQA1576E The environment is not yet fully initialized.
***************************** BOTTOM OF MONITOR ****************************



SOURCE: TRADERB --1----+----2----+----3----+----4----+----5--- LINE: 150 OF 811
     150              05 TR-CUSTOMER-NAME          PIC X(30).                 .
     151              05 FILLER                     PIC X(1) VALUE '.'.        .
     152              05 TR-COMPANY-NAME            PIC X(20).                 .
     153              05 TR-REQUEST-TYPE            PIC X(15).                 .
     154              05 TR-NO-OF-SHARES            PIC 9(4).                  .
     155              05 TR-SUBTYPE                 PIC X(1).                  .
     156              05 FILLER                     PIC X(4).                  .
     157                                                                        .
     158        ***************************************************************.
LOG 0----+----1----+----2----+----3----+----4----+----5----+-- LINE: 144 OF 144
PF  1:PK13-24    2:KEEP       3:QUIT       4:CURRLINE   5:FIND       6:SETBREAK
PF  7:UP         8:DOWN       9:GO 1      10:ZOOM WIN  11:ZOOM LOG  12:GO
                                                                      02/021
```

Figure 13-75 MFI - Monitor list

3. Take a couple of steps (using PF9); Figure 13-76 is displayed.

```
COBOL    LOCATION: TRADERB :> 369.1
Command ===> _                                                Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 4
****************************** TOP OF MONITOR ******************************
0001  1 TR-CUSTOMER-NAME     '000000000000000000000000000000000000000000000
0002    000000000000'
0003  2 EQA1252I ********** AUTOMONITOR ***********
0004    02 TRADERB:>WS-CUST-FILE-STATUS     '00'
***************************** BOTTOM OF MONITOR ****************************


SOURCE: TRADERB --1----+----2----+----3----+----4----+----5--- LINE: 366 OF 811
     366        *    END-EVALUATE                                            .
     367                                                                      .
     368             OPEN I-O      IO-CUSTOMER-FILE                          1
     369             EVALUATE      WS-CUST-FILE-STATUS                       0
     370                WHEN ZERO                                             .
     371                     CONTINUE                                         0
     372                WHEN OTHER                                            .
     373                     DISPLAY 'OPEN OF CUSTOMER-FILE WAS UNSUCCESSFUL.' 0
     374                     DISPLAY 'FILE STATUS ' WS-CUST-FILE-STATUS       0
LOG 0----+----1----+----2----+----3----+----4----+----5----+-- LINE: 167 OF 167
PF  1:PK13-24    2:KEEP       3:QUIT       4:CURRLINE   5:FIND       6:SETBREAK
PF  7:UP         8:DOWN       9:GO 1      10:ZOOM WIN  11:ZOOM LOG  12:GO
                                                                      02/015
```

Figure 13-76 MFI - Step command

4. The variable referenced in the current statement is displayed because the SET AUTOMONITOR ON command is part of your preference file.

   Add an additional breakpoint. Using the following commands, set a breakpoint when the values changes:

   ```
   AT CHANGE TR-COMPANY-NAME
   AT CHANGE TR-REQUEST-TYPE
   ```

   The result is shown in Figure 13-77.

```
COBOL    LOCATION: TRADERB :> 369.1
Command ===> AT CHANGE TR-REQUEST-TYPE_                          Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 4
******************************** TOP OF MONITOR ********************************
0001  1 TR-CUSTOMER-NAME     '0000000000000000000000000000000000000000000000000
0002    000000000000'
0003  2 EQA1252I ********** AUTOMONITOR **********
0004    02 TRADERB:>WS-CUST-FILE-STATUS     '00'
****************************** BOTTOM OF MONITOR ******************************


SOURCE: TRADERB --1----+----2----+----3----+----4----+----5--- LINE: 366 OF 811
     366        *    END-EVALUATE                                              .
     367                                                                        .
     368             OPEN I-O      IO-CUSTOMER-FILE                             1
     369             EVALUATE      WS-CUST-FILE-STATUS                          0
     370                WHEN ZERO                                               .
     371                     CONTINUE                                           0
     372                WHEN OTHER                                              .
     373                     DISPLAY 'OPEN OF CUSTOMER-FILE WAS UNSUCCESSFUL.'  0
     374                     DISPLAY 'FILE STATUS ' WS-CUST-FILE-STATUS         0
LOG 0----+----1----+----2----+----3----+----4----+----5----+-- LINE: 168 OF 168
PF  1:PK13-24    2:KEEP       3:QUIT       4:CURRLINE   5:FIND       6:SETBREAK
PF  7:UP         8:DOWN       9:GO 1      10:ZOOM WIN  11:ZOOM LOG  12:GO
                                                                      02/040
```

Figure 13-77 MFI - AT CHANGE command

5. Verify how these two commands were executed. Press PF11:Zoom Log to expand the log window shown in Figure 13-78.

```
COBOL    LOCATION: TRADERB :> 369.1
Command ===> _                                                   Scroll ===> CSR
LOG 0----+----1----+----2----+----3----+----4----+----5----+-- LINE: 151 OF 169
0151 EQA1238I The current location is TRADERB ::> TRADERB :> 330.1.
0152 EQA1140I 03 TRADERB:>WS-YR  of 02 TRADERB:>WS-CURRENT-DATE  = '2004'
0153 EQA1140I 03 TRADERB:>WS-MM  of 02 TRADERB:>WS-CURRENT-DATE  = '11'
0154 EQA1140I 03 TRADERB:>WS-DD  of 02 TRADERB:>WS-CURRENT-DATE  = '24'
0155 EQA1238I The current location is TRADERB ::> TRADERB :> 332.1.
0156 EQA1238I The current location is TRADERB ::> TRADERB :> 349.1.
0157 EQA1238I The current location is TRADERB ::> TRADERB :> 352.1.
0158 EQA1140I FD TRADERB:>IN-TRANSACT-FILE  = CLOSED, FILE STATUS: 00
0159 EQA1140I FD TRADERB:>IN-COMPANY-FILE  = CLOSED, FILE STATUS: 00
0160 EQA1140I FD TRADERB:>REPORT-FILE  = CLOSED, FILE STATUS: 00
0161 EQA1140I FD TRADERB:>TRAN-REPORT-FILE  = CLOSED, FILE STATUS: 00
0162 EQA1238I The current location is TRADERB ::> TRADERB :> 355.1.
0163 EQA1140I 02 TRADERB:>WS-TRAN-FILE-STATUS  = '00'
0164 EQA1238I The current location is TRADERB ::> TRADERB :> 368.1.
0165 EQA1140I FD TRADERB:>IO-CUSTOMER-FILE  = CLOSED, FILE STATUS: 00
0166 EQA1238I The current location is TRADERB ::> TRADERB :> 369.1.
0167 EQA1140I 02 TRADERB:>WS-CUST-FILE-STATUS  = '00'
0168  AT CHANGE TR-COMPANY-NAME ;
0169  AT CHANGE TR-REQUEST-TYPE ;
PF  1:PK13-24    2:KEEP       3:QUIT       4:CURRLINE   5:FIND       6:SETBREAK
PF  7:UP         8:DOWN       9:GO 1      10:ZOOM WIN  11:ZOOM LOG  12:GO
                                                                      02/015
```

Figure 13-78 LMFI - Log window

6. Because there is no error, you can return to your original view using PF11:Zoom Log. Run the program to that breakpoint by typing `GO` or pressing PF12:Go. The result is shown in Figure 13-79.

```
COBOL    LOCATION: TRADERB :> 422.1
Command ===> DESCRIBE ATTRIBUTE TRANSACT-REC_                  Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 3
****************************** TOP OF MONITOR ******************************
0001  1 TR-CUSTOMER-NAME   'ERIC                          '
0002  2 EQA1252I ********** AUTOMONITOR **********
0003    03 TRADERB:>TR-REQUEST-TYPE    'BUY_SELL       '
****************************** BOTTOM OF MONITOR ******************************



SOURCE: TRADERB --1----+----2----+----3----+----4----+----5--- LINE: 420 OF 811
     420                                                                      .
     421                 NOT AT END                                           .
     422                     EVALUATE TR-REQUEST-TYPE                         0
     423                        WHEN SHARE-VALUE-REQ                          .
     424                             PERFORM GENERATE-CUSTOMER-REPORT THRU    0
     425                                     GENERATE-CUSTOMER-REPORT-EXIT    .
     426                        WHEN BUY-SELL-REQ                             .
     427                             PERFORM BUY-SELL                         0
     428                        WHEN OTHER                                    .
LOG 0----+----1----+----2----+----3----+----4----+----5----+-- LINE: 172 OF 172
PF  1:PK13-24    2:KEEP       3:QUIT       4:CURRLINE   5:FIND       6:SETBREAK
PF  7:UP         8:DOWN       9:GO 1      10:ZOOM WIN  11:ZOOM LOG  12:GO
                                                                        02/046
```

Figure 13-79 MFI - GO command

7. The program stops at line 422 because the TR-REQUEST-TYPE has been changed. You know that TR-REQUEST-TYPE is part of the TRANSACT-REC, and you need to know how it was defined. Use the `DESCRIBE ATTRIBUTE TRANSACT-REC` command; Debug Tool displays the definition in its log. Press PF11:Zoom Log to display Figure 13-80.

```
COBOL    LOCATION: TRADERB :> 422.1
Command ===> _                                                 Scroll ===> CSR
LOG 0----+----1----+----2----+----3----+----4----+----5----+-- LINE: 167 OF 185
0167 EQA1140I 02 TRADERB:>WS-CUST-FILE-STATUS  = '00'
0168  AT CHANGE TR-COMPANY-NAME ;
0169  AT CHANGE TR-REQUEST-TYPE ;
0170  GO ;
0171 EQA1238I The current location is TRADERB ::> TRADERB :> 422.1.
0172 EQA1140I 03 TRADERB:>TR-REQUEST-TYPE  = 'BUY_SELL       '
0173  DESCRIBE ATTRIBUTES TRANSACT-REC ;
0174 EQA1102I   ATTRIBUTES for TRANSACT-REC
0175 EQA1105I     Its length is 80
0176 EQA1103I     Its address is 126694F0
0177 EQA1112I     02 TRADERB:>TRANSACT-REC
0178 EQA1112I       03 TRADERB:>TR-ACCOUNT-NUMBER   X(5) DISP
0179 EQA1112I       03 TRADERB:>TR-CUSTOMER-NAME   X(30) DISP
0180 EQA1112I       03 TRADERB:>FILLER   X DISP
0181 EQA1112I       03 TRADERB:>TR-COMPANY-NAME   X(20) DISP
0182 EQA1112I       03 TRADERB:>TR-REQUEST-TYPE   X(15) DISP
0183 EQA1112I       03 TRADERB:>TR-NO-OF-SHARES   9999 DISP
0184 EQA1112I       03 TRADERB:>TR-SUBTYPE   X DISP
0185 EQA1112I       03 TRADERB:>FILLER   XXXX DISP
PF  1:PK13-24    2:KEEP       3:QUIT       4:CURRLINE   5:FIND       6:SETBREAK
PF  7:UP         8:DOWN       9:GO 1      10:ZOOM WIN  11:ZOOM LOG  12:GO
                                                                        02/015
```

Figure 13-80 MFI - DESCRIBE ATTRIBUTE command

8. If you step several times, because the FREQUENCY was turned ON via your preference file, the last right column of the source window displays the statement execution count as shown in Figure 13-81.

```
COBOL    LOCATION: TRADERB :> 482.1
Command ===> _                                                Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 3
0001  1 TR-CUSTOMER-NAME   'ERIC                          '
0002  2 EQA1252I ********** AUTOMONITOR **********
0003    EQA1945I There are no variables in the statement to display.
****************************** BOTTOM OF MONITOR ******************************



SOURCE: TRADERB --1----+----2----+----3----+----4----+----5--- LINE: 479 OF 811
     479                    THEN                                              .
     480                      PERFORM REWRITE-CUSTFILE                        3
     481                      IF RETURN-VALUE = CLEAN-RETURN                  3
     482                         PERFORM WRITE-TRANSACTION-REPORT             2
     483                      END-IF                                          .
     484                    END-IF                                            .
     485              WHEN CUSTOMER-NOT-FOUND                                 .
     486                    PERFORM BUILD-NEW-CUSTOMER                        0
     487                    PERFORM WRITE-CUSTFILE                            0
LOG 0----+----1----+----2----+----3----+----4----+----5----+ LINE: 1040 OF 1040
PF  1:PK13-24    2:KEEP       3:QUIT       4:CURRLINE   5:FIND       6:SETBREAK
PF  7:UP         8:DOWN       9:GO 1      10:ZOOM WIN  11:ZOOM LOG  12:GO
                                                                      02/015
```

Figure 13-81 MFI - Frequency

Note: Because the collection of frequency data can add a substantial amount of overhead, set the SET FREQUENCY command to ON only when you intend to make use of this data. Do not routinely set the SET FREQUENCY command to ON in debug sessions in which you do not intend to make use of this data.

9. Quit debugging and let the program continue until it is finished using the QUIT DEBUG command. The result is displayed in Figure 13-82.

```
BUY RECORDS        =          2
SELL RECORDS       =          0
UPDATE RECORDS     =          0
*** _
                                                                      04/006
```

Figure 13-82 End of program

13.3.4 Using IBM Distributed Debugger

1. For PL/I programs the first step it to start the distributed debugger listening using the command:

   ```
   C:\IBMDebug\bin\idegub.exe -qdaemon -quiport=8001
   ```

   The result is shown in Figure 13-83.

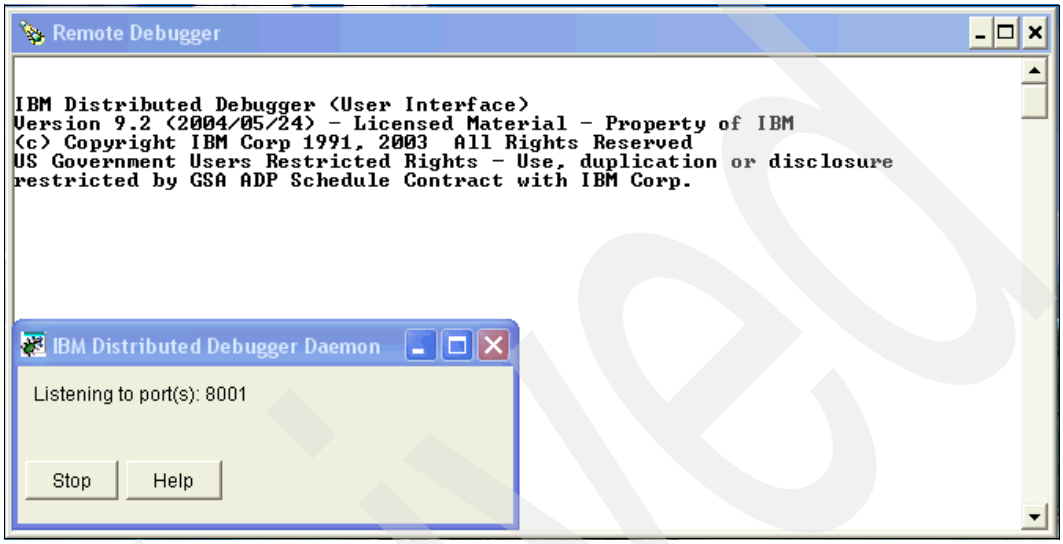


Figure 13-83 Remote debugger initialization

2. Issue the **Ipconfig** command from a Windows Command prompt window to display your IP-address as shown in Figure 13-84.

```
Ethernet adapter Local Area Connection (LAN):

        Connection-specific DNS Suffix  . : svl.ibm.com
        IP Address. . . . . . . . . . . . : 9.30.60.207
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 9.30.60.1
```

Figure 13-84 IP address

3. You can now use Debug Tool Set Utility (DTSU) to start the remote debugger interface. Modify the Debug Tool Foreground - Modify Parameter String panel as shown in Figure 13-85.

```
--------------- Debug Tool Foreground - Modify Parameter String --------------
Command ===> ____________________________________________________________________
                                                                   More:     +
Modify Test options, other run-time options, and Program arguments

Select Test Options:

Test Option      ==> TEST            Test/Notest
Test Level       ==> ALL             All/Error/None
Commands File                        *, DDname, or Data Set Name
                 ==> *
Prompt Level     ==> PROMPT          Prompt/NoPrompt
Preference File                      *, DDname, or Data Set Name
                 ==> INSPPREF

Select (/) a session type and provide parameters:

 _ Full-screen mode
   Terminal LU      ==> ____________ blank or MFI VTAM Terminal LU

 / Remote debug mode
   Connection type ==> MULTIPLE     SINGLE/MULTIPLE socket
   Address         ==> 9.30.60.207
   Port            ==> 8001
                                                                     24/025
```

Figure 13-85 Debug Tool - Modify Parameter String (for Remote Debugger)

4. Press PF3 to validate your values. This result is shown in Figure 13-86.

```
EDIT - Edit Setup File 'CHABERT.DEBUG.SETUPFIL(REMDEBUG)'     Row 1 to 7 of 10
Command ===> run_____________________________________________ Scroll ===> PAGE

Modify information and use the  Run command to execute,
        or the  Submit command to submit to Batch.
     Press HELP for a list of all available commands.

Load Module Name TRADERB
Choose the format of your parameter string:
1 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs    - Run-time Options  / Program Arguments

 _ Enter / to modify parameters '/TEST(ALL,'*',PROMPT,'VADTCPIP&9.30.60.207%8001
:*') '
____________________________________________________________________________

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)            DISP
___ ________ ___ _ ******************* Top of Data ********************  ____
___ COMPFILE 1__ _ 'CHABERT.TRADER.COMPFILE'                           SHR
___ CUSTFILE 1__ _ 'CHABERT.TRADER.CUSTFILE'                           SHR
___ REPOUT   1__ _ SYSOUT=*                                            ____
___ STEPLIB  1__ _ 'CHABERT.TRADER.LOAD'                               SHR
___ ________ 2__ _ 'ADTOOLS.DEBUG.TOOL51.SEQAMOD'                      SHR
___ SYSABEND 1__ _ SYSOUT=*                                            ____
      X SYSTEM                                                         02/018
```

Figure 13-86 Debug Tool - Edit and Run command (for Remote Debugger)

5. Issue the command `RUN` on the command line to start debugging using the IBM Distributed Debugger interface as shown in Figure 13-87.

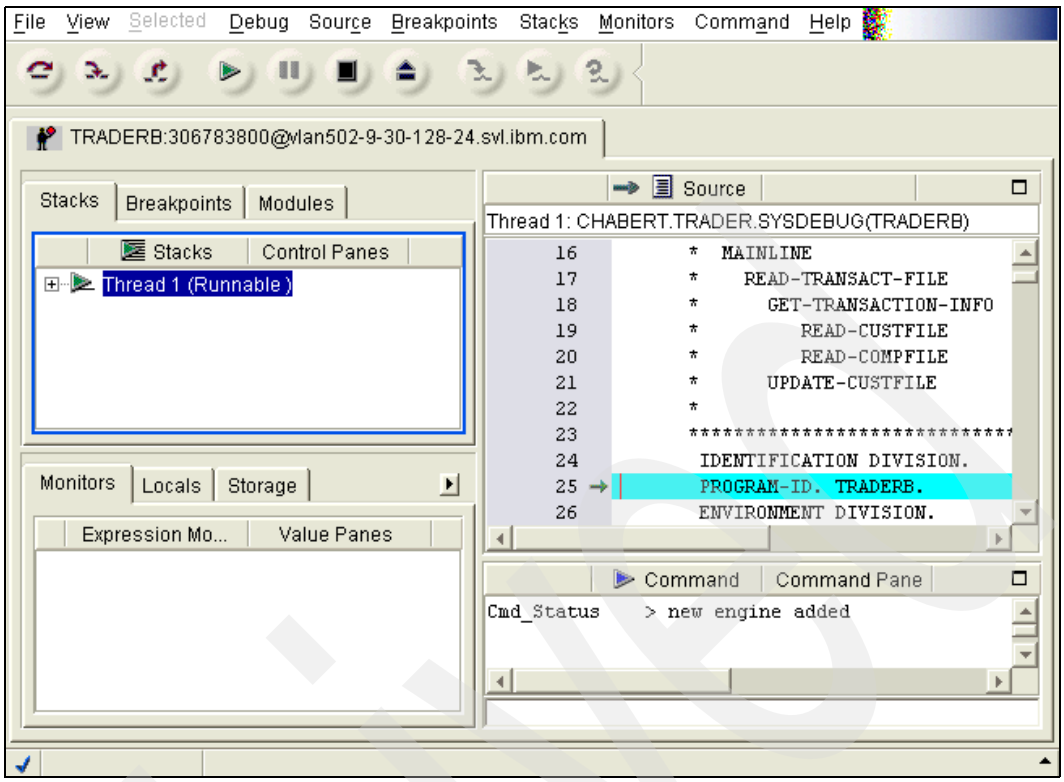


Figure 13-87 Remote Debugger - with Cobol source on entry

6. Place a breakpoint at the beginning of the actual program as shown in Figure 13-88.

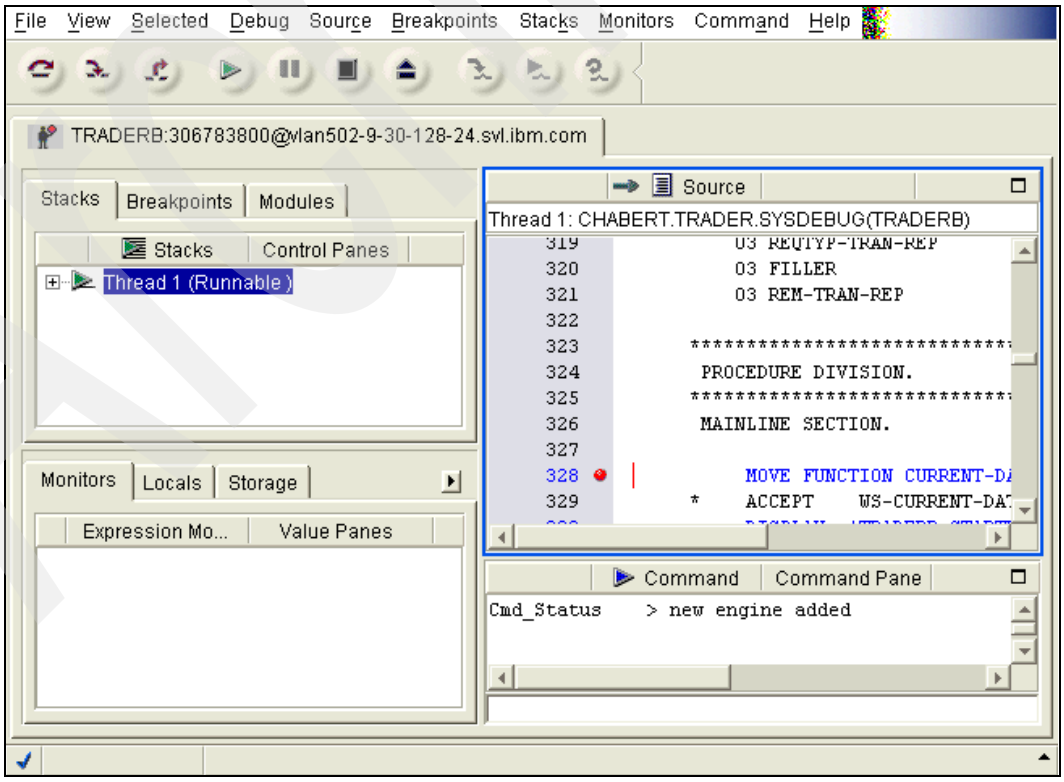


Figure 13-88 Remote Debugger - Breakpoint

7. Figure 13-89 shows the local variables when the breakpoint is triggered.

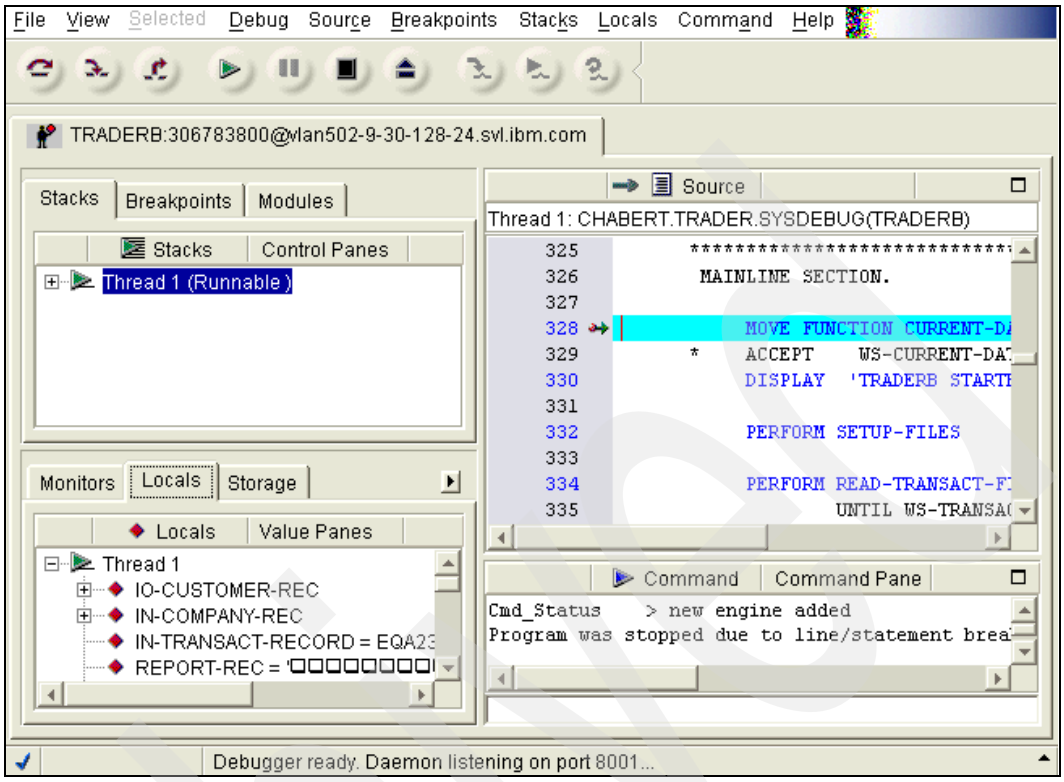


Figure 13-89 Remote Debugger - Local variable display at our initial breakpoint.

The IBM Distributed Debugger usage is the same as for PL/I.

13.4 Debugging non-LE programs using Debug Tool

There are several considerations that you must make when you debug programs that do not run under the Language Environment.

Some of these are unique to programs that contain no Language Environment routines, others pertain only when the initial program does not execute under control of the Language Environment, and still others apply to all programs that have mixtures of non-Language Environment and Language Environment programs.

13.4.1 Debugging exclusively non-LE programs

When Language Environment is not active, you can debug only assembler and disassembly programs. Debugging programs written in other languages requires the presence of an active Language Environment. In the case of OS/VSCobol, it can be run with LE only if you link-edit with LE first, before running with LE runtime library.

13.4.2 Debugging MVS batch or TSO non-LE initial programs

If the initial program that is invoked does not run under Language Environment, and you want to begin debugging before Language Environment is initialized, you must use the EQANMDBG program to start both Debug Tool and your user program.

You do not have to use EQANMDBG to initiate a Debug Tool session if the initial user program runs under control of the Language Environment, even if other parts of the program do not run under the Language Environment.

13.4.3 Debugging CICS non-LE initial programs

The initial program that you request in a DTCN or CADP debug profile to start up a non-Language Environment debugging session must be the first program started for the CICS transaction or the initial program that runs for an EXEC CICS LINK or XCTL.

If you set the debug profile to a non-Language Environment program that meets both of the following requirements, Debug Tool will not start:

- The program is called only statically.
- The program is called dynamically and does not cross a Link boundary by using EXEC CICS LOAD/CALL.

13.5 Debugging Assembler programs using Debug Tool

This section describes debugging Assembler programs using Debug Tool.

13.5.1 Preparing the program

In order to utilize the Debug Tool's full capabilities to debug an assembler program, you must first prepare the program using the following steps:

1. Assemble your program with the proper options.
2. Create the EQALANGX file.
3. Link-edit your program.

If you use Debug Tool Utilities to prepare your assembler program, you can combine steps 1 and 2 into a single step.

You must have Debug Tool Utilities and Advanced Functions installed on your system to prepare and debug assembler programs.

You can debug an assembler program several ways:

1. Using MFI Debugger
2. Using IBM Distributed Debugger
3. Using the Compiled Language Debugger component of WebSphere Studio Enterprise Developer

Before beginning to debug

When you debug an assembler program, you can use most of the Debug Tool commands. There are three differences between debugging an assembler program and debugging programs written in other programming languages supported by Debug Tool:

- After you assemble your program, you must create a debug information file, also called the EQALANGX file. Debug Tool uses this file to obtain information about your assembler program.
- Debug Tool assumes all compile units are written in some high-level language (HLL). You must inform Debug Tool that a compile unit is an assembler compile unit and instruct

Debug Tool to load the assembler compile unit's debug information. Do this by entering the LOADDEBUGDATA (or LDD) command.

- Assembler does not have language elements you can use to write expressions. Debug Tool provides assembler-like language elements you can use to write expressions for Debug Tool commands that require an expression. See *Debug Tool for z/OS Reference and Messages* for a description of the syntax of the assembler-like language.

After you verify that your assembler program meets these requirements, prepare your assembler program.

13.5.2 Preparing programs and modifying setup files with Debug Tool Utilities

The examples in the following sections show you how to use Debug Tool Utilities to prepare your programs and how to create, manage, and use a setup file. The examples guide you through the following tasks:

1. Creating personal data sets with the correct attributes.
2. Starting Debug Tool Utilities.
3. Compiling or assembling your program using Debug Tool Utilities. You must have Debug Tool Utilities and Advanced Functions installed on your system to run the steps in this task. If you do not have this product installed, you can build your program through your usual methods and resume the example with the next step.
4. Modifying and using a setup file to run your program in the foreground or in batch.

Creating personal data sets for assembler

Create the data sets with the names and attributes described in Table 13-3. Allocate 5 tracks for each of the data sets. Partitioned data sets should be specified with 5 blocks for the directory.

Table 13-3 Names and attributes to use when you create your own data sets

Data Set Name	LRECL	BKLSIZE	RECFM	DSORG
prefix.sample.asm	80	*	FB	PO

Copy the members of the hlq.SEQASAMP data set described in Table 13-4 into the personal data sets you just created.

Table 13-4 Description of samples

SEQASAMP member name	Sample data set name	Description of member
EQAWPP5	*prefix*.sample.asm(wpp5)	assembler source code
EQAWSU5	*prefix*.sample.asm(wsu5)	setup file for EQAWPP5

Starting Debug Tool Utilities

To start Debug Tool Utilities, do one the following:

- If Debug Tool Utilities was installed as an option on an existing ISPF panel, select that option.
- If Debug Tool Utilities data sets were installed as part of your log-on procedure, enter the following command from ISPF option 6:

```
EQASTART
```

- If Debug Tool Utilities was installed as a separate application, enter the following command from ISPF option 6:

  ```
  EX 'hlq.SEQAEXEC(EQASTART)'
  ```

The Debug Tool Utilities primary panel (EQA@PRIM) is displayed. On the command line, enter the PANELID command. This command displays the name of each panel in the upper left corner of the screen. These names are used as navigation aids in the instructions provided in this section. After you complete these examples, you can stop the display of these names by entering the PANELID command.

13.5.3 Assembling your program using Debug Tool Utilities

To do the steps in this task, you must have Debug Tool Utilities and Advanced Functions installed on your system. To assemble your program, perform the following steps:

1. Figure 13-90 shows the Main Panel of Debug Tool. Select option **1** to begin program preparation.

```
EQA@PRIM ------------------- Debug Tool Utilities ----------------------------
Option ===> 1_____________________________________________________________________

                                                                   More:     +
0  Manage Job Card
   For Program Preparation and Setup File Management

1  Program Preparation
   Compile old or new COBOL programs with newer compilers, convert old COBOL
   source into new COBOL source, use other compilers, and link edit.

2  Manage and Use Debug Tool Setup Files
   You can manage setup files and use them to run your program interactively
   with Debug Tool in TSO Foreground or submit your program to run in
   the background using MVS batch.

3  Code Coverage
   Measure code coverage in programs written in COBOL, PL/I, C/C++ and
   Assembler when compiled with specific IBM compilers and HLASM.

4  Manage IMS Programs

 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-90 Assembler program preparation selection

2. From the next screen, shown in Figure 13-91, select option **5** and press Enter to assemble using high-level assembler.

```
EQAPP ----------------- Debug Tool Program Preparation -----------------------
Option ===> 5_____________________________________________________________________

                                                                      More:     +
1  COBOL Compile
   Using 5655-G53 IBM Enterprise COBOL for z/OS and OS/390

2  COBOL Convert and Compile
   Using 5648-B05 COBOL and CICS Command Level Conversion Aid
   and   5655-G53 IBM Enterprise COBOL for z/OS and OS/390

3  PL/I  Compile
   Using 5655-H31  IBM(R) Enterprise PL/I for z/OS

4  C and C++ Compile
   Using 5694A01 z/OS C/C++

5  Assemble
   Using High Level Assembler

L  Link Edit
   Using z/OS Binder
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-91 Selecting option 5 to assemble using high level assembler

3. Prepare an assembler program. As shown in Figure 13-92, the following inputs are required:
   a. Location of your CEE library in the Syslib data set Name field.
   b. Source Library names:
      - Project: *prefix* (for this example, HLQ is `gracine`)
      - Group: `sample`
      - Type: `asm`
      - Member: `wpp5`
   c. Choose either foreground (F) or background (B) mode. In this example background was selected.
   d. Enter / to edit options and specify a naming pattern for the output data sets in the field Data set naming pattern. Press Enter.

      Note: In this sampling, CICS and DB2 are not part of this test, therefore, option N for no has been selected.

```
EQAPPC5 -----Debug Tool Program Preparation - High Level Assembler------------
Command ===> ____________________________________________________________________

Assemble using High Level Assembler
E Foreground or Batch Processing  (F B)
                                                                   More:     +
Specify primary input data set for compilation.
Source Library:
   Project . . . GRACINE
   Group . . . . SAMPLE  . . . ________ . . . ________ . . . ________
   Type  . . . . ASM
   Member  . . . WPP5        (Blank or pattern for member selection list)

Other Partitioned or Sequential Data Set:
   Data Set Name . . . ____________________________________________________
   Volume Serial . . . ______    (If not cataloged)

Data set name patterns, CICS and SQL processing options:
/ Enter / to edit options and data set name patterns
N CICS    S Separate CICS translator, N None
N DB2/SQL S Separate DB2 precompiler, N None

 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-92 Options panel for asm program preparation (part 1 of 2)

4. Press the PF8 key to scroll down the panel. This is important because the LE maclib library is required for the SYSLIB DDname, as shown in Figure 13-93.

```
EQAPPC5 -----Debug Tool Program Preparation - High Level Assembler------------
Command ===> ____________________________________________________________________

Assemble using High Level Assembler
E Foreground or Batch Processing  (F B)
                                                                   More:   - +

Assembler Options Data Set(ASMAOPT):
        ____________________________________________________
Assembler Options String:
____________________________________________________________________________________

Specify secondary input data sets for Copy processing:
DDname         Syslib data set Names
  SYSLIB  . . . CEE.SCEEMAC
  ________ . . . ____________________________________________
  ________ . . . ____________________________________________
  ________ . . . ____________________________________________
  ________ . . . ____________________________________________
  ________ . . . ____________________________________________
  ________ . . . ____________________________________________
  ________ . . . ____________________________________________
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-93 Options panel for asm program preparation (part 2 of 2)

5. Press PF3 to get back to the previous screen, then press Enter to display the edit options and data set name patterns as shown in Figure 13-94.

   PanelID eqappc5a is displayed. This is part one of a two-part panel displaying the options and data set name patterns.

```
EQAPPC5A igh Level Assembler - Edit options and data set name patterns--------
Command ===> _____________________________________________________________

Assemble using High Level Assembler
                                                                   More:     +

Specify data set name patterns for output data sets:
Pattern characters: /n, /B, /L, /M, /U, /P  Enter Help for usage information.
Assembler output data sets:
   Listing . '/U./B.ASMLIST(/M)'
   Object. . '/U./B.OBJECT(/M)'
   SYSADATA. '/U./B.SYSADATA(/M)'
   LangFile. '/U.EQALANGX(/M)'
   SYSTERM . *

CICS Translator and DB2 SQL Processor options (Integrated and Separate):
   CICS. . . CICS
   DB2 . . . HOST(ASM)

DB2 SQL Output DBRM data set (Integrated and Separate):
   DBRMLIB . '/U./B.DBRMLIB(/M)'

 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-94 Options and data set patterns display panel (part 1 of 2)

6. Press PF8 to scroll forward and display the second portion of this panel as shown in Figure 13-95.

```
EQAPPC5A igh Level Assembler - Edit options and data set name patterns--------
Command ===> _____________________________________________________________

Assemble using High Level Assembler
                                                                   More:   -
   LangFile. '/U.EQALANGX(/M)'
   SYSTERM . *

CICS Translator and DB2 SQL Processor options (Integrated and Separate):
   CICS. . . CICS
   DB2 . . . HOST(ASM)

DB2 SQL Output DBRM data set (Integrated and Separate):
   DBRMLIB . '/U./B.DBRMLIB(/M)'

CICS Translator data set name patterns (Separate):
   Listing . '/U./B.CICSCC(/M)'
   SYSTERM . *

DB2 Precompiler data set name patterns (Separate):
   Listing . '/U./B.DB2CC(/M)'
   SYSTERM . *
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-95 Options and data set patterns display panel (part 2 of 2)

7. Press Enter; the Verify Selections panel is displayed. as shown in Figure 13-96.

```
EQAPPC5B ----------High Level Assembler - Verify Selections-------------------
Command ===> _____________________________________________________________________

Assemble using High Level Assembler
                                                                      More:     +
Input data set: 'GRACINE.SAMPLE.ASM(WPP5)'

Assembler Options Data Set(ASMAOPT):
          ____________________________________________________________
Assembler Options:
ADATA
_____________________________________________________________________________________

These data set names were generated using patterns that were specified
in the "Edit data set name patterns" panel.  Override these names by
overtyping. These data sets will be created, if they do not exist.

  Assembler output data sets:
    Listing . 'GRACINE.SAMPLE.ASMLIST(WPP5)'
    Object. . 'GRACINE.SAMPLE.OBJECT(WPP5)'
    SYSADATA: 'GRACINE.SAMPLE.SYSADATA(WPP5)'
    LangFile: 'GRACINE.EQALANGX(WPP5)'
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-96 Verification selection panel displayed

8. Select F to process these programs in the foreground. Specify N for CICS translator and N for DB2 precompiler. None of these programs contain CICS or DB2 instructions. Press Enter.
9. Make a note of the data set name for Object compilation output. For an ASSEMBLER program, the data set name will look similar to the following name: *prefix*.SAMPLE.OBJECT(WPP5). You will use this name when you link your object modules (in this scenario, prefix HLQ is gracine).
10. Press Enter. Panel EQAPPA1 is displayed as shown in Figure 13-97.

```
EQAPPA1 ----------------------- Create data set-------------------------------
Command ===> _____________________________________________________________________


Specify parameters for Listing
   'GRACINE.SAMPLE.ASMLIST(WPP5)'

Data set Allocation Parameters:
 Data set name type. . . . PDSE___       (PDSE, PDS, or SEQ)

 Allocate command string:
CYLINDER SPACE(1 1) LRECL(137) RECFM(V B A)
_____________________________________________________________________________________

 Press  ENTER to create the data set
 Press  EXIT or CANCEL to cancel creation




 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-97 Panel display EQAPPA1, creating listing PDSE data set

11. Press Enter to create the specified parameters for the PDSE data set as shown in Figure 13-98.

```
EQAPPA1 ------------------------ Create data set-------------------------------
Command ===> _____________________________________________________________________


Specify parameters for Object
   'GRACINE.SAMPLE.OBJECT(WPP5)'

Data set Allocation Parameters:
 Data set name type. . . . PDSE___       (PDSE, PDS, or SEQ)

 Allocate command string:
CYLINDER SPACE(1 1)
_____________________________________________________________________________

 Press  ENTER to create the data set
 Press  EXIT or CANCEL to cancel creation





 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-98 Panel display EQAPPA1, creating object PDSE data set

12. Press Enter to create the SYSADATA PDSE data set as shown in Figure 13-99.

```
EQAPPA1 ------------------------ Create data set-------------------------------
Command ===> _____________________________________________________________________


Specify parameters for SYSADATA
   'GRACINE.SAMPLE.SYSADATA(WPP5)'

Data set Allocation Parameters:
 Data set name type. . . . PDSE___       (PDSE, PDS, or SEQ)

 Allocate command string:
CYLINDER SPACE(1 1)
_____________________________________________________________________________

 Press  ENTER to create the data set
 Press  EXIT or CANCEL to cancel creation





 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-99 Panel display EQAPPA1, creating sysadata PDSE data set

13. Press Enter to create the EQALANGX PDSE data set as shown in Figure 13-100.

```
EQAPPA1 ------------------------ Create data set-------------------------------
Command ===> _____________________________________________________________________


Specify parameters for LangFile
   'GRACINE.EQALANGX(WPP5)'

Data set Allocation Parameters:
 Data set name type. . . . PDSE    _       (PDSE, PDS, or SEQ)

 Allocate command string:
CYLINDER SPACE(1 1) LRECL(1562) RECFM(V B)
```

```
 Press  ENTER to create the data set
 Press  EXIT or CANCEL to cancel creation





 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-100 Panel display EQAPPA1, creating eqalangx PDSE data set

14. Press Enter once and a pop-up log screen will be displayed as shown in Figure 13-101.

```
EQAZ040I High Level Assembler started for 'GRACINE.SAMPLE.ASM(WPP5)' using 'GRA
CINE.SAMPLE.ASM(WPP5)'.
EQAZ041I High Level Assembler library: ASMA90 invoked.
EQAZ047I EQALANGX started for 'GRACINE.SAMPLE.SYSADATA(WPP5)'.
EQAZ048I EQALANGX EQALANGX invoked from 'ADTOOLS.DEBUG.TOOL51.SEQAMOD(EQALANGX)
'.
*** _
```

Figure 13-101 Pop-up message screen

15. Press Enter to continue. The View Outputs screen is displayed as shown in Figure 13-102.

```
EQAPPC5C -------------High Level Assembler - View Outputs---------------------
Command ===> ______________________________________________________________________
                                                                      More:     +
Assemble using High Level Assembler

DB2 Precompiler      return code = NOT RUN
CICS Translator      return code = NOT RUN
High Level Assembler return code = 0

The following data sets were used for this compilation.
Enter E to Edit, V to View, or B to browse these data sets.

  _ Input . : 'GRACINE.SAMPLE.ASM(WPP5)'

  Assembler output data sets:
  _ Listing : 'GRACINE.SAMPLE.ASMLIST(WPP5)'
  _ Object. : 'GRACINE.SAMPLE.OBJECT(WPP5)'
  _ SYSADATA: 'GRACINE.SAMPLE.SYSADATA(WPP5)'
  _ LangFile: 'GRACINE.EQALANGX(WPP5)'
  _ SYSTERM : *

N DB2 SQL output DBRM data set:
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-102 View outputs display screen

As you see, the return code result is displayed. A return code of 0 (zero) is always a very good sign. If the result is greater than a return code of 4 (four), proper problem determination is necessary to assure a successful execution.

Important: Accepted return codes are 0 (zero) and 4 (four).

16. Type a `b` in the Listing field and press Enter to browse through the listing file as shown in Figure 13-103.

```
EQAPPC5C -------------High Level Assembler - View Outputs---------------------
Command ===> ____________________________________________________________________
                                                                   More:     +
Assemble using High Level Assembler

DB2 Precompiler      return code = NOT RUN
CICS Translator      return code = NOT RUN
High Level Assembler return code = 0

The following data sets were used for this compilation.
Enter E to Edit, V to View, or B to browse these data sets.

  _ Input . : 'GRACINE.SAMPLE.ASM(WPP5)'

  Assembler output data sets:
  b Listing : 'GRACINE.SAMPLE.ASMLIST(WPP5)'
  _ Object. : 'GRACINE.SAMPLE.OBJECT(WPP5)'
  _ SYSADATA: 'GRACINE.SAMPLE.SYSADATA(WPP5)'
  _ LangFile: 'GRACINE.EQALANGX(WPP5)'
  _ SYSTERM : *

N DB2 SQL output DBRM data set:
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-103 Browsing the listing data set

17. The next five figures show the beginning and the end of the listing file. Due to the length of it (more than 25 screens of information), only the options for the Assembly and the last pages are shown in this series of figures. Press PF8 repeatedly to scroll through the file.

```
  Menu  Utilities  Compilers  Help
 ___________________________________________________________________________
 ISRBROBA  GRACINE.SAMPLE.ASMLIST(WPP5)              Line 00000000 Col 001 080
 Command ===> ____________________________________________ Scroll ===> PAGE
******************************** Top of Data ***********************************
                                   High Level Assembler Option Summary

  No Overriding ASMAOPT Parameters
  Overriding Parameters-  ADATA
  No Process Statements


  Options for this Assembly
3   ADATA
    ALIGN
    ASA
    BATCH
    CODEPAGE(047C)
  NOCOMPAT
  NODBCS
  NODECK
    DXREF
 F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
F10=Left   F11=Right  F12=Cancel
```

Figure 13-104 Panelid ISRBROBA, Listing data set display (Part 1 of 5)

```
  Menu  Utilities  Compilers  Help
 ISRBROBA  GRACINE.SAMPLE.ASMLIST(WPP5)              Line 00000018 Col 001 080
 Command ===> _____________________________________________ Scroll ===> PAGE
    ESD
  NOEXIT
    FLAG(0,ALIGN,CONT,EXLITW,NOIMPLEN,NOPAGE0,PUSH,RECORD,NOSUBSTR,USING0)
  NOFOLD
  NOGOFF
  NOINFO
    LANGUAGE(EN)
  NOLIBMAC
    LINECOUNT(60)
    LIST(121)
    MXREF(SOURCE)
    OBJECT
    OPTABLE(UNI,NOLIST)
  NOPCONTROL
  NOPESTOP
  NOPROFILE
  NORA2
  NORENT
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 13-105 Panelid ISRBROBA, Listing data set display (Part 2 of 5)

```
  Menu  Utilities  Compilers  Help
 ISRBROBA  GRACINE.SAMPLE.ASMLIST(WPP5)              Line 00000036 Col 001 080
 Command ===> _____________________________________________ Scroll ===> PAGE
    RLD
    RXREF
    SIZE(MAX)
    SYSPARM()
  NOTERM
  NOTEST
    THREAD
  NOTRANSLATE
    TYPECHECK
    USING(NOLIMIT,MAP,WARN(15))
    XREF(SHORT,UNREFS)

  No Overriding DD Names
                                                External Symbol Dictionary
Symbol   Type   Id     Address  Length     LD ID  Flags Alias-of
WPP5      SD 00000001 00000000 0000019C             07
CEEINT    ER 00000002
CEEMAIN   SD 00000003 000001A0 00000008             07
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 13-106 Panelid ISRBROBA, Listing data set display (Part 3 of 5)

18. Type the letter `M` on the command line and press PF8 to jump to the last page of the listing report as shown in Figure 13-107 (Part 4 of 5).

```
  Menu  Utilities  Compilers  Help
 ISRBROBA  GRACINE.SAMPLE.ASMLIST(WPP5)                Line 00000036 Col 001 080
 Command ===> m_____________________________________________ Scroll ===> PAGE
    RLD
    RXREF
    SIZE(MAX)
    SYSPARM()
  NOTERM
  NOTEST
    THREAD
  NOTRANSLATE
    TYPECHECK
    USING(NOLIMIT,MAP,WARN(15))
    XREF(SHORT,UNREFS)

  No Overriding DD Names
                                                External Symbol Dictionary
Symbol   Type   Id     Address  Length     LD ID  Flags Alias-of
WPP5      SD 00000001 00000000 0000019C             07
CEEINT    ER 00000002
CEEMAIN   SD 00000003 000001A0 00000008             07
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 13-107 Panelid ISRBROBA, Listing data set display (Part 4 of 5)

The last page of the listing report is shown in Figure 13-108 (Part 5 of 5).

```
  Menu  Utilities  Compilers  Help
 ISRBROBA  GRACINE.SAMPLE.ASMLIST(WPP5)                Line 00000865 Col 001 080
 Command ===> _____________________________________________ Scroll ===> PAGE
     No Statements Flagged in this Assembly
HIGH LEVEL ASSEMBLER, 5696-234, RELEASE 4.0, PTF UQ89875
SYSTEM: z/OS 01.04.00             JOBNAME: GRACINE     STEPNAME: TPROC02    PRO
Datasets Allocated for this Assembly
 Con DDname   Dataset Name                                Volume  Member
  P1 SYSIN    GRACINE.SAMPLE.ASM                          STF603  WPP5
  L1 SYSLIB   CEE.SCEEMAC                                 IPL14D
     SYSADATA GRACINE.SAMPLE.SYSADATA                     STF622  WPP5
     SYSLIN   GRACINE.SAMPLE.OBJECT                       STFS61  WPP5
     SYSPRINT GRACINE.SAMPLE.ASMLIST                      STF614  WPP5

   8608K allocated to Buffer Pool,          261K would be required for this to b
    111 Primary Input Records Read         1657 Library Records Read
      0 ASMAOPT Records Read                881 Primary Print Records Written
     17 Punch Records Written              1413 ADATA Records Written
Assembly Start Time: 11.19.41 Stop Time: 11.19.42 Processor Time: 00.00.00.0279
Return Code 000
********************************* Bottom of Data *********************************
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 13-108 Panelid ISRBROBA, Listing data set display (Part 5 of 5)

19. Press PF3 as many times as necessary (this will be at least 3 times), until you return to the main panel of Debug Tool as shown in Figure 13-109.

```
EQA@PRIM -------------------- Debug Tool Utilities ----------------------------
Option ===> 1_____________________________________________________________________

                                                                    More:     +
0  Manage Job Card
   For Program Preparation and Setup File Management

1  Program Preparation
   Compile old or new COBOL programs with newer compilers, convert old COBOL
   source into new COBOL source, use other compilers, and link edit.

2  Manage and Use Debug Tool Setup Files
   You can manage setup files and use them to run your program interactively
   with Debug Tool in TSO Foreground or submit your program to run in
   the background using MVS batch.

3  Code Coverage
   Measure code coverage in programs written in COBOL, PL/I, C/C++ and
   Assembler when compiled with specific IBM compilers and HLASM.

4  Manage IMS Programs

 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-109 Panelid EQA@PRIM

20. From the main panel of Debug Tool, select option **1** again and press Enter; the panel shown in Figure 13-110 is displayed.

```
EQAPP ----------------- Debug Tool Program Preparation -----------------------
Option ===> L_____________________________________________________________________

                                                                    More:     +
1  COBOL Compile
   Using 5655-G53 IBM Enterprise COBOL for z/OS and OS/390

2  COBOL Convert and Compile
   Using 5648-B05 COBOL and CICS Command Level Conversion Aid
   and   5655-G53 IBM Enterprise COBOL for z/OS and OS/390

3  PL/I  Compile
   Using 5655-H31  IBM(R) Enterprise PL/I for z/OS

4  C and C++ Compile
   Using 5694A01 z/OS C/C++

5  Assemble
   Using High Level Assembler

L  Link Edit
   Using z/OS Binder
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-110 Panelid EQAPP

21. Select option **L** to link the object modules. The panel in Figure 13-111 is displayed.

```
EQAPPCL ----------Debug Tool Program Preparation - Link Edit------------------
Command ===> ____________________________________________________________________

Link Edit using z/OS Binder
E Foreground or Batch Processing  (F B)
                                                                   More:     +
Specify primary input data set for link edit.
Source Library:
   Project . . . ________
   Group . . . . ________ . . . ________ . . . ________ . . . ________
   Type  . . . . ________
   Member  . . . ________        (Blank or pattern for member selection list)

Other Partitioned or Sequential Data Set:
   Data Set Name . . . ______________________________________________________
   Volume Serial . . . ______    (If not cataloged)

Data set name patterns:
_ Enter / to edit data set name patterns

Link Edit Options:
____________________________________________________________________________
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-111 Panelid EQAPPCL, link edit display

22. In this panel, specify `F` to process the programs in the foreground. Then make the following entries, as shown in Figure 13-112:
    - Project = *`prefix`* (for this sampling, HLQ used is *gracine*)
    - Group = `SAMPLE`
    - Type = `OBJECT`
    - Member = `WPP5`

    Press PF8 to scroll to the lower part of this panel, shown in Figure 13-113. Specify the names of the other libraries you need to link to your program:
    - In the field Syslib data set Name, specify the prefix of your CEE library: `CEE.SCEELKED`

```
EQAPPCL ----------Debug Tool Program Preparation - Link Edit------------------
Command ===> ____________________________________________________________________

Link Edit using z/OS Binder
E Foreground or Batch Processing  (F B)
                                                                   More:     +
Specify primary input data set for link edit.
Source Library:
   Project . . . GRACINE
   Group . . . . SAMPLE  . . . ________ . . . ________ . . . ________
   Type  . . . . OBJECT
   Member  . . . WPP5____        (Blank or pattern for member selection list)

Other Partitioned or Sequential Data Set:
   Data Set Name . . . ______________________________________________________
   Volume Serial . . . ______    (If not cataloged)

Data set name patterns:
_ Enter / to edit data set name patterns

Link Edit Options:
____________________________________________________________________________
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-112 Panelid EQAPPCL (Part 1 of 2)

```
EQAPPCL ----------Debug Tool Program Preparation - Link Edit------------------
Command ===> _____________________________________________________________

Link Edit using z/OS Binder
F Foreground or Batch Processing  (F B)
                                                                   More:   -
_____________________________________________________________________________

Specify secondary input data sets for Include processing:
DDname          Syslib data set Names
  SYSLIB  . . . CEE.SCEELKED_
  ________ . . . ____________________________________________
  ________ . . . ____________________________________________
  ________ . . . ____________________________________________
  ________ . . . ____________________________________________
  ________ . . . ____________________________________________
  ________ . . . ____________________________________________
  ________ . . . ____________________________________________
  ________ . . . ____________________________________________
  ________ . . . ____________________________________________
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-113 Panelid EQAPPCL (Part 2 of 2)

```
EQAPPCLB ----------------Link Edit - Verify Selections------------------------
Command ===> _____________________________________________________________

Link Edit using z/OS Binder

Input data set: 'GRACINE.SAMPLE.OBJECT(WPP5)'

Link Edit Options:
_____________________________________________________________________________

These data set names were generated using patterns that were specified
in the "Edit data set name patterns" panel.  Override these names by
overtyping. These data sets will be created, if they do not exist.

    Listing . 'GRACINE.SAMPLE.LINKLIST(WPP5)'
    Load. . . 'GRACINE.SAMPLE.LOAD(WPP5)'
    SYSTERM . *

 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-114 Panelid EQAPPCB

23. From panel EQAPPCLB, shown in Figure 13-114, make a note of the data set name in the Load link-edit output field. You will use this name when you modify a setup file. Press Enter; the Listing data set will be created as shown in Figure 13-115.

```
EQAPPA1 ------------------------ Create data set-------------------------------
Command ===> _____________________________________________________________________


Specify parameters for Listing
   'GRACINE.SAMPLE.LINKLIST(WPP5)'

Data set Allocation Parameters:
 Data set name type. . . . PDSE___       (PDSE, PDS, or SEQ)

 Allocate command string:
CYLINDER SPACE(1 1)
_____________________________________________________________________________

 Press  ENTER to create the data set
 Press  EXIT or CANCEL to cancel creation





 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-115 Panelid EQAPPA1

24. Press Enter. The Load data set is created as shown in Figure 13-116.

```
EQAPPA1 ------------------------ Create data set-------------------------------
Command ===> _____________________________________________________________________


Specify parameters for Load
   'GRACINE.SAMPLE.LOAD(WPP5)'

Data set Allocation Parameters:
 Data set name type. . . . PDSE___       (PDSE, PDS, or SEQ)

 Allocate command string:
CYLINDER SPACE(1 1)
_____________________________________________________________________________

 Press  ENTER to create the data set
 Press  EXIT or CANCEL to cancel creation





 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-116 Panelid EQAPPA1

25. Press Enter to view the display log screen shown in Figure 13-117.

```
EQAZ040I z/OS Binder started for 'GRACINE.SAMPLE.OBJECT(WPP5)' using 'GRACINE.S
AMPLE.OBJECT(WPP5)'.
EQAZ041I z/OS Binder library: IEWL invoked.
*** _
```

Figure 13-117 Pop-up display log screen

26. Press Enter to clear the pop-up display log screen; panelid EQAPPCLC is displayed as shown in Figure 13-118.

```
EQAPPCLC ------------------Link Edit - View Outputs--------------------------
Command ===> ____________________________________________________________________


Link Edit using z/OS Binder
Link Edit return code = 0

The following data sets were used for this link edit.
Enter E to Edit, V to View, or B to browse these data sets.

  _ Input . : 'GRACINE.SAMPLE.OBJECT(WPP5)'

  _ Listing : 'GRACINE.SAMPLE.LINKLIST(WPP5)'
  _ Load. . : 'GRACINE.SAMPLE.LOAD(WPP5)'
  _ SYSTERM : *





 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-118 Panelid EQAPPCLC, view outputs

27. In panel EQAPPCLC, check for a 0 return code. Type a `V` in the Listing field and press Enter; Figure 13-119 is displayed.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------
ISREDDE2   GRACINE.SAMPLE.LINKLIST(WPP5) - 01.00          Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
****** ***************************** Top of Data ******************************
000001 1z/OS V1 R3 BINDER     13:47:04 TUESDAY NOVEMBER 30, 2004
000002  BATCH EMULATOR  JOB(GRACINE ) STEP(TPROC02 ) PGM= IEWL      PROCEDURE(
000003
000004
000005
000006
000007 1PROCESSING OPTIONS:
000008
000009     ALIASES              NO
000010     ALIGN2               NO
000011     AMODE                UNSPECIFIED
000012     CALL                 YES
000013     CASE                 UPPER
000014     COMPAT               UNSPECIFIED
000015     DCBS                 NO
000016     DYNAM                NO
000017     EXTATTR              UNSPECIFIED
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 13-119 Listing data set

28. In panel ISREDDE2, review the messages. After you review the messages, press PF3 (Exit). Press PF3 repeatedly to return to the main Debug Tool Utilities screen.

13.5.4 Modifying and using a setup file

This section describes how to modify a setup file and then use it to run the examples in the TSO foreground or run the examples in the background by submitting an MVS batch job.

Run the program in foreground

To modify and run the setup file so your program runs in the foreground, perform the following steps:

1. In panel EQA@PRIM, select option **2** as shown in Figure 13-120. Press Enter.

```
EQA@PRIM -------------------- Debug Tool Utilities ----------------------------
Option ===> 2_____________________________________________________________________

                                                                      More:     +
0  Manage Job Card
   For Program Preparation and Setup File Management

1  Program Preparation
   Compile old or new COBOL programs with newer compilers, convert old COBOL
   source into new COBOL source, use other compilers, and link edit.

2  Manage and Use Debug Tool Setup Files
   You can manage setup files and use them to run your program interactively
   with Debug Tool in TSO Foreground or submit your program to run in
   the background using MVS batch.

3  Code Coverage
   Measure code coverage in programs written in COBOL, PL/I, C/C++ and
   Assembler when compiled with specific IBM compilers and HLASM.

4  Manage IMS Programs

 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-120 Selecting option 2 to manage setup files

2. In panel EQAPFOR, shown in Figure 13-121, make the following entries:
   - Project = *prefix* (in this case, HLQ is GRACINE)
   - Group = `SAMPLE`
   - Type = `DTSF`
   - Member = `WSU5`

   Press Enter.

```
EQAPFOR ----------- Debug Tool Foreground - Edit Setup File ------------------
Command ===> ______________________________________________________________________


Setup File Library:
   Project . . . GRACINE
   Group . . . . TRADER   . . . ________ . . . ________ . . . ________
   Type  . . . . DTSF
   Member  . . . EQAWSU5        (Blank or pattern for member selection list)
                                (or existing or new member name)

Other Data Set Name:
   Data Set Name . . . 'GRACINE.SAMPLE.ASM(EQAWSU5)'
   Volume Serial . . . ______    (If not cataloged)


_ Initialize New setup file for DB2 (/)




 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-121 Panelid EQAPFOR

3. In panel EQAPFORS, shown in Figure 13-122, do the following:
   a. Replace &LOADDS. with the name of the load data mentioned in Step 9 of linking your object modules from the previous section.

b. Replace &EQAPRFX. with the prefix of your EQAW (Debug Tool) library.

c. Replace &CEEPRFX. with the prefix of your CEE (Language Environment) library.

```
EQAPFORS it Setup File 'GRACINE.SAMPLE.ASM(EQAWSU5)'          Row 1 to 5 of 8
Command ===> _____________________________________________ Scroll ===> CSR

Modify information and use the  Run command to execute,
        or the  Submit command to submit to Batch.
     Press HELP for a list of all available commands.

Load Module Name WPP5
Choose the format of your parameter string:
2 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs    - Run-time Options  / Program Arguments

/ Enter / to modify parameters 'TEST(ALL,CMDS,PROMPT,MFI:*)/'

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)            DISP
___ ________ ___ _ ***************** Top of Data ******************** _____
___ CMDS     1   _ SYSIN                                             _____
___ INSPLOG  1   _ SYSOUT=*                                          _____
___ STEPLIB  1   _ 'GRACINE.SAMPLE.LOAD'                             SHR
___ ________ 2   _ 'ADTOOLS.DEBUG.TOOL51.SEQAMOD'                    SHR
 F1=Help      F3=Exit      F4=Run       F7=Backward  F8=Forward  F10=Submit
F12=Cancel
```

Figure 13-122 Panelid EQAPFORS (part 1 of 2)

4. Press PF8 to scroll to the lower portion of panelid EQAPFORS, shown in Figure 13-123. to review the entries.

```
EQAPFORS it Setup File 'GRACINE.SAMPLE.ASM(EQAWSU5)'          Row 6 to 8 of 8
Command ===> _____________________________________________ Scroll ===> CSR

Modify information and use the  Run command to execute,
        or the  Submit command to submit to Batch.
     Press HELP for a list of all available commands.

Load Module Name WPP5
Choose the format of your parameter string:
2 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs    - Run-time Options  / Program Arguments

_ Enter / to modify parameters 'TEST(ALL,CMDS,PROMPT,MFI:*)/'

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)            DISP
___ ________ 3   _ 'CEE.SCEERUN'                                     SHR
___ SYSOUT   1   _ SYSOUT=*                                          _____
___ SYSPRINT 1   _ SYSOUT=*                                          _____
****************************** Bottom of data ******************************

 F1=Help      F3=Exit      F4=Run       F7=Backward  F8=Forward  F10=Submit
F12=Cancel
```

Figure 13-123 Panelid EQAPFORS (part 2 of 2)

5. Press PF7 to scroll up; press Enter to go to panelid EQAPFPRM shown in Figure 13-124.

```
EQAPFPRM ------ Debug Tool Foreground - Modify Parameter String --------------
Command ===> _____________________________________________________________
                                                                   More:     +
Modify Test options, other run-time options, and Program arguments

Select Test Options:

Test Option      ==> TEST            Test/Notest
Test Level       ==> ALL             All/Error/None
Commands File                        *, DDname, or Data Set Name
                 ==> CMDS
Prompt Level     ==> PROMPT          Prompt/NoPrompt
Preference File                      *, DDname, or Data Set Name
                 ==> *

Select (/) a session type and provide parameters:

 / Full-screen mode
   Terminal LU      ==> ____________ blank or MFI VTAM Terminal LU

 _ Remote debug mode
   Connection type ==> SINGLE       SINGLE/MULTIPLE socket
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-124 Panelid EQAPFPRM (part 1 of 2)

6. Press PF8 to scroll down as shown in Figure 13-125.

```
EQAPFPRM ------ Debug Tool Foreground - Modify Parameter String --------------
Command ===> _____________________________________________________________
                                                                   More:   -

 / Full-screen mode
   Terminal LU      ==> ____________ blank or MFI VTAM Terminal LU

 _ Remote debug mode
   Connection type ==> SINGLE       SINGLE/MULTIPLE socket
   Address         ==> ____________________________________________________
   Port            ==> 8001

Other run-time options: ____________________________________________________
____________________________________________________________________________
____________________________________________________________________________

Program arguments: _________________________________________________________
____________________________________________________________________________
____________________________________________________________________________



 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-125 Panelid EQAPFPRM (part 2 of 2)

7. Press PF3 to return to panelid EQAPFORS. Enter e in the Cmd field next to DD Name CMDS as shown in Figure 13-126.
8. In the window that is displayed, if there is a QUIT ; statement at the end of the data set, remove it. Press PF3 (Exit).

```
EQAPFORS it Setup File 'GRACINE.SAMPLE.ASM(EQAWSU5)'          Row 1 to 5 of 8
Command ===> ____________________________________________ Scroll ===> CSR

Modify information and use the  Run command to execute,
        or the  Submit command to submit to Batch.
     Press HELP for a list of all available commands.

Load Module Name WPP5
Choose the format of your parameter string:
2 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs    - Run-time Options  / Program Arguments

_ Enter / to modify parameters 'TEST(ALL,'CMDS',PROMPT,'MFI:*') /'

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)          DISP
___ ________ ___ _ ***************** Top of Data ********************
e_  CMDS     1   _ SYSIN
___ INSPLOG  1   _ SYSOUT=*
___ STEPLIB  1   _ 'GRACINE.SAMPLE.LOAD'                            SHR
___ ________ 2   _ 'ADTOOLS.DEBUG.TOOL51.SEQAMOD'                   SHR
 F1=Help      F3=Exit      F4=Run       F7=Backward  F8=Forward  F10=Submit
F12=Cancel
```

Figure 13-126 Editing DDname CMDS

9. Once in edit mode, remove the quit and return to panelid EQAPFORS by pressing PF3 as shown in Figure 13-127.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help

ISREDDE2   GRACINE.DTSUTEMP.XIN1D007                     Columns 00001 00072
Command ===> ____________________________________________ Scroll ===> CSR
****** **************************** Top of Data ******************************
000001         LDD WPP5 ;
000002         Comment Simple test of WPP5 ;
000003         AT TERMINATION
000004          DO ;
000005           Q LOC ;
000006          END ;
000007         AT 49
000008          DO ;
000009           LIST STORAGE (MSG2_ST, 18) ;
000010          END ;
000011         GO ;
****** **************************** Bottom of Data ***************************

 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 13-127 Editing temporary DDname CMDS file

10. Type **run** in the command line as shown in Figure 13-128. Press Enter.

```
EQAPFORS it Setup File 'GRACINE.SAMPLE.ASM(EQAWSU5)'          Row 1 to 5 of 8
Command ===> run                                            Scroll ===> CSR

Modify information and use the  Run command to execute,
        or the  Submit command to submit to Batch.
     Press HELP for a list of all available commands.

Load Module Name WPP5
Choose the format of your parameter string:
2 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs    - Run-time Options  / Program Arguments

_ Enter / to modify parameters 'TEST(ALL,'CMDS',PROMPT,'MFI:*') /'

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)          DISP
___ ________ ___ _ ***************** Top of Data ********************
___ CMDS     1   _ SYSIN
___ INSPLOG  1   _ SYSOUT=*
___ STEPLIB  1   _ 'GRACINE.SAMPLE.LOAD'                            SHR
___ ________ 2   _ 'ADTOOLS.DEBUG.TOOL51.SEQAMOD'                   SHR
 F1=Help      F3=Exit      F4=Run       F7=Backward  F8=Forward  F10=Submit
F12=Cancel
```

Figure 13-128 Execute RUN command (or can use PF4)

11. Debug Tool is started and the Debug Tool window is displayed. Enter any valid Debug Tool commands to verify that you can debug the program. Enter **qq** in the command line to stop Debug Tool and close the Debug Tool window.
12. In panel EQAPFORS, check the return code message:

    For the assembler program, the return code (RC) is 0. Press PF3 (Exit). All the changes made to the setup file are saved.
13. In panel EQAPFOR, press PF3 (Exit) to return to the main Debug Tool Utilities screen.

13.5.5 Using MainFrame Interface debugger

Mainframe interface debugger is activated using the following LU modify parameters in test:

```
PARM:    'TEST(ALL,'CMDS',PROMPT,'MFI:*')/'
```

When the Debug tool is started the first screen displayed is as shown in Figure 13-129.

```
Assemble LOCATION: WPP5 :> 49
Command ===> _                                                  Scroll ===> PAG
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 0 OF
****************************** TOP OF MONITOR ******************************
****************************** BOTTOM OF MONITOR ***************************



SOURCE: WPP5 +----1----+----2----+----3----+----4----+----5---- LINE: 85 OF 62
      49  000000B4 4110 B0C0      +         LA    1,CEET0002                Ge
      50  000000B8 58F0 B198      +         L     15,=V(CEETREC)            Ge
      51  000000BC 05EF           +         BALR  14,15                     Ca
      52  000000C0                +CEET0002 DC    A(*+8)                Parm 1
      53  000000C4                +         DC    A(*+8+X'80000000')    Parm 2
      54  000000C8                +         DC    A(0)                  Enc Mod
LOG 0----+----1----+----2----+----3----+----4----+----5----+---- LINE: 13 OF 1
0013    END ;
0014  GO ;
0015 1210C80A  E6D7D7F5 404040C8 C5D3D3D6 40E6D6D9   *WPP5   HELLO WOR*
0016 1210C81A  D3C4                                  *LD              *
PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEA
PF  7:UP         8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEV
```

Figure 13-129 Debug Tool, using MFI

You can quit the Debug Tool at any time by entering **QQ** on the command line.

13.5.6 Using IBM Distributed Debugger

This section demonstrates how to use the IBM Distributed Debugger when activating Debug Tool.

Debugging in remote debug mode

Debug Tool can run in remote debug mode by using TCP/IP to connect to a remote debugger installed on your workstation.

Workstation information needed

Using the Command Prompt, perform the IPCONFIG text-based function as shown in Figure 13-130.

```
Command Prompt
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Administrator>ipconfig

Windows IP Configuration


Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . : ibm.com
        IP Address. . . . . . . . . . . . : 172.20.101.106
        Subnet Mask . . . . . . . . . . . : 255.255.0.0
        Default Gateway . . . . . . . . . : 172.20.1.1

Ethernet adapter AGN Virtual Network Adapter:

        Connection-specific DNS Suffix  . : ibm.com
        IP Address. . . . . . . . . . . . : 9.28.193.194
        Subnet Mask . . . . . . . . . . . : 255.255.255.255
        Default Gateway . . . . . . . . . :

C:\Documents and Settings\Administrator>_
```

Figure 13-130 IP address required from the Command Prompt text-based ipconfig command

In this case the IP address is 9.28.193.194; it will be used in the parameter shown in Figure 13-131.

```
EQAPFPRM ------ Debug Tool Foreground - Modify Parameter String --------------
Command ===> ____________________________________________________________________
                                                                   More:   -

 _ Full-screen mode
   Terminal LU      ==> ____________ blank or MFI VTAM Terminal LU

 / Remote debug mode
   Connection type ==> SINGLE__      SINGLE/MULTIPLE socket
   Address         ==> 9.28.193.194
   Port            ==> 8001_______

Other run-time options: ____________________________________________________

Program arguments: _________________________________________________________

 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13-131 IP & Port requirements

The next requirement is to have the IBM Remote Distributed Debugger active on your workstation, as shown in Figure 13-132.

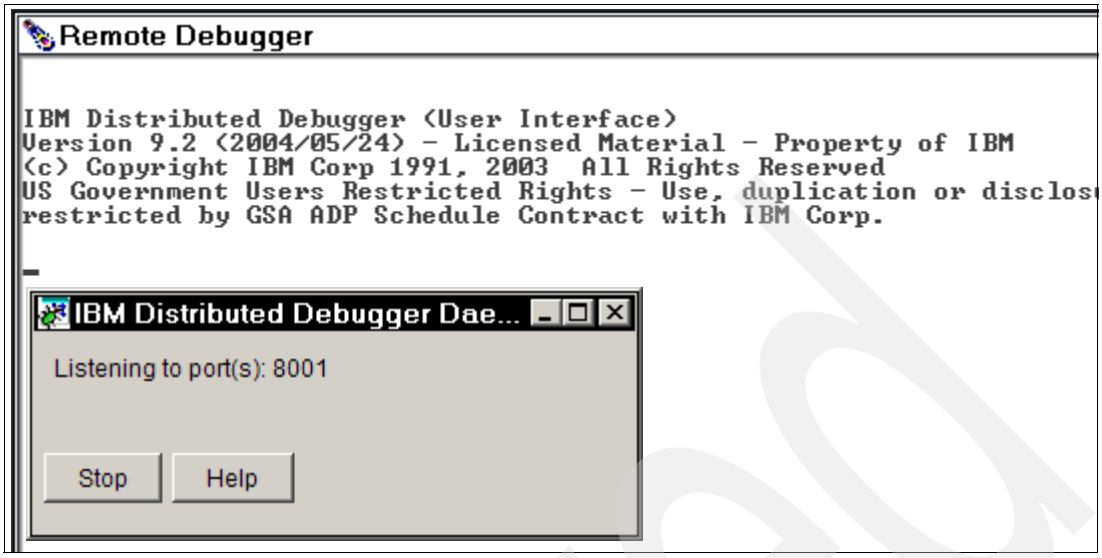


Figure 13-132 IBM Remote Distributed Debugger activated

Debug Tool is activated with the IBM Remote Distributed Debugger, using the TCPIP address as shown in Figure 13-133.

```
EQAPFORS it Setup File 'GRACINE.SAMPLE.ASM(EQAWSU5)'          Row 1 to 5 of 8
Command ===> _____________________________________________ Scroll ===> CSR

Modify information and use the  Run command to execute,
        or the  Submit command to submit to Batch.
     Press HELP for a list of all available commands.

Load Module Name WPP5
Choose the format of your parameter string:
2 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs    - Run-time Options  / Program Arguments

_ Enter / to modify parameters 'TEST(ALL,'*',PROMPT,'TCPIP&9.28.193.194%8001:*
) /'

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)            DISP
___ ________ ___ _ ****************** Top of Data ********************  _____
___ CMDS     1   _ SYSIN                                              _____
___ INSPLOG  1   _ SYSOUT=*                                           _____
___ STEPLIB  1   _ 'GRACINE.SAMPLE.LOAD'                              SHR
___ ________ 2   _ 'ADTOOLS.DEBUG.TOOL51.SEQAMOD'                     SHR
 F1=Help      F3=Exit      F4=Run       F7=Backward  F8=Forward  F10=Submit
F12=Cancel
```

Figure 13-133 Activating Debug Tool with IBM Remote Distributed Debugger parameters

Press PF4 or type **run** on the command line; the screen shown in Figure 13-134 is displayed.

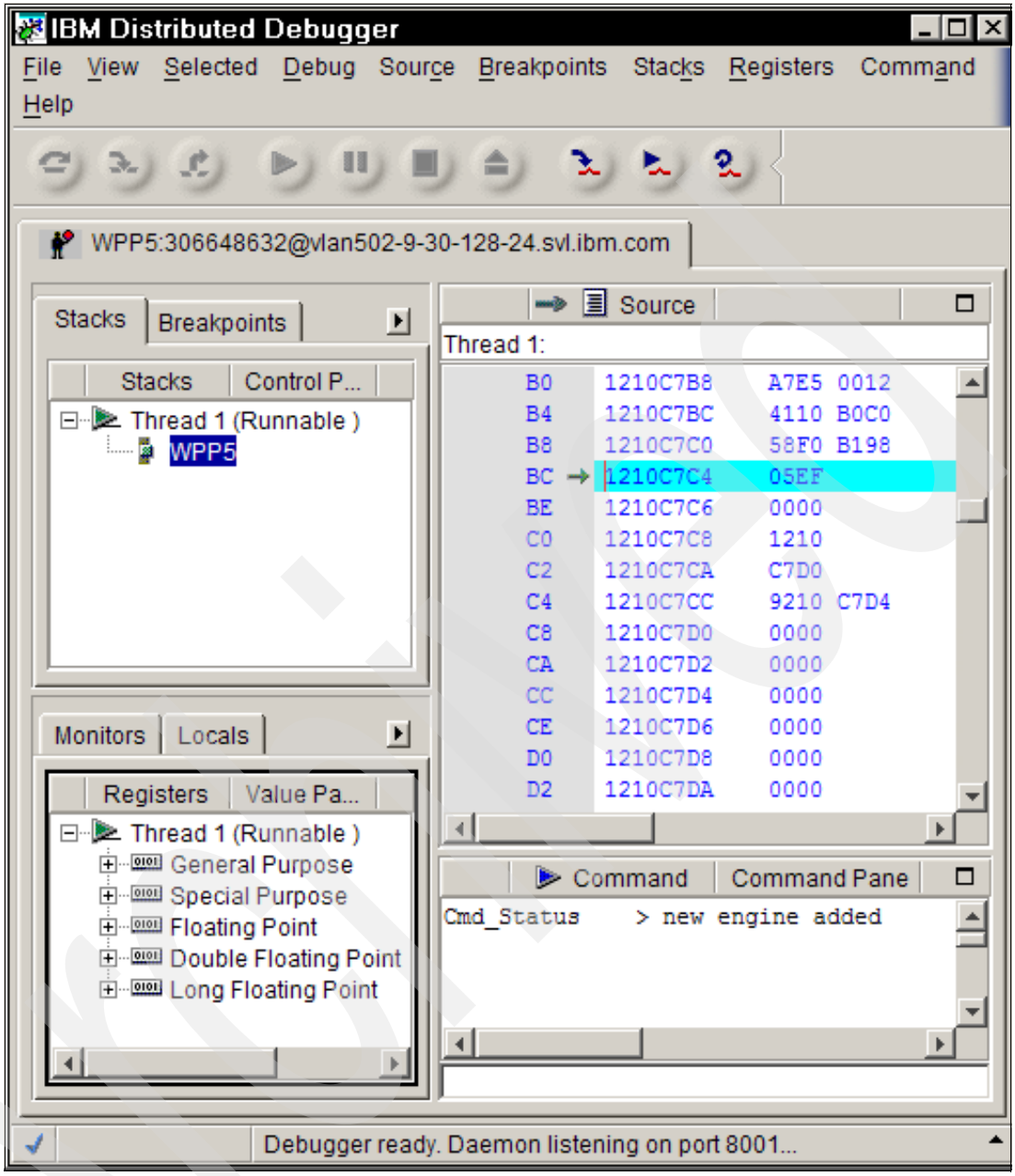


Figure 13-134 Using IBM Remote Distributed Debugger for an assembler program

13.5.7 Debug Tool commands supported in remote debug mode

The following remote debuggers can be used:

- IBM Distributed Debugger (also known as VisualAge Remote Debugger)
- Compiled Language Debugger component of WebSphere Studio Enterprise Developer

These remote debuggers support single and multiple connection types, except for earlier versions of the IBM Distributed Debugger.

If you are using a version of IBM Distributed Debugger that is earlier than version 9.2 (copyright date 2003/10/19), you must specify multiple connection type. Specifying a single connection type is recommended because it uses less resource and avoids security problems

when you use a firewall. Specify single connection type by using the TCPIP& of the TEST run-time option; specify multiple connection type by using the VADTCPIP& suboption of the TEST run-time option.

When you specify the TCPIP& or VADTCPIP& suboption, you must specify the port number that the remote debugger is using to listen for a debug session. By default, IBM Distributed Debugger uses port 8001 and the Compiled Language Debugger component of WebSphere Studio Enterprise Developer uses port 8001.

When you use remote debug mode, consider the following possible errors:

- The tcpip_workstation_id or port_id parameters must be syntactically and functionally correct.

  If they are not and you try to start a remote debug mode session, Debug Tool starts a full-screen mode session.

  For example, if you try to start a remote debug mode session from TSO or a CICS program using incorrect parameters, a full-screen mode session is displayed on your 3270-type terminal.

  This error is recorded in the MVS SDSF log as an allocation failure.
- If the tcpip_workstation_id or port_id parameters are not syntactically and functionally correct and you try to debug batch program, Debug Tool terminates and the batch program runs as though no debug session was started. This error occurs when, for example, you run a JES batch job or CICS batch transaction.

  This error is recorded in the MVS SDSF log as an allocation failure.
- If your z/OS or OS/390 environment is not using the default TCP/IP data set named TCPIP.TCPIP.DATA and you try to start a remote debug mode session to debug a batch program, Debug Tool terminates.

  The batch program runs as though no debug session was started. This error is recorded in the MVS SDSF log as an allocation error.

  To fix this error, specify the SYSTCPD DDNAME with the appropriate TCP/IP data set name, for example, //SYSTCPD DD DISP=SHR,DSN=MY.TCPIP.DATA.
- For TCP/IP sessions, the remote debug daemon must be started at the workstation before you start Debug Tool. Refer to the remote debugger information for help in using the remote debug daemon.

14

Using Debug Tool under UNIX System Services

This chapter provides a small example of compiling for and using Debug Tool in the UNIX System Services (USS) environment. Enterprise PL/I, Enterprise COBOL, and all C/C++ compilers shipped with the operating system support compilation under the UNIX System Services shell. The example used here to demonstrate the required steps is a C/C++ DLL one.

This chapter covers compilation, invoking Debug Tool in this environment, and some restrictions on what can be done; it does not go into debugging the application.

Conventions used

For the purposes of this chapter, all examples use product-provided utilities and we assume they are properly customized. It is also assumed that Debug Tool is available in linklist.

If Debug Tool is not available in linklist, you must set up the appropriate STEPLIB prior to trying these examples.

14.1 Common considerations for all languages

The UNIX System Services (USS) shell enforces re-entrancy, so non-reentrant programs should not be run under the USS shell.

The default single screen debugging cannot be used when debugging USS applications. The VTAM interface or the remote debugger interface must be used.

14.1.1 Specifying the TEST runtime option

Under UNIX System Services, the LE runtime options are specified using the _CEE_RUNOPTS environment variable. This variable can be set using an export command in the shell, for example:

```
export _CEE_RUNOPTS='TEST(,,,VADTCPIP&9.30.62.149%8000:*)'
```

This runtime option will now apply to every job that runs until you unset the environment variable using the command:

```
unset _CEE_RUNOPTS
```

Be very careful about what you run in the shell when the TEST runtime option is specified in _CEE_RUNOPTS. Commands such as **tar**, **c89**, and **make** can cause the tar, c89, and make utilities to appear in the remote debugger screen (in an assembler view since no debug information is available). To check what Language Environment runtime options you currently have set, you can use the command:

```
echo $_CEE_RUNOPTS
```

Figure 14-1 shows:

- Setting the TEST Language Environment runtime option.
- Querying which runtime options are currently specified.
- Running a program and bringing up the remote debugger (not shown).
- Unsetting the Language Environment runtime options.
- Querying which runtime options are currently specified.

```
(C) Copyright Software Development Group, University of Waterloo, 1989.

All Rights Reserved.

U.S. Government users - RESTRICTED RIGHTS - Use, Duplication, or
Disclosure restricted by GSA-ADP schedule contract with IBM Corp.

IBM is a registered trademark of the IBM Corp.

$ export _CEE_RUNOPTS='TEST(,,,VADTCPIP&9.30.62.149%8000:*)'
$ echo $_CEE_RUNOPTS
TEST(,,,VADTCPIP&9.30.62.149%8000:*)
$ a.out
hello world!
$ echo $_CEE_RUNOPTS
TEST(,,,VADTCPIP&9.30.62.149%8000:*)
$ unset _CEE_RUNOPTS
$ echo $_CEE_RUNOPTS

$
 ===>
                                                                    INPUT
ESC=¢   1=Help      2=SubCmd    3=HlpRetrn  4=Top       5=Bottom    6=TSO
        7=BackScr   8=Scroll    9=NextSess 10=Refresh  11=FwdRetr  12=Retrieve
```

Figure 14-1 Setting and clearing Language Environment runtime options in the USS shell

Note: When running under USS, the test option can always be specified in _CEE_RUNOPTS regardless of the language used to compile the program entry point. No slashes (/) are required.

Other methods of specifying runtime options

CEEUOPTS, and the C #pragma runopts directive can also be used under the shell to invoke the debugger. However, the _CEE_RUNOPTS environment variable tends to be the preferred method because TCP/IP addresses and VTAM terminal LU IDs can change, and the environment variable allows the person debugging the program to change their TEST runtime option without having to re-compile or relink their program.

14.1.2 Special considerations for running DLL applications from the shell

When running a USS module and looking for DLLs to load, the loader looks in the path specified by the LIBPATH environment variable. This should be set to point to your DLLs prior to running your DLL application.

14.2 Using C/C++ and Debug Tool under USS

In this example we rebuild the DLL application used in the C/C++ example. In this case our source is called:

- main.c
- setup.c
- process.c
- report.c
- cleanup.c

Each file will be compiled into its own DLL and its export file will be saved.

14.2.1 Compiling a C/C++ application

The following commands were used to build the DLL application. Notice the `-g` to ask for debug information and the `LANGLVL(EXTENDED)` to allow the use of the ctest function.

```
c89 -g -Wc,EXPORTALL -Wl,DLL -o setupd  setup.c
c89 -g -Wc,EXPORTALL,DLL,LANGLVL\(extended\) -Wl,DLL -o processd process.c setupd.x
c89 -g -Wc,EXPORTALL,DLL -Wl,DLL -o reportd  report.c setupd.x
c89 -g -Wc,EXPORTALL,DLL -Wl,DLL -o cleand cleanup.c setupd.x
c89 -g -Wc,DLL  -o linecount main.c setupd.x processd.x reportd.x cleand.x
```

14.2.2 Debugging a USS application

Next we give some examples of debugging a USS application.

Setup

We are running the program in a different directory than we built it in, so we need to ensure that the PATH and LIBPATH are set up correctly. Our main and DLLs are in /u/redbk1/dllexmp/:

```
$ echo $PATH
/usr/lpp/java/IBM/J1.3/bin:/usr/lpp/Printsrv/bin:/bin:.
$ echo $LIBPATH
/lib:/usr/lib:/usr/lpp/Printsrv/lib:.
```

The current directory is in both paths, but our dllexmp directory is not in either of them. We must modify the PATH and LIBPATH as follows:

```
$ export PATH=/u/redbk1/dllexmp:$PATH
$ export LIBPATH=/u/redbk1/dllexmp:$LIBPATH
$ echo $PATH
/u/redbk1/dllexmp:/usr/lpp/java/IBM/J1.3/bin:/usr/lpp/Printsrv/bin:/bin:.
$ echo $LIBPATH
/u/redbk1/dllexmp:/lib:/usr/lib:/usr/lpp/Printsrv/lib:.
```

Now we can do a test run of the linecount program:

```
$ linecount bigtestfile.txt
Report on File bigtestfile.txt
Total number of records: 127169
```

Invocation

First we set up the TEST runtime option. From the C/C++ example, we see the appropriate test option is:

```
NOTEST(ALL,*,NOPROMPT,VADTCPIP&9.30.62.149%8000:*)
```

We export our runtime option as follows:

```
export _CEE_RUNOPTS='NOTEST(ALL,*,NOPROMPT,VADTCPIP&9.30.62.149%8000:*)'
```

Now we start the remote debugger and run the program:

```
$ linecount bigtestfile.txt
```

As before, the remote debugger is triggered right at the ctest() call in our program. However, because the name of the source files is placed in the module as a relative name, the debugger cannot find the source file and asks where it can be found, as shown in Figure 14-2.

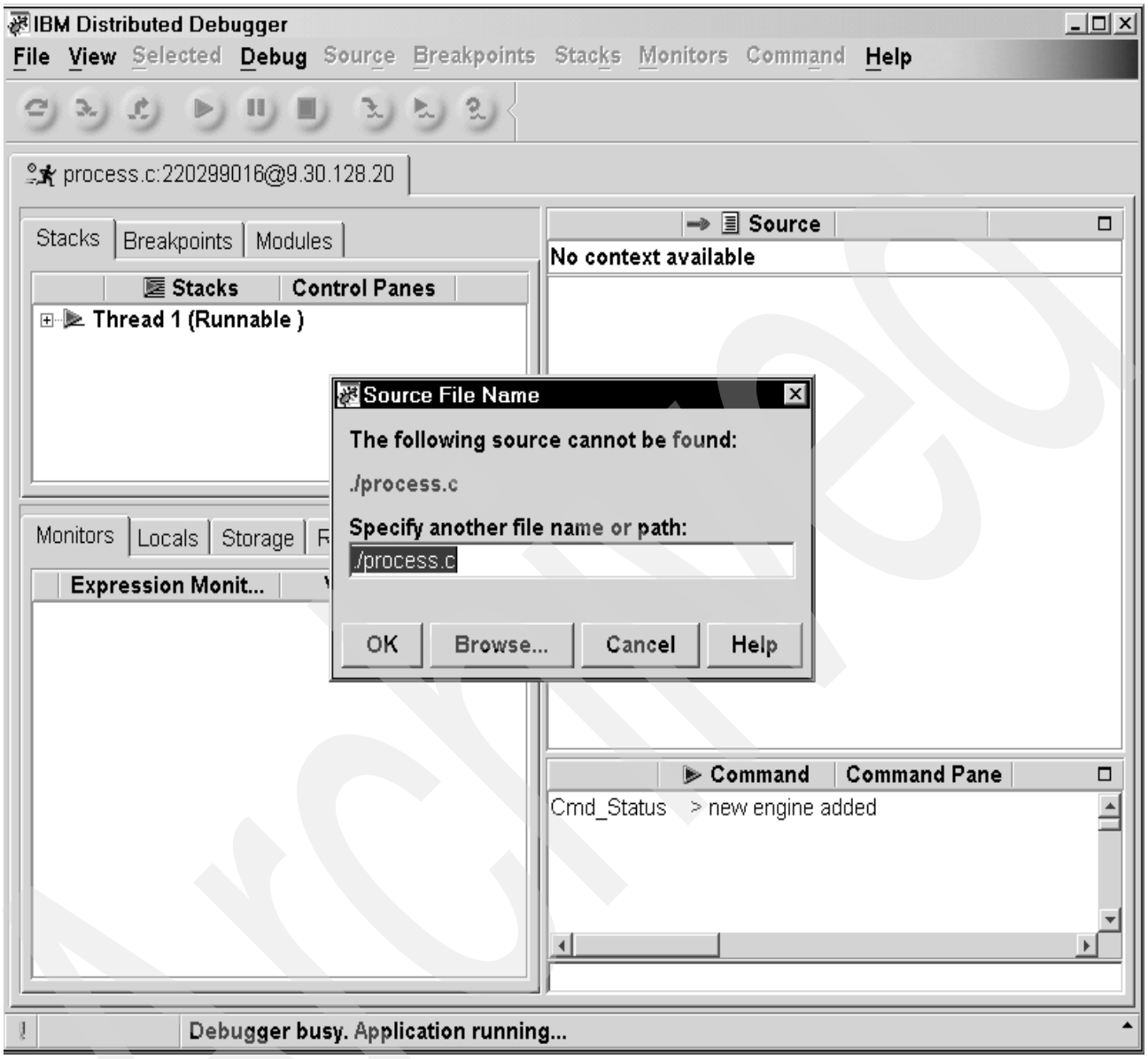


Figure 14-2 Debugger asking for source file location on invocation

We then fill in the correct absolute address as shown in Figure 14-3.

Figure 14-3 Providing the correct absolute address

The debugger now loads the correct source file as shown in Figure 14-4.

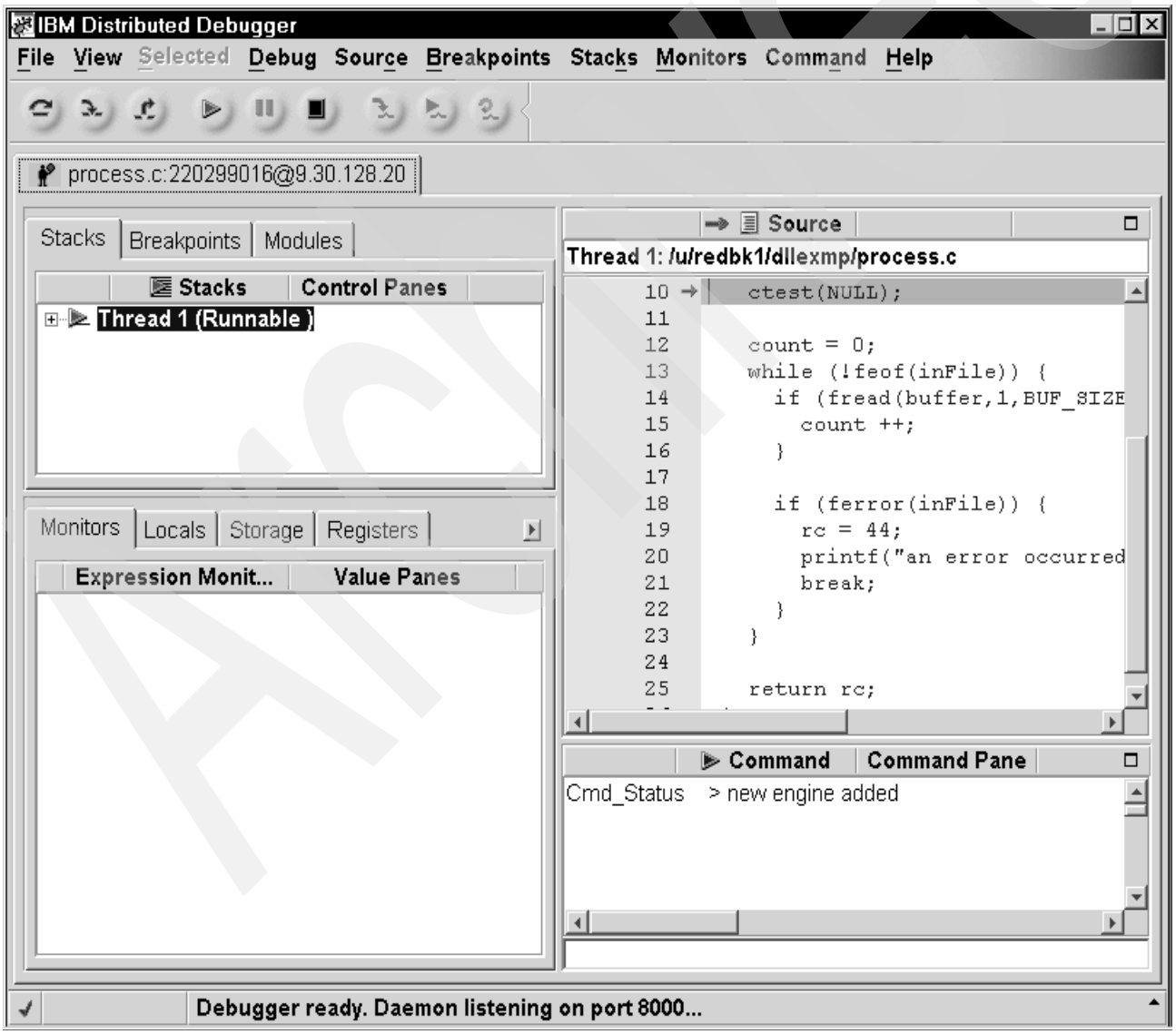


Figure 14-4 The debug screen once it has found its source

Alternatively, if we had run the program from the directory where we compiled it, we could just refer to the test file:

```
linecount ../test/bigtestfile.txt
```

In this case the debugger would not have needed to prompt for the source file location as shown in Figure 14-5.

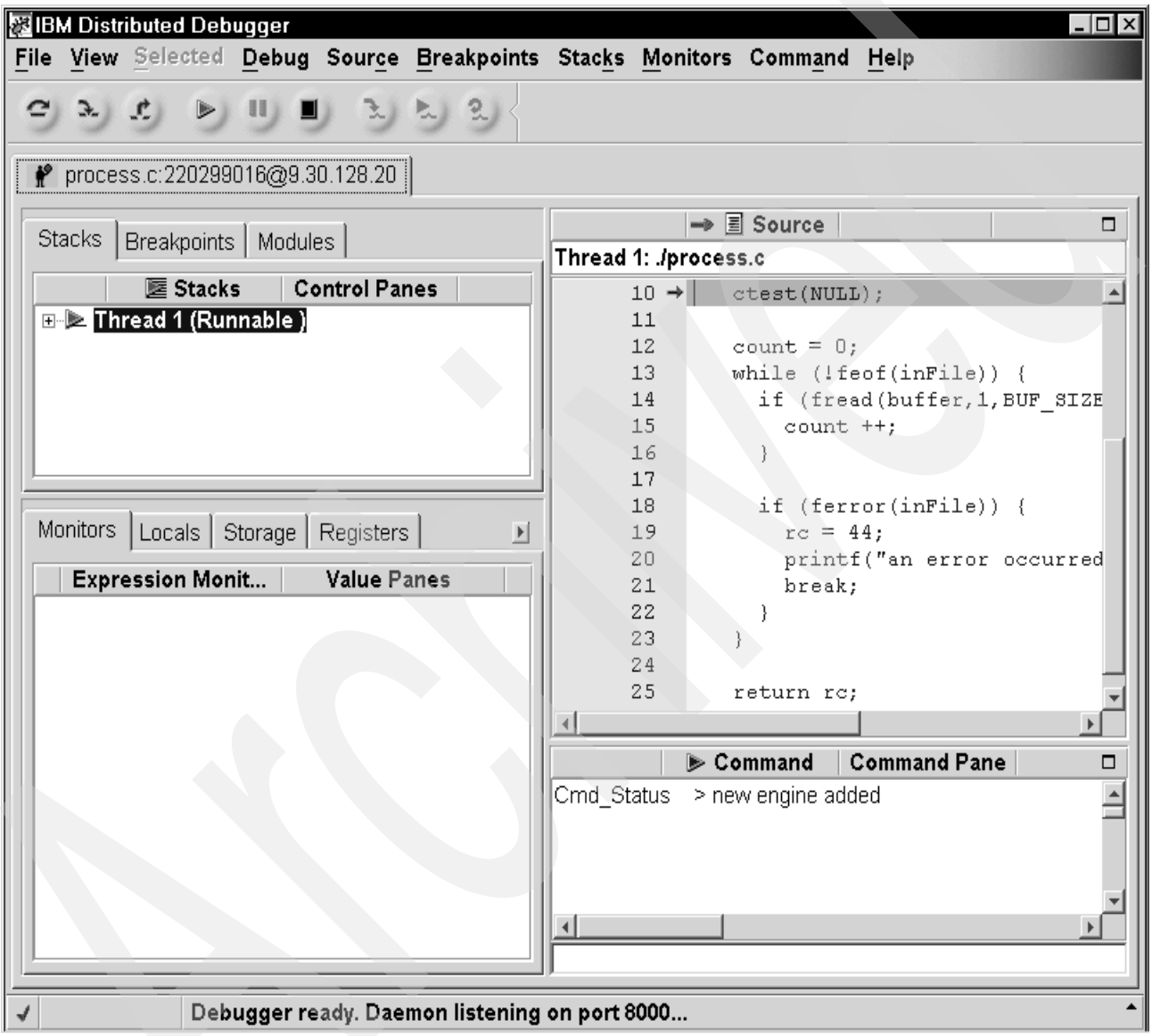


Figure 14-5 When run from the directory it was compiled in, the source comes up immediately

Note: We were unable to test the **pli** and **cob2** commands under USS. We do not know if they store full paths or relative paths to their data files.

15

Debug Tool and subsystems

This chapter describes how Debug Tool can help you debug your application while it runs in a host environment, such as CICS, IBM IMS, or IBM DB2 Universal Database™ (including IBM DB2 stored procedures) environments.

15.1 Debug Tool and CICS

This topic is divided into four parts:

- How to add support for debugging under CICS.
- How to prepare a CICS program, it includes:
- How to start Debug Tool under CICS, it includes:
- How to debug CICS programs, it includes:

15.1.1 How to add support for debugging under CICS

To debug applications that run in CICS, the system administrator must perform several steps. Some of them are mandatory, the others are only required in certain circumstances.

Basic CICS customization

The steps for basic CICS customization are as follows:

1. CICS definitions

   Several Debug Tool resources have to be defined and installed into the CICS region. The CICS definitions are in EQACCSD and EQACDCT members of the hlq.SEQASAMP data set.

   EQACDCT contains six Destination Control Table macro definitions as part of the Debug Tool under CICS. If your policy is to define CICS resources using CSD, you do not have to use these definitions.

   EQACCSD contains the CICS definitions that are part of the Debug Tool under CICS; you have to remove comments around the DEFINE TDQUEUE statements.

   > **Note:** To use a COBOL side file, the Transient Data Queue CIGZ is needed. This queue might also exist in Language Environment CICS resource definitions. In that case be sure which one will be used.

   The group (EQA) that contains the Debug Tool run time routines, as well as transactions and TD queues, must be in the group list used during CICS start-up.

   > **Note:** If your policy is to autoinstall program definitions, you have to add comments around the DEFINE PROGRAMS statements.

2. JCL updates

   Include Debug Tool's hlq.SEQAMOD data set in the DFHRPL concatenation.

   Include, if it is not already the case, the Language Environment runtime libraries (SCEECICS, SCEERUN and if it is required by your applications SCEERUN2) in the DFHRPL concatenation.

   Include EQA00DYN and EQA00HFS from the Debug Tool's hlq.SEQAMOD data set in the STEPLIB concatenation in one of the following ways:

   – Use the Authorized Program Facility to authorize the hlq.SEQAMOD data set and add it to the STEPLIB concatenation.

   – Copy the EQA00DYN and EQA00HFS modules from hlq.SEQAMOD data set to a library that is already to the STEPLIB concatenation.

   Ensure that no DD statements exist for CINSPIN, CINSPLS, CINSPOT, or IGZDBGIN.

3. For any terminal that Debug Tool uses to display a debugging session, verify that the CICS TYPETERM definition for that terminal specifies a minimum value of 4096 for the RECEIVESIZE parameter or sets the BUILDCHAIN parameter to YES.
4. Verify that users are able to run the CDT# transaction without receiving any errors.
5. If you are running your CICS programs in a distributed environment, the DTCN transaction name must be defined the same across all local and remote systems. If the DTCN transaction name is changed, or if a DTCN transaction is duplicated and given a different name, the name must be changed on all systems.
6. If you are planning to debug command-level assembler application programs that do not run under or use Language Environment services, you need to activate the CICS Non-Language Environment Exits as described in the next section.

Activating CICS Non-Language Environment exits

To debug non-Language Environment assembler programs that run under CICS, you must start the required Debug Tool global user exits before you start the programs. Debug Tool provides the following global user exits to help you debug non-Language Environment applications: XPCFTCH, XEIIN, XEIOUT, XPCTA, and XPCHAIR.

DTCX transaction

You can turn the exits on and off by using the transaction DTCX.

To activate all of the exits, from a clear CICS terminal screen enter DTCXXO. You need to activate the exits only once. After you enter DTCXXO, if all exits are activated successfully, the following message is displayed:

```
EQA9970I - CICS exit activation successful.
```

To deactivate all of the exits, enter DTCXXF. When you enter DTCXXF, the following message is displayed:

```
EQA9971I - CICS exit deactivation successful.
```

If there is a problem starting or activating one of the exits, an error message like the following is displayed:

```
EQA9977E - OOSVC is backlevel. Exits not enabled.
EQA9974I Error enabling XPCFTCH - EQANCFTC
```

If you see this error message, verify that the CICS CSD is properly updated to include the latest Debug Tool resource definitions, and that the Debug Tool SEQAMOD data is in the DFHRPL DD concatenation for the CICS region. You can start the exits during region initialization by using a sequential terminal. You are not required to shut down the exits prior to or during a region shutdown.

PLT solution

The exits can also be activated automatically during CICS region startup by using the Debug Tool-supplied CICS Program List Table (PLT) post-initialization program EQANCPLT. This program must run in the second or third phase of the CICS PLT process. See the sample provided in the hlq.SEQASAMP member EQACXITP for more details.

Sharing the DTCN repository profile items among CICS systems

The DTCN debug profile repository is a CICS temporary storage queue called EQADTCN2. If you want to share the repository among CICS systems, you can use either of the following methods.

1. Designate a single CICS region as the Queue-Owning Region (QOR) and define the queue as REMOTE in a TSMODEL resource definition on regions that need to access it remotely. This makes the queue profile items owned by one CICS system accessible to other CICS systems as shown in Figure 15-1.

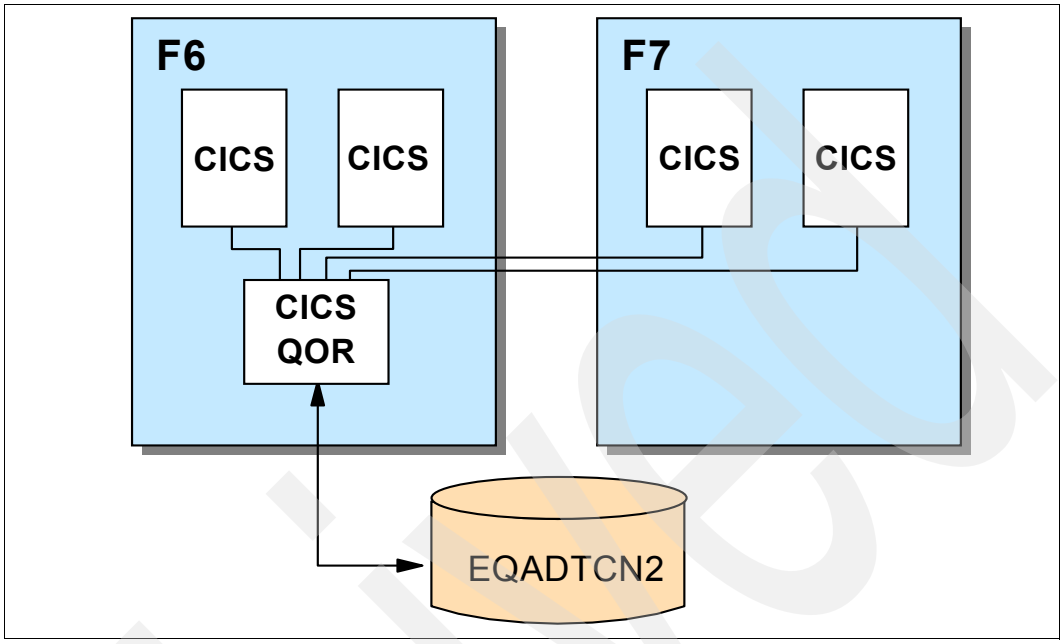


Figure 15-1 Sharing DTCN profile using remote TSQ

Example 15-1 is a sample resource definition that shows how to define the Debug Tool EQADTCN2 temporary storage queue in a region that will use it remotely.

Note: To optimize the performance of Debug Tool, it is important that you define this queue as Location MAIN.

Example 15-1 TSmodel remote

```
CEDA View TSmodel(DTCN1)
   TSmodel       ==> DTCN1
   Group         ==> DTCNREM
   Description   ==> TEST DTCN TSQ REMOTE
   PRefix        ==> EQADTCN2
   XPrefix       ==>
   Location      ==> Main          Auxiliary | Main
RECOVERY ATTRIBUTES
   RECovery      ==> No            No | Yes
SECURITY ATTRIBUTES
   Security      ==> No            No | Yes
SHARED ATTRIBUTES
   POolname      ==>
REMOTE ATTRIBUTES
   REMOTESystem ==> C31F
   REMOTEPrefix ==> EQADTCN2
   XRemotepfx   ==>
```

2. If you have a Parallel Sysplex® environment you can update the CFRM (Coupling Facility Resource Manager) policy to define a new structure named DFHXQLS_*poolname* and start a CICS TS data sharing server to share TS queues across your CICSplex as shown Figure 15-2.

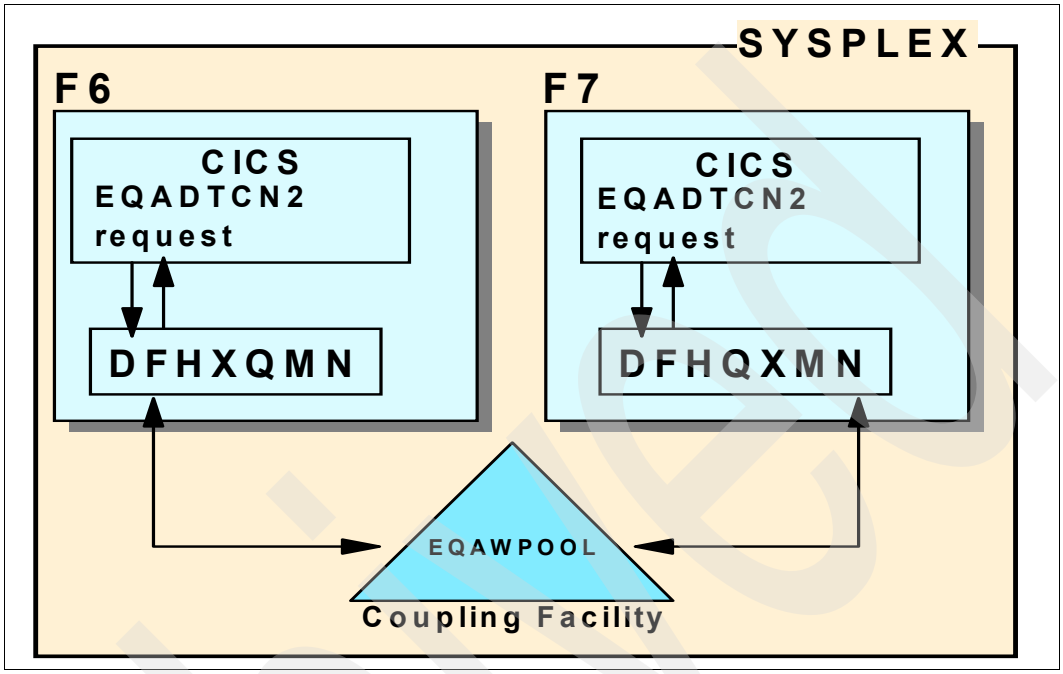


Figure 15-2 Sharing DTCN profile using shared TSQ

Example 15-2 is a sample of how to define the DFHXQLS_*poolname* structure.

Example 15-2 DFHXQLS_poolname definition

```
STRUCTURE
   NAME (DFHXQLS_EQAWPOOL)
   SIZE(50000)
   INITSIZE(20000)
   PREFLIST(FACIL01,FACIL02)
```

Example 15-3 shows how to write the job to manage the DFHQXLS_*poolname*.

Example 15-3 TS data sharing server JCL

```
//CTSTSHF6 JOB 1,RACFUSER,TIME=1440,NOTIFY=&SYSUID,REGION=4M,
//            CLASS=A,MSGCLASS=X,MSGLEVEL=(1,1)
//*    Start CICS TS data sharing server
//TSSERVER EXEC PGM=DFHXQMN,REGION=64M
//STEPLIB DD DSN=CICSVS.TS310.CICS.SDFHAUTH,DISP=SHR
//*    Authorized library containing DFHXQMN
//SYSPRINT DD SYSOUT=*
//*    Options
//SYSIN DD    *
FUNCTION=SERVER
POOLNAME=EQAWPOOL  Pool name
MAXQUEUES=1000     Allow up to 1003 List Structures
```

Example 15-4 shows how to define the Debug Tool EQADTCN2 temporary storage queue in a region that will use it shared.

Example 15-4 TSmodel shared

```
CEDA View TSmodel(DTCN1)
   TSmodel      ==> DTCN1
   Group        ==> DTCNREM
   Description  ==> TEST DTCN TSQ REMOTE
   PRefix       ==> EQADTCN2
   XPrefix      ==>
   Location     ==> Main          Auxiliary | Main
RECOVERY ATTRIBUTES
   RECovery     ==> No            No | Yes
SECURITY ATTRIBUTES
   Security     ==> No            No | Yes
SHARED ATTRIBUTES
   POolname     ==> EQAWPOOL
REMOTE ATTRIBUTES
   REMOTESystem ==>
   REMOTEPrefix ==>
   XRemotepfx   ==>
```

These settings store a profile item in one CICS system, or one TS server region, and make it readable to other CICS systems.

Enabling communication between Debug Tool and a remote debugger

If you use a remote debugger, Debug Tool communicates with the remote debugger by using TCP/IP Sockets.

If you have CICS Transaction Server for z/OS Version 2 Release 3 or later, Debug Tool uses the CICS Sockets Domain. To start the Sockets Domain, ensure that the CICS system initialization parameter TCPIP is set to YES. You also have to define a TCPIPSERVICE resource. Example 15-5 shows the sample resource definition and how to define it.

Example 15-5 TCpiservervice definition for the Remote Debugger

```
CEDA  ALter TCpipservice( REMDEBUG )
 TCpipservice   : REMDEBUG
 GROup          : DEBUGTOL
 DEscription  ==> REMOTE DEBUG TOOL TCPIP PORT
 Urm          ==> DFHWBADX
 POrtnumber   ==> 08001              1-65535
 STatus       ==> Open               Open ! Closed
 PROtocol     ==> Http               Iiop ! Http ! Eci ! User
 TRansaction  ==> CWXN
 Backlog      ==> 00005              0-32767
 TSqprefix    ==>
 Ipaddress    ==>
 SOcketclose  ==> No                 No ! 0-240000 (HHMMSS)
 Maxdatalen   ==> 006000             3-536870
SECURITY
 SSl          ==> No                 Yes ! No ! Clientauth
 CErtificate  ==>
 (Mixed Case)
 PRIvacy      ==>                    Notsupported ! Required ! Supported
```

```
    CIphers      ==>
    AUthenticate ==> No                 No ! Basic ! Certificate ! AUTORegister
                                        ! AUTOMatic ! ASserted
    ATtachsec    ==>                    Local ! Verify
   DNS CONNECTION BALANCING
    DNsgroup     ==>
    GRPcritical  ==> No                 No ! Yes
```

If you have an earlier version of CICS, Debug Tool uses the TCP/IP Socket Interface for CICS feature of TCP/IP for MVS.

You can use EZAC, a TCP/IP-supplied CICS transaction, to inquire about the status of TCP/IP connections in the CICS region.

EZAC,DISPLAY,LISTENER provides information about the TCP/IP listener task that needs to be running.

EZAC,DISPLAY,CICS displays information about the CICS-TCP/IP configuration.

Enabling the CADP transaction

If you have CICS Transaction Server for z/OS Version 2 Release 3, and you want to use the CADP transaction to manage debugging profiles, set the DEBUGTOOL system initialization parameter to YES.

The default setting of DEBUGTOOL=NO results in the DTCN transaction being used for Debug Tool profiles.

You can still use CADP to define profiles with DEBUGTOOL=NO, but they will not be used by Debug Tool.

You can dynamically switch between the CADP and DTCN debug profiles that are used by Debug Tool. Use the following commands to switch between the profiles after the CICS region is started:

`CEMT SET DEBUG` - use CADP profiles

`CEMT SET NODEBUG` - use DTCN profiles

Enabling the CADP Web interface

If you have CICS Transaction Server for z/OS Version 2 Release 3 or later, Debug Tool uses the CICS Sockets Domain. To start the Sockets Domain, ensure that the CICS system initialization parameter TCPIP is set to YES. You also have to define a TCPIPSERVICE resource. Example 15-6 shows how to define it.

Example 15-6 TCpipservice definition for the CADP Web interface

```
OBJECT CHARACTERISTICS                                      CICS RELEASE = 0630
 CEDA  View TCpipservice( DEBUGWEB )
  TCpipservice   : DEBUGWEB
  GROup          : DEBUG
  DEscription    : TCPIP SERVICE FOR CADP Web interface
  Urm            : DFHWBADX
  POrtnumber     : 05000              1-65535
  STatus         : Open               Open | Closed
  PROtocol       : Http               Iiop | Http | Eci
  TRansaction    : CWXN
  Backlog        : 00005              0-32767
```

```
 TSqprefix      :
 Ipaddress      :
 SOcketclose    : No                 No | 0-240000 (HHMMSS)
SECURITY
 SSl            : No                 Yes | No | Clientauth
 Certificate    :
 (Mixed Case)
 PRIvacy        :                    Notsupported | Required | Supported
 AUthenticate   : No                 No | Basic | Certificate | AUTORegister
                                     | AUTOMatic | ASserted
 ATtachsec      :                    Local | Verify
DNS CONNECTION BALANCING
 DNsgroup       :
 GRPcritical    : No                 No | Yes

                                                 SYSID=ADT2 APPLID=A6T1ADT2
```

Start the application debugging profile manager by typing its URL in your Web browser. The URL that you enter will depend upon how CICS Web support is configured.

For example, if your Web browser connects directly to CICS, and your system is configured to use the default analyzer program (DFHWBADX), the URL is:

```
http://mvs_address:5000/CICS/CWBA/dfhdpwb
```

Your system administrator will tell you the URL to use for your system.

Sharing the CADP repository profile items among CICS systems

The CADP debug profile repository is VSAM files called DFHDPFMB and DFHDPFMP. If you want to share the repository among CICS systems, use either of the following methods.

1. Designate a single CICS region as the File-Owning Region (FOR) and define the file as REMOTE in a FILE resource definition on regions that need to access it remotely. This makes the file profile items owned by one CICS system accessible to other CICS systems as shown in Figure 15-3.

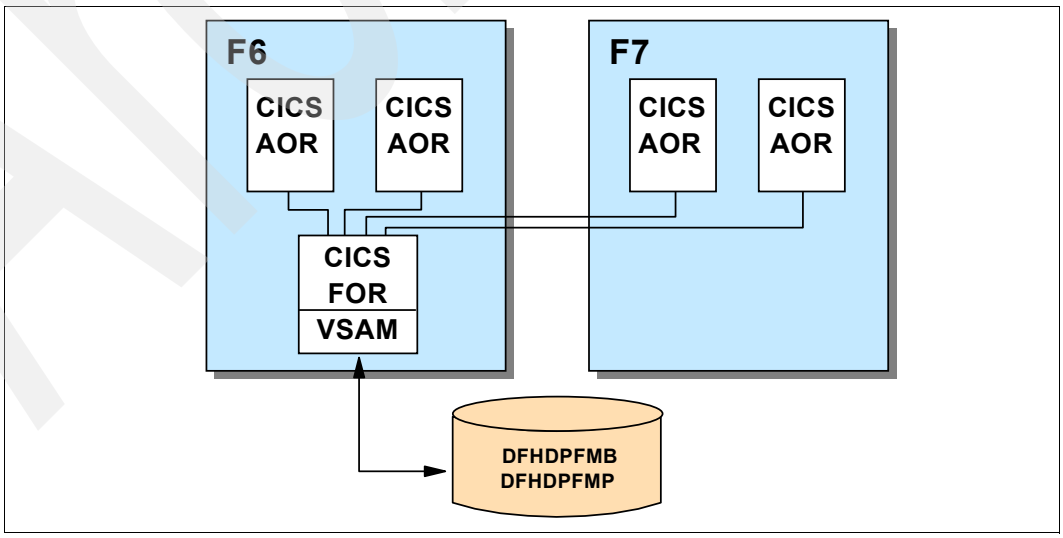


Figure 15-3 Remote file

Example 15-7 is a sample resource definition that shows how to define the Debug Tool DFHDPFMB and DFHDPFMP files in a region that will use them remotely.

Example 15-7 DFHDPFMB - Remote definition

```
CEDA  ALter File( DFHDPFMB )
 File           : DFHDPFMB
 Group          : ERCEQAW
 DEScription  ==>
VSAM PARAMETERS
 DSNAme       ==>
 Password     ==>                    PASSWORD NOT SPECIFIED
 RLsaccess    ==> No                 Yes ! No
 LSrpoolid    ==> 1                  1-8 ! None
 READInteg    ==> Uncommitted        Uncommitted ! Consistent ! Repeatable
 DSNSharing   ==> Allreqs            Allreqs ! Modifyreqs
 STRings      ==> 001                1-255
 Nsrgroup     ==>
REMOTE ATTRIBUTES
 REMOTESystem ==> C31F
 REMOTEName   ==> DFHDPFMB
REMOTE AND CFDATATABLE PARAMETERS
 RECORDSize   ==>                    1-32767

                                                SYSID=C31G APPLID=CICSC31G
```

The same definition has to be created for DFHDPFMP.

2. The two files can be shared across a sysplex using a coupling facility if the VSAM Record Level Sharing is available on your sysplex. What you need is the following:
   - VSAM server - SMSVSAM
   - SHaring Control Data Set (SHCDS)
   - Common buffers and control blocks management:
     - SMSVSAM data space
     - Coupling facility data caches structure
     - Coupling facility lock structure named IGWLOCK00

   Figure 15-4 illustrates the basic structure of this configuration.

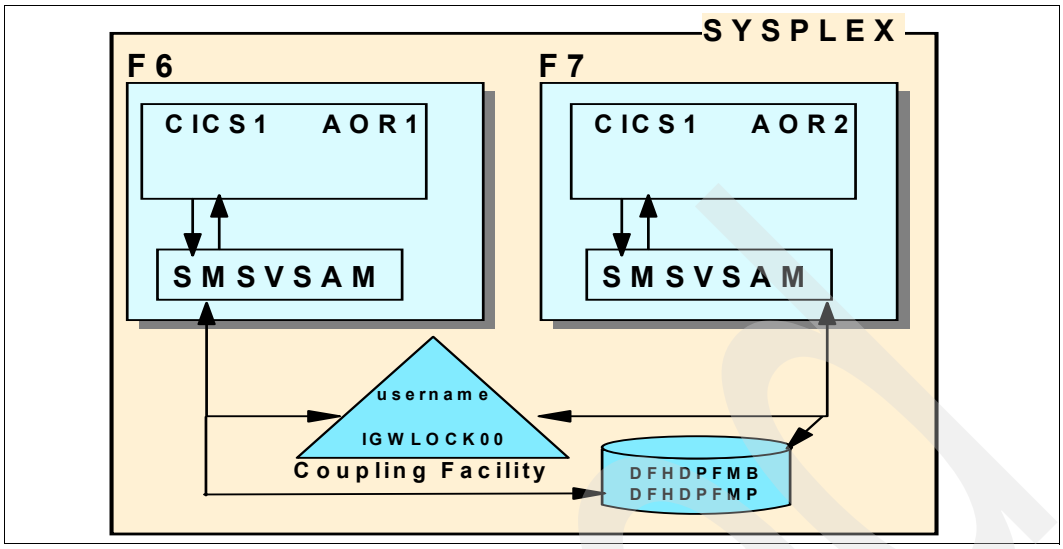


Figure 15-4 VSAM RLS configuration

15.1.2 How to prepare a CICS program

To prepare a CICS program for debugging, you must do the following tasks:

1. Complete the program preparation tasks for assembler, C, C++, COBOL, or PL/I. The detailed steps to do this are covered in Chapter 13, “Debugging programs using Debug Tool” on page 193.
2. If you are using debugging profiles to indicate which programs to debug, choose one of the following CICS transactions to manage your debugging profiles:
   – If you are using versions of CICS earlier than CICS Transaction Server for z/OS Version 2 Release 3, use DTCN to manage debugging profiles.

     Link-edit EQADCCXT into the CICS program that you want to debug. This step is required only for PL/I and C/C++ programs.
   – If you are using CICS Transaction Server for z/OS Version 2 Release 3 or later, you can use either CADP or DTCN to manage debugging profiles.

     If you use CADP, you do not need to link-edit EQADCCXT into the CICS program that you want to debug. You cannot use both CADP and DTCN, you must choose only one.
3. Create and store a profile that specifies the combination of resource IDs that you want to debug.
4. Run your program.

Do you need to link edit EQADCCXT into your program?

Debug Tool provides a Language Environment CEEBXITA assembler exit called EQADCCXT to help you activate, by using the DTCN transaction, a debugging session under CICS.

You do not need to use this exit if you are running either of the following options:

- You are running under CICS Transaction Server for z/OS Version 2 Release 3 or later and you use the CADP transaction to define debug profiles.
- You are using the DTCN transaction and you are debugging COBOL programs.

If you activate a debug session by using DTCN and you are debugging a C, C++, or PL/I programs, you need to link the EQADCCXT exit into the main load module of the program. When you use EQADCCXT, be aware of the following conditions:

- If your site does not use a Language Environment assembler exit (CEEBXITA), then link-edit member EQADCCXT, which contains the CSECT CEEBXITA and is in library hlq.SEQAMOD, into your main program.
- If your site uses an existing CEEBXITA, the EQADCCXT exit provided by Debug Tool must be merged with it. The source for EQADCCXT is in hlq.SEQASAMP(EQADCCXT). Link the merged exit into your main program.

After you link-edit your program, use the DTCN transaction to create a profile that specifies the combination of resource IDs that you want to debug.

How to create and store a DTCN profile

The DTCN transaction stores one profile for each DTCN terminal in a repository. Each profile is retained in the repository until one of the following events occurs:

- The profile is explicitly deleted by the terminal that entered it.
- DTCN detects that the terminal which created the profile has been disconnected.
- The CICS region is terminated, except if you share EQADTCN2 through the coupling facility.

To create and store a DTCN profile, perform the following steps:

1. Log on to a CICS terminal and enter the transaction ID **DTCN**. The DTCN transaction displays the main DTCN screen, shown in Figure 15-5.

```
DTCN                Debug Tool CICS Control - Primary Menu            CICSC22F

Select the combination of resources to debug (see Help for more information)

 Terminal Id      ==> 0007
 Transaction Id   ==>
 Program Id(s)    ==>          ==>          ==>          ==>
                  ==>          ==>          ==>          ==>
 User Id          ==> CHABERT
 NetName          ==>

Select type and ID of debug display device

 Session Type      ==> MFI                 MFI, TCP
 Port Number       ==>                     TCP Port
 Display Id        ==> 0007
 Connection Type   ==> SINGLE              Single, Multiple

Generated String:

Repository String: No string currently saved in repository


PF1=HELP 2=GHELP 3=EXIT 4=SAVE 6=DELETE 7=SHOW 9=OPTIONS
                                                                     05/023
```

Figure 15-5 DTCN - Primary Menu

2. Specify the combination of resource IDs that you want to debug.

Terminal Id Specify the CICS terminal to debug.
By default, this ID is set to the terminal that is currently running DTCN.

Transaction Id Specify the CICS transaction to debug.
If you specify a transaction ID without any other resource, Debug Tool

is started every time that transaction is run, including times when other users run the transaction.

Program Id(s) Specify the CICS program or programs to debug.
If you specify a program ID without any other resource, Debug Tool is started every time the program is run, including times when other users run the program.

Specifying a CICS program in the Program Id(s) field is similar to setting a breakpoint by using the AT ENTRY command.

If Debug Tool is already running and it cannot find the separate debug file, then Debug Tool does not stop at the CICS program specified in the Program Id(s) field. Use the AT APPEARANCE or AT ENTRY command to stop at this CICS program.

User Id Specify the CICS user ID to debug.
All programs that are run by this user will start Debug Tool.

NetName Specify the NETNAME of a CICS terminal.
All programs that are running by this Netname will start Debug Tool.

3. Specify the type of debugging and the ID of the display device.

Session Type Select one of the following options:

- **MFI** - Indicates that Debug Tool initializes on a 3270 type of terminal.
- **TCP** - Indicates that you want to interact with Debug Tool from your workstation using TCP/IP and a remote debugger.

Port Number Specifies the TCP/IP port number that is listening for debug sessions on your workstation. By default, IBM Distributed Debugger uses port 8000; the Compiled Language Debugger component of WebSphere Studio Enterprise Developer uses port 8001.

Display Id Identifies the target destination for Debug Tool information. Depending on the session type that you've selected, the display ID is one of the following:

- If you selected MFI, the display ID is a CICS 3270 terminal ID. This ID is set by default to the terminal ID that is currently running DTCN, but you can change this to direct MFI screens to a different CICS terminal.
- If you selected TCP, enter either the IP address or host name of the workstation that will display the debug screens. For the debug session to start, the appropriate software must be running on that workstation.

Connection Type Specifies whether you are using a single or multiple socket connection to the workstation.

> **Note:** If you are using a version of the Distributed Debugger that is version 9.2 or higher, or the Compiled Language Debugger component of WebSphere Studio Enterprise Developer, we recommend you specify a single socket connection.
>
> If you are using a version of the Distributed Debugger that is earlier than version 9.2 (copyright date 2003/10/19), we recommend you specify a multiple socket connection.

4. Specify the debugging options by pressing PF9 to display the secondary options menu, shown in Figure 15-6.

```
DTCN                Debug Tool CICS Control - Menu 2                    CICSC22F

Select Debug Tool options

Test Option     ==> TEST                Test/Notest
Test Level      ==> ALL                 All/Error/None
Commands File   ==> *
Prompt Level    ==> PROMPT
Preference File ==> *

Any other valid Language Environment options
==>

PF1=HELP 2=GHELP 3=RETURN
                                                                      05/022
```

Figure 15-6 DTCN - Menu 2

Test Option TEST/NOTEST specifies the conditions under which Debug Tool assumes control during the initialization of your application.

Test Level ALL/ERROR/NONE specifies what conditions need to be met for Debug Tool to gain control.

Command File A valid fully qualified data set name that specifies the primary commands file for this run.

Do not enclose the name of the data set in single or double quotes.

Prompt Level Specifies whether Debug Tool is started at Language Environment initialization.

Preference File A valid fully qualified data set name that specifies the preference file to be used.

Do not enclose the name of the data set in single or double quotes.

Any other valid Language Environment Options

You can change any Language Environment option that your site has defined as over-rideable except the STACK option.

5. Press PF3 to return to the main DTCN panel.
6. Press PF4 to save the profile. DTCN performs data verification on the data that you entered in the DTCN panel. When DTCN discovers an error, it places the cursor in the erroneous field and displays a message. You can use context-sensitive help (PF1) to find what is wrong with the input.

Now, any tasks that run in the CICS system and match the resource IDs that you specified in the previous steps will start Debug Tool.

To display all of the active DTCN profiles in the CICS region, press PF7. The Debug Tool CICS Control - All Sessions screen is displayed, as shown in Figure 15-7.

```
DTCN                Debug Tool CICS Control - All Sessions            A6T1ADT2

Overtype "_" with a "D" to delete a profile.

  Owner  Term  Tran  User      Netname   Applid   Options String

_ C5UE   C5U9  ERC1  ________  ________  A6T1ADT2 TEST(ALL,'*',PROMPT,'MFI%C5UE
         Program(s)  ________  ________  ________  ________
                     ________  ________  ________  ________
_ C5UG   C5UG  ERC2  ________  ________  A6T1ADT2 TEST(ALL,,PROMPT,'VADTCPIP&9.
         Program(s)  ERC2PGM*  ________  ________  ________
                     ________  ________  ________  ________

PF1=HELP 2=GHELP 3=RETURN 7=BACK 8=FORWARD
                                                                    07/002
```

Figure 15-7 DTCN All Sessions

The column titles are defined as follows:

Owner The ID of the terminal that created the profile by using DTCN.

Term The value that was entered on the main DTCN screen in the Terminal Id field.

Tran The value that was entered on the main DTCN screen in the Transaction Id field.

User The value that was entered on the main DTCN screen in the User Id field.

Netname The value the entered on the main DTCN screen in the Netname field.

Applid The application identifier associated with this profile.

Options String The value of Repository String field on the main DTCN screen; it's created based on the values that the user enters in the other fields.

Program(s) The values that were entered on the main DTCN screen in the Program Ids field.

DTCN also reads the Language Environment NOTEST option supplied to the CICS region in CEECOPT or CEEROPT. You can supply suboptions, such as the name of a preference file, with the NOTEST option to supply additional defaults to DTCN.

How to create and store a CADP profile

First you have to start the CICS Application Debugging Profile (CADP) manager 3270 interface by running the CADP transaction. CICS displays the panel shown in Figure 15-8.

```
CADP      -    CICS Application Debugging Profile Manager   -     CICSC31F

List Debugging Profiles    (A=Activate,I=Inactivate,D=Delete,C=Copy)

  Owner    Profile  S Tran Program  Compile Unit Applid   Userid   Term Type
_ $EXAMPLE CORBA    I T*                         *        IORWERTH      Corb
_ $EXAMPLE EJB      I *                          *        *             EJB
_ $EXAMPLE JAVA     I TR*                        *        PENFOLD*      Java
_ $EXAMPLE LE1      I T*   P*       *            CICSREG1 PANDREWS TTT1 LE
_ $EXAMPLE LE2      I TR   *        SAMPCOMPUN + CICSREG2 DRBEARD* TTT2 LE
_ $EXAMPLE LE3      I TRN3 PROG3    *            CICSREG3 *        TTT2 LE

6   profile(s). All profiles shown
Enter=Process PF1=Help 2=Filter 3=Exit 4=View 5=Create LE 6=Create Java
                 9=Set display device 10=Edit 11=Sort
                                                                   06/002
```

Figure 15-8 CADP Main panel

The List Debugging Profiles screen

When you use the debugging profile manager for the first time, CICS displays all the profiles that you own. Subsequently, CICS displays the profiles that were selected when you last used it.

The list contains selected information from the debugging profiles. The columns on the screen are:

Owner The userid of the profile owner; that is, of the user who created the profile.

Profile The name of the profile.

S The status of the profile (**A** for Active, or **I** for Inactive).

The following columns display information specified when the profile is created:

Tran Displays the contents of the transaction field.

Program Displays the contents of the program field.

Compile Unit Displays the first ten characters of the Compile Unit field. If the Compile Unit name is longer, a + to the right of the name shows that only part of the name is displayed.

Applid Displays the contents of the Applid field.

Userid Displays the contents of the Userid field.

Term Displays the contents of the Terminal field.

Type Displays the type of program specified in the debugging profile:
Corb - CORBA object
EJB™ - Enterprise bean
Java - Java program
LE - Compiled language program

Not all the information in the debugging profile is displayed on this screen. To display the additional information, move the cursor to the line that contains the profile, and press PF4.

You can change the way CICS displays information on the List Debugging Profiles screen:

- Selecting which profiles are displayed. Use PF2 to cycle through the available options in turn. The options are:
  - Display all the profiles in the system. This is the setting the first time you use the debugging profile manager.
  - Display the profiles that you own.
  - Display all active profiles.
- Sorting the list. Use PF11 to cycle through the available options in turn. The options are:
  - Re-display the profiles in sequence of profile name. This is the sequence the first time you use the debugging profile manager.
  - Re-display the profiles in sequence of transaction ID.
  - Re-display the profiles in sequence of program name.
  - Re-display the profiles in sequence of owner.

In each case, CICS uses the EBCDIC sorting sequence.

Your choice of which profiles are displayed, and your chosen sequence, are preserved and used the next time you use the debugging profile manager.

Creating a new profile for a compiled language program

To create a new profile for a compiled language program, complete the following steps:

1. From the List Debugging Profiles screen press PF5. CICS displays the Create LE Debugging Profile screen as shown in Figure 15-9.

```
CADP      -    CICS Application Debugging Profile Manager   -     CICSC31F

Create LE Debugging Profile ==> _         for CHABERT


CICS Resources To Debug (use * to specify generic values e.g. *, A*, AB*, etc.)
Transaction      ==>                                  Applid   ==> CICSC31F
Program          ==>                                  Userid   ==> CHABERT
Compile Unit     ==>                                  Termid   ==> 0019
                                                      Netname  ==> TCP00019

Debug Tool Language Environment Options
Test Level       ==> All                                     (All,Error,None)
Command File     ==>
Prompt Level     ==> PROMPT
Preference File  ==>

Other Language Environment Options
==>
==>
==>
==>

Enter=Create PF1=Help 2=Save options as defaults 3=Exit 10=Replace 12=Return
                                                                        03/034
```

Figure 15-9 CADP New LE profile

The fields on the screen are:

Create LE Debugging Profile
Specify the name of the profile. If you change the name of an existing profile, CADP creates a new profile with the new name, and leaves the original profile unchanged.

Transaction
Specify a value in this field when you want to debug only those programs that run under the specified transaction. You can specify a generic value.

Program
Specify a value in this field when you want to debug only the specified program. You can specify a generic value.

Compile unit
Specify a value in this field when you want to debug only the specified compile unit. You can specify a generic value.

Applid
Specify a value in this field when you want to confine debugging to programs that run in the specified CICS region. You can specify a generic value.

Userid
Specify a value in this field when you want to confine debugging to programs that are being run by the specified user. You can specify a generic value.

Termid
Specify a value in this field when you want to confine debugging to programs that are being run by the specified terminal. You can specify a generic value.

Netname
Specify a value in this field when you want to confine debugging to programs that are being run by the specified netname. You can specify a generic value.

Test level
Specifies which conditions raised by your program will cause Debug Tool to gain control. You can enter the following values: All, Error, or None.

Command file
Specifies the primary commands file associated with the profile. You can specify the fully qualified name of a sequential dataset or a member of a partitioned data set.

Prompt level
Specifies whether an initial commands list is unconditionally executed during program initialization. Enter one of the following: PROMPT, NOPROMPT or command.

Preference file
Specifies the preference file that Debug Tool uses when debugging programs that match this profile. You can specify the fully qualified name of a sequential dataset or a member of a partitioned data set.

Other Language Environment options
Specifies Language Environment runtime options for programs that match this profile. When a program is selected for debugging because it matches the profile, the runtime options specified will override other runtime options that you may have in effect.

2. Press Enter. CICS checks that you have entered valid data.
3. Press PF12 to return to the List Debugging Profiles screen.

Specifying default values for Debug Tool and LE options

You can specify default values for the following Debug Tool options, and the Language Environment options. The saved values are used by default each time you create a debugging profile for a compiled language program. The Debug Tool options are:

- Test level
- Command file
- Prompt level
- Preference file

To save the default values, start with the Create LE Debugging Profile screen, and follow these steps:

1. Type the default values that you want to specify for the Debug Tool options and Language Environment options.
2. Press PF2. CICS saves the values that you have specified.

The values that you save will be used by default each time you create a new profile.

Creating a new profile for a Java program

To create a new profile for a Java progeam, complete the following steps:

1. From the List Debugging Profiles screen press PF6. CICS displays the Create Java Debugging Profile screen as shown in Figure 15-10 and Figure 15-11.

```
CADP      -    CICS Application Debugging Profile Manager   -     CICSC31F

Create Java Debugging Profile ==>  _         for CHABERT

CICS Resources To Debug (use * to specify generic values e.g. *, A*, AB*, etc.)
Transaction      ==>                          Applid       ==> CICSC31F
                                              Userid       ==> CHABERT


Debugging Options
JVM Profile      ==>

Java Resources To Debug
Type             ==> J       (J=Java Applications, E=Enterprise Beans, C=Corba)

Class (Java Applications or Corba)
==>
==>
==>
==>

Press PF8 to set Bean and Method

Enter=Create PF1=Help 2=Save options as defaults 3=Exit 8=Forward
10=Replace 12=Return
                                                                        03/037
```

Figure 15-10 CADP New Java profile (Part 1 of 2)

```
CADP      -    CICS Application Debugging Profile Manager   -     CICSC31F

Java Resources To Debug

Bean (Enterprise Beans only)
==> _
==>
==>
==>

Method (Enterprise Beans or Corba)
==>
==>
==>
==>









Enter=Create PF1=Help 3=Exit 7=Back 10=Replace 12=Return
                                                              06/006
```

Figure 15-11 CADP New Java profile (Part 2 of 2)

- The fields on the screen are:

  Create Java Debugging Profile
  Specify the name of the profile. If you change the name of an existing profile, CADP creates a new profile with the new name, and leaves the original profile unchanged.

  Transaction
  Specify a value in this field when you want to debug only those programs that run under the specified transaction. You can specify a generic value.

  Applid
  Specify a value in this field when you want to confine debugging to programs that run in the specified CICS region. You can specify a generic value.

  Userid
  Specify a value in this field when you want to confine debugging to programs that are being run by the specified user. You can specify a generic value.

  JVM™ Profile
  Specify the name of the JVM profile that is used for Java programs that match this profile. The profile should specify that the Java program is to run in debug mode.

- The following fields specify which Java resources should trigger the start of a debugging session when the profile is active:

  Type
  Specifies the type of Java resource that you want to debug:

  - **J** Enter this value when you want to debug a Java program.
  - **E** Enter this value when you want to debug an enterprise bean.
  - **C** Enter this value when you want to debug a stateless CORBA object.

  Class
  For Java programs and stateless CORBA objects only, specify a value in this field when you want to debug only the specified class. You can specify a generic value.

Bean
For enterprise beans only, specify a value in this field when you want to debug only the specified bean. You can specify a generic value.

Method
For enterprise beans and CORBA objects only, specify a value in this field when you want to debug only the specified method. When an inbound request initiated by a Java remote method invocation is received, the value specified is compared with the mangled name in the inbound request to determine if the profile matches the request. If it is possible that mangling can take place, do not specify a method name in the debugging profile, but specify a generic method instead.

2. Complete the fields that you need to specify your profile.
3. Press Enter. CICS checks that you have entered valid data.
   – If you have specified valid data, the profile is saved.
   – If your data contains an error, CICS displays a message. Re-enter the data, and press Enter again.
4. Press PF12 to return to the List Debugging Profiles screen.

How to create and store a profile using the CADP Web interface

When you start the application debugging profile manager's Web interface, CICS displays the List profiles page shown in Figure 15-12.

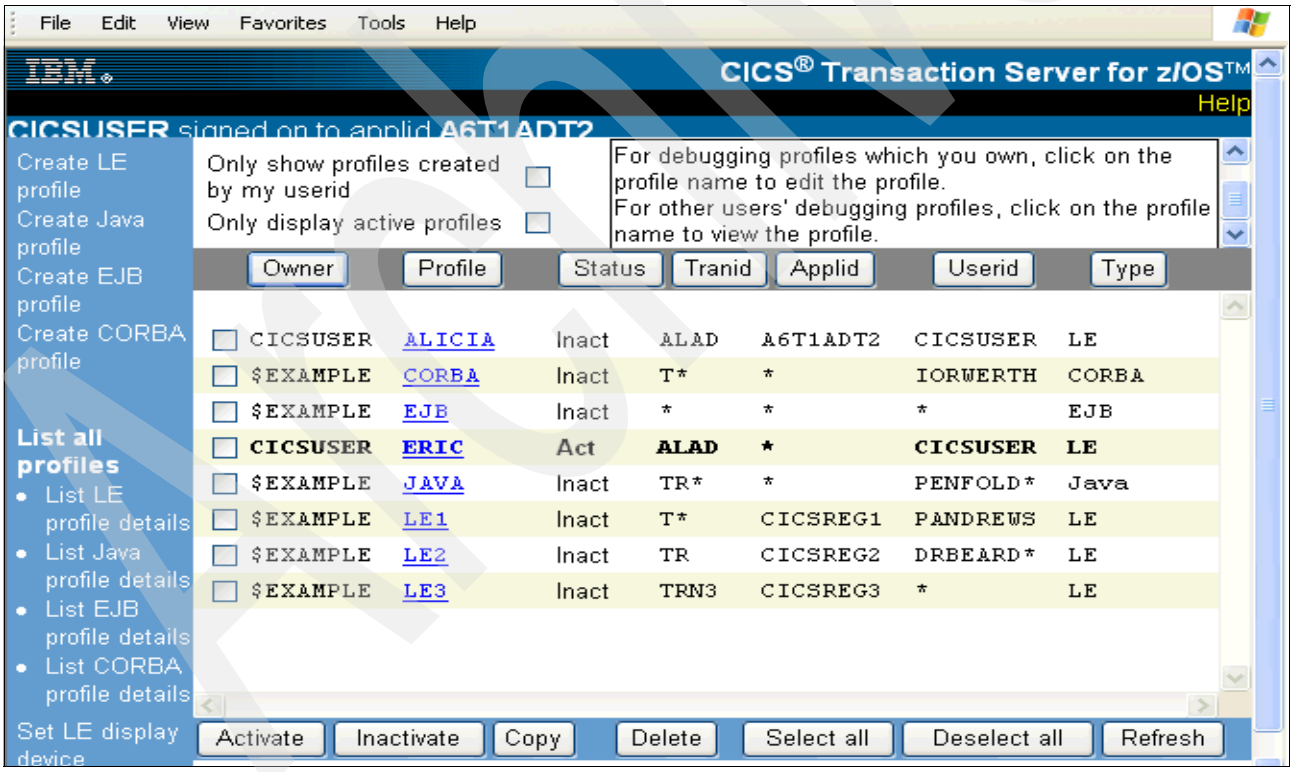


Figure 15-12 CADP Web interface - The List profiles page

When you use the debugging profile manager for the first time, CICS displays all profiles. Subsequently, CICS displays the profiles that were selected when you last used it.

If there are more profiles than can be displayed in the window, use the scrollbars to scroll backwards and forwards through the list. If you have no profiles, CICS displays an empty list

There are four variants of the List profiles page:

List LE profiles Lists only the compiled language profiles as shown in Figure 15-13.

List Java profiles Lists only the profiles for Java programs as shown in Figure 15-14.

List EJB profiles Lists only the profiles for enterprise beans as shown in Figure 15-15.

List CORBA profiles Lists only the profiles for stateless CORBA objects a shown in Figure 15-16.

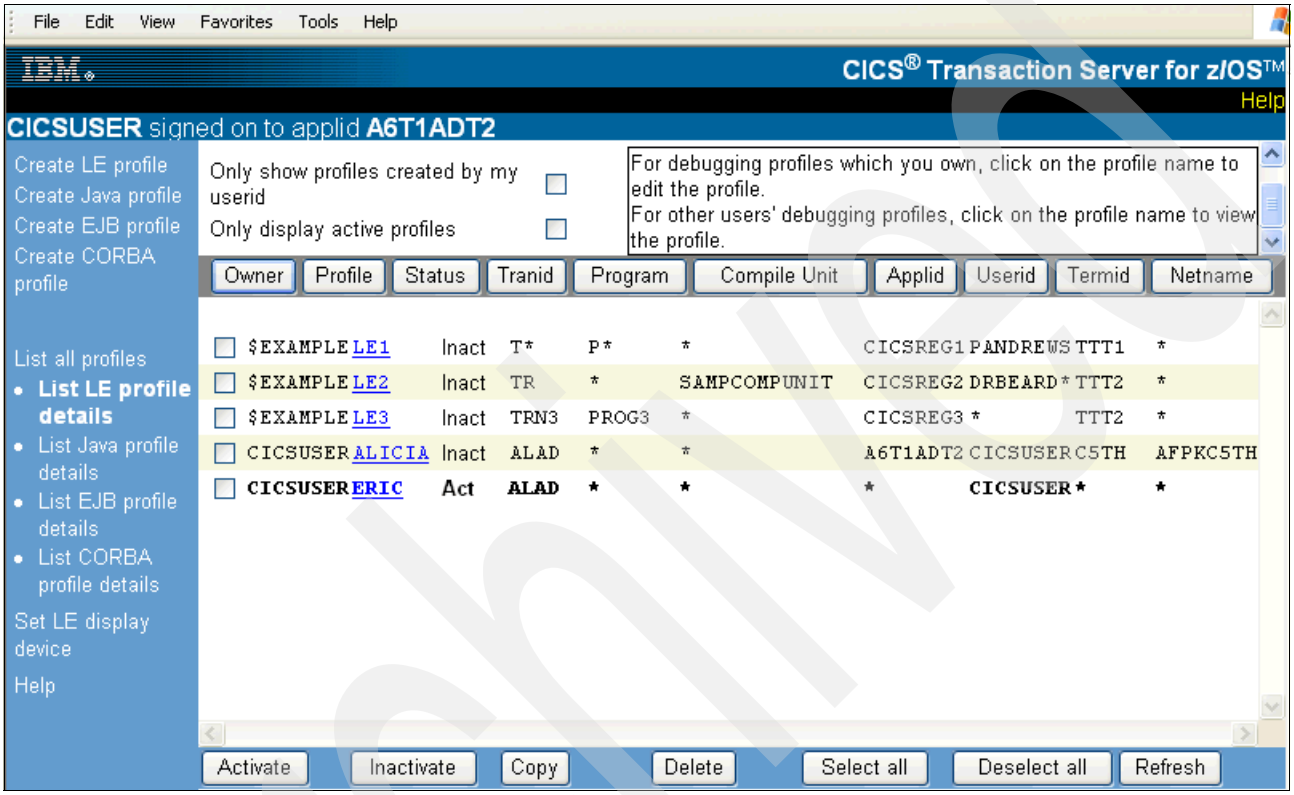


Figure 15-13 CADP - Web interface - LE profiles

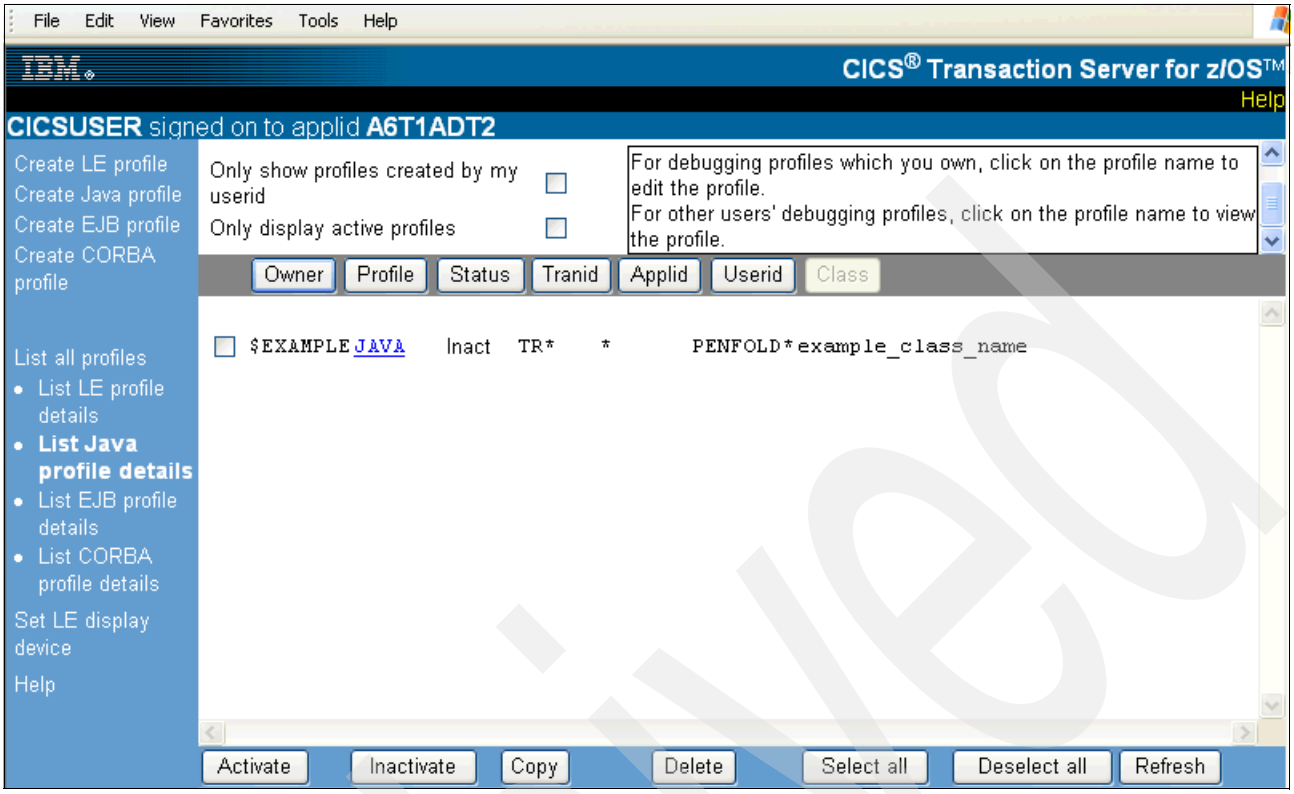


Figure 15-14 CADP Web interface - Java profiles

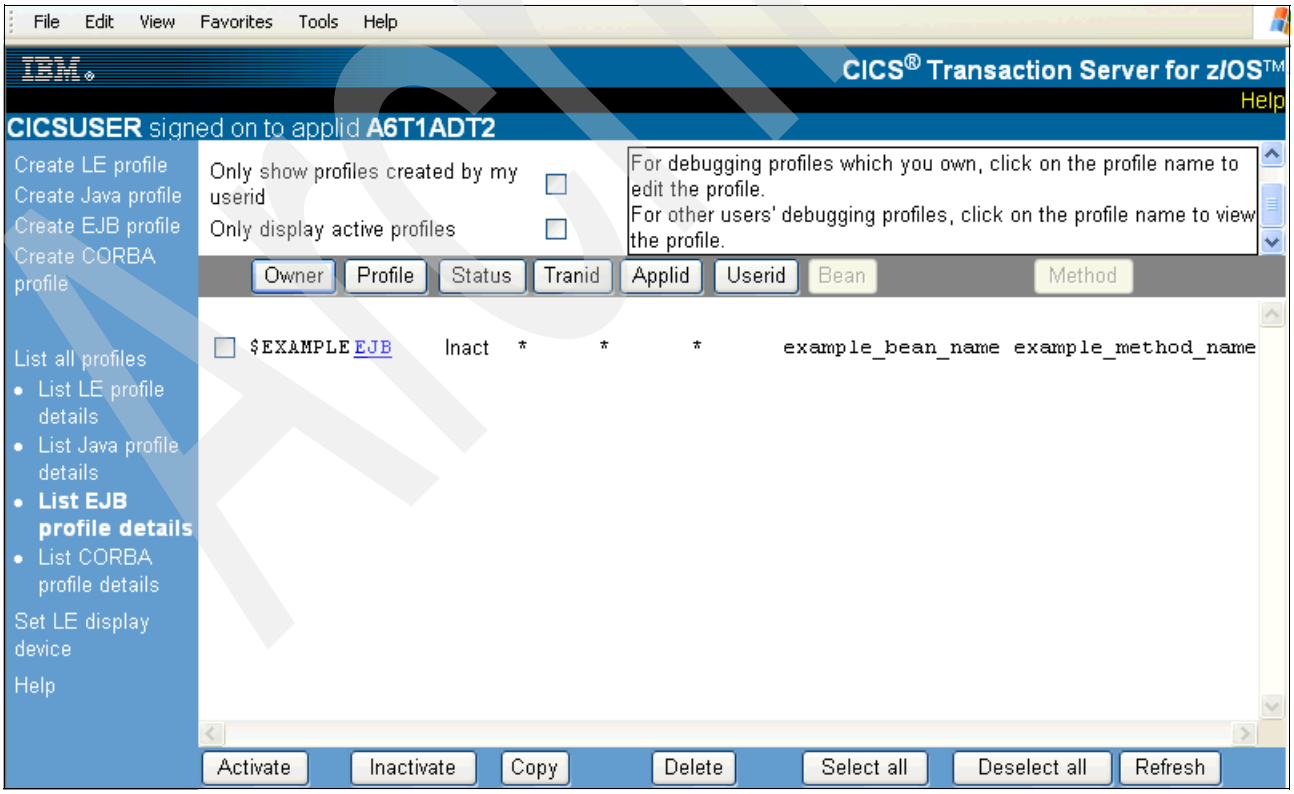


Figure 15-15 CADP Web interface - EJB profiles

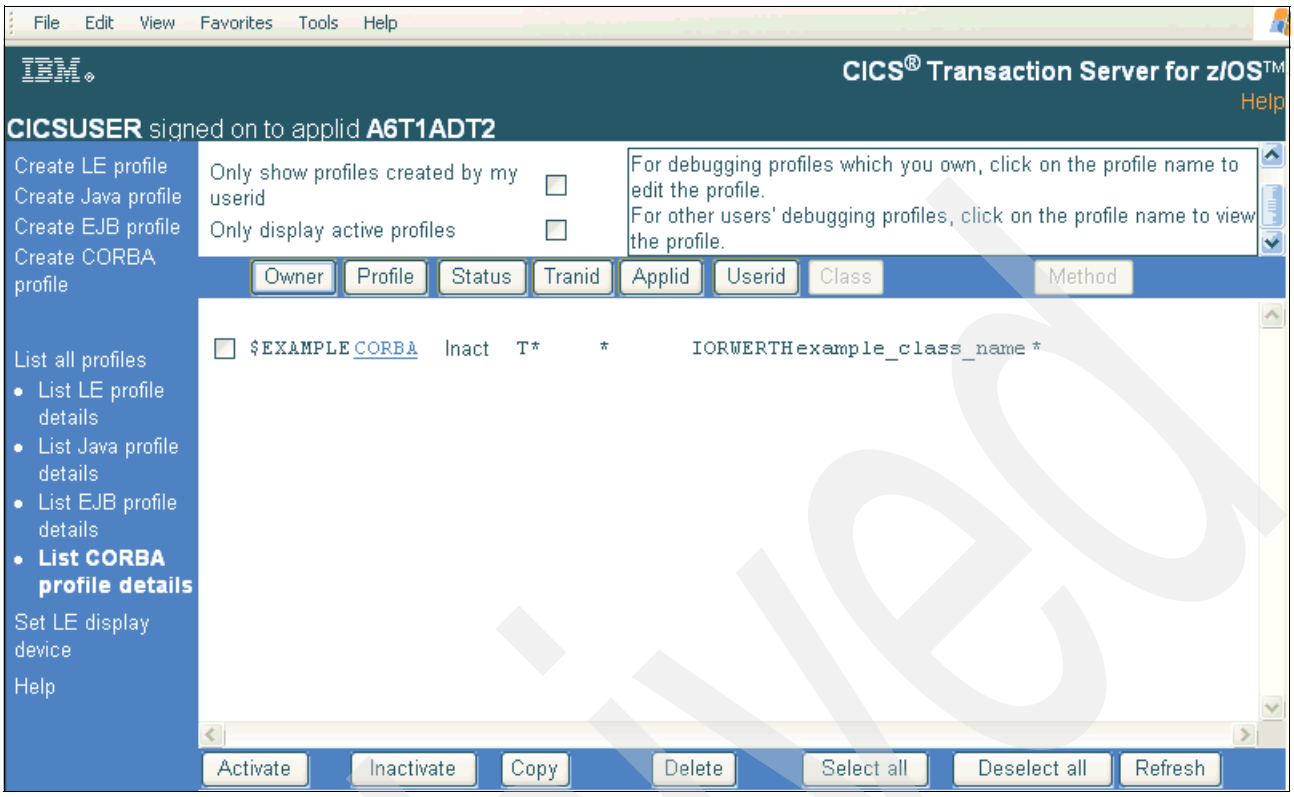


Figure 15-16 CADP Web interface - CORBA profiles

The behavior of these pages is identical to the “List profiles” page; however, the information displayed on each is specific to the type of profile.

The list contains selected information from the debugging profile. The columns on the page are:

Owner The userid of the profile owner; that is, of the user who created the profile.

Profile The name of the profile.

Status The status of the profile (**Act** for Active, or **Inact** for Inactive).

The following columns display information specified when the profile is created:

Tranid Displays the contents of the transaction field.

Program On the List profiles and List LE profiles pages only, displays the contents of the program field.

Compile Unit On the List profiles and List LE profiles pages only, displays the contents of Compile Unit field.

If the Compile Unit name is too long to display in the available space, the leading characters are displayed, followed by an ellipsis (...). To display the Compile Unit name in full, click on the profile name.

Applid Displays the contents of the Applid field.

Userid Displays the contents of the Userid field.

Termid On the List profiles and List LE profiles pages only, displays the contents of the Terminal field.

Type On the List profiles page only, displays the type of program specified in the debugging profile:

- **CORBA** - CORBA object
- **EJB** - Enterprise bean
- **Java** - Java program
- **LE** - Compiled language program

Netname On the List LE profiles page only, displays the contents of the Netname field.

Class On the List Java profiles and List CORBA profiles pages only, displays the contents of the Class field.

If the Class name is too long to display in the available space, the trailing characters are displayed, preceded by an ellipsis (...). To display the Class name in full, click the profile name.

Bean On the List EJB profiles page only, displays the contents of the Bean field.

If the bean name is too long to display in the available space, the leading characters are displayed, followed by an ellipsis. To display the bean name in full, click the profile name.

Method On the List EJB profiles and List CORBA profiles pages only, displays the contents of the Method field. If the Method name is too long to display in the available space, the leading characters are displayed, followed by an ellipsis. To display the Method name in full, click the profile name.

You can change the way information is displayed on the List profiles page, specifying which proiles are displayed and the order in which entries appear.

- **Selecting which profiles are displayed**

  Use the check boxes at the top of the page to select which debugging profiles are displayed. The options are:

  - Display all profiles
  - Display all profiles that you created
  - Display all active profiles
  - Display only active profiles that you created

- **Sorting the list**

  Use the buttons above each column to re-display the list in the sequence determined by the contents of the column. CICS uses the EBCDIC sorting sequence when it re-displays the list.

Your choice of which profiles are displayed, and your chosen sequence, are preserved and used the next time you use the debugging profile manager.

Buttons on the List Profiles page

The buttons at the head of the following columns are used to re-display the list of profiles in sequence:

- Owner
- Profile
- Status
- Tranid
- Program

- Compile
- Unit
- Applid
- Userid
- Termid
- Netname
- Type

The following buttons are inactive, and cannot be used to re-sequence the list of profiles:

- Class
- Bean
- Method

Other actions are performed using the buttons at the bottom of the List profiles page:

Activate	Activate selected profiles.
Inactivate	Inactivate selected profiles.
Copy	Copy selected profiles.
Delete	Delete selected profiles.
Select all	Selects all the profiles in the list.
Deselect all	Deselects all the profiles in the list.
Refresh	Refresh the List profiles page. The list is updated to show any changes that you, and other users, have made.

Creating a new LE debugging profile with the Web interface

To create a new LE debugging profile with the Web interface, complete the following steps:

1. Click **Create LE profile**; CICS displays the Create LE profile page as shown in Figure 15-17.
2. **Create EJB profile** and CICS displays the “Create EJB profile” page as shown in Figure 15-19
3.

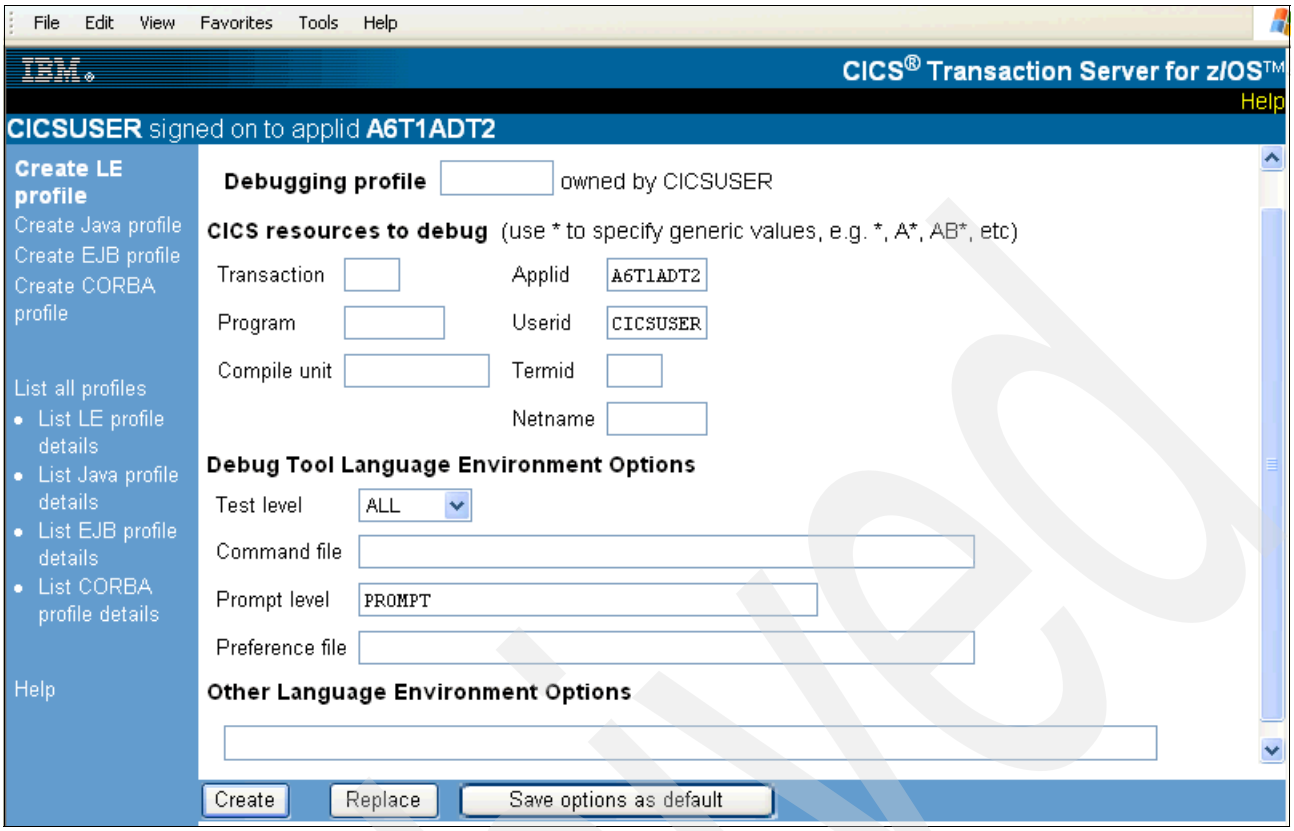


Figure 15-17 CADP - Web interface - Create LE profile page

- Fields on the Create LE profile page are:

  Debugging profile
  Specify the name of the profile. If you change the name of an existing profile, CADP creates a new profile with the new name, and leaves the original profile unchanged.

- The following fields specify which programs should trigger the start of a debugging session when the profile is active. You can specify a generic value for any of these fields.

  Transaction
  Specify a value in this field when you want to debug only those programs that run under the specified transaction.

  Program
  Specify a value in this field when you want to debug only the specified program.

  Compile unit
  Specify a value in this field when you want to debug only the specified compile unit.

  Applid
  Specify a value in this field when you want to confine debugging to programs that run in the specified CICS region.

  Userid
  Specify a value in this field when you want to confine debugging to programs that are being run by the specified user.

  Termid
  Specify a value in this field when you want to confine debugging to programs that are being run by the specified terminal.

Netname
Specify a value in this field when you want to confine debugging to programs that are being run by the specified netname.

- The following fields specify options that are passed to Debug Tool.

  Test level
  Specifies which conditions raised by your program will cause Debug Tool to gain control. You can enter the following values: All, Error, or None.

  Command file
  Specifies the primary commands file associated with the profile. You can specify the fully qualified name of a sequential dataset or a member of a partitioned data set.

  Prompt level
  Specifies whether an initial commands list is unconditionally executed during program initialization. Enter one of the following PROMPT, NOPROMPT or *command*.

  Preference file
  Specifies the preference file that Debug Tool uses when debugging programs that match this profile. You can specify the fully qualified name of a sequential dataset or a member of a partitioned data set.

  Other Language Environment options
  Specifies Language Environment runtime options for programs that match this profile. When a program is selected for debugging because it matches the profile, the runtime options specified will override other runtime options that you may have in effect.

4. Click **Create**. CICS checks that you have entered valid data.
5. Click **List profiles** to return to the List profiles page.

Buttons on the Create LE profile page

The buttons on the Create LE profile page are:

Create Create a new profile using the information entered on the page.

Replace Update an existing profile using the information entered on the page.

Save options as default
Save the contents of the following fields. The saved values are used by default each time you create an LE debugging profile.
- Test level
- Command file
- Prompt
- Preference file
- Language Environment options

Creating a new Java, EJB, or CORBA profile with the Web interface

To create a new Java, EJB, or CORBA profile with the Web interface, click the appropriate item on the screen:

- **Create Java profile**; CICS displays the Create Java profile page shown in Figure 15-18.
- **Create EJB profile**; CICS displays the Create EJB profile page as shown in Figure 15-19.
- **Create CORBA profile**; CICS displays the Create CORBA profile page as shown in Figure 15-20.

File Edit View Favorites Tools Help

IBM. CICS® Transaction Server for z/OS™

Help

CICSUSER signed on to applid **A6T1ADT2**

Create LE profile
Create Java profile
Create EJB profile
Create CORBA profile

List all profiles
- List LE profile details
- List Java profile details
- List EJB profile details
- List CORBA profile details

Help

Debugging profile owned by CICSUSER

CICS resources to debug (use * to specify generic values, e.g. *, A*, AB*, etc)

Transaction Applid A6T1ADT2 Userid CICSUSER

Java resources to debug (use * to specify generic values, e.g. *, set*, etc)

Class

Debugging options

JVM profile

Create Replace Save options as default

Figure 15-18 CADP - Web interface - Create Java profile page

File Edit View Favorites Tools Help

IBM. CICS® Transaction Server for z/OS™

Help

CICSUSER signed on to applid **A6T1ADT2**

Create LE profile
Create Java profile
Create EJB profile
Create CORBA profile

List all profiles
- List LE profile details
- List Java profile details
- List EJB profile details
- List CORBA profile details

Help

Debugging profile owned by CICSUSER

CICS resources to debug (use * to specify generic values, e.g. *, A*, AB*, etc)

Transaction Applid A6T1ADT2 Userid CICSUSER

Java resources to debug (use * to specify generic values, e.g. *, set*, etc)

Bean

Method

Debugging options

JVM profile

Create Replace Save options as default

Figure 15-19 CADP - Web interface - Create EJB profile page

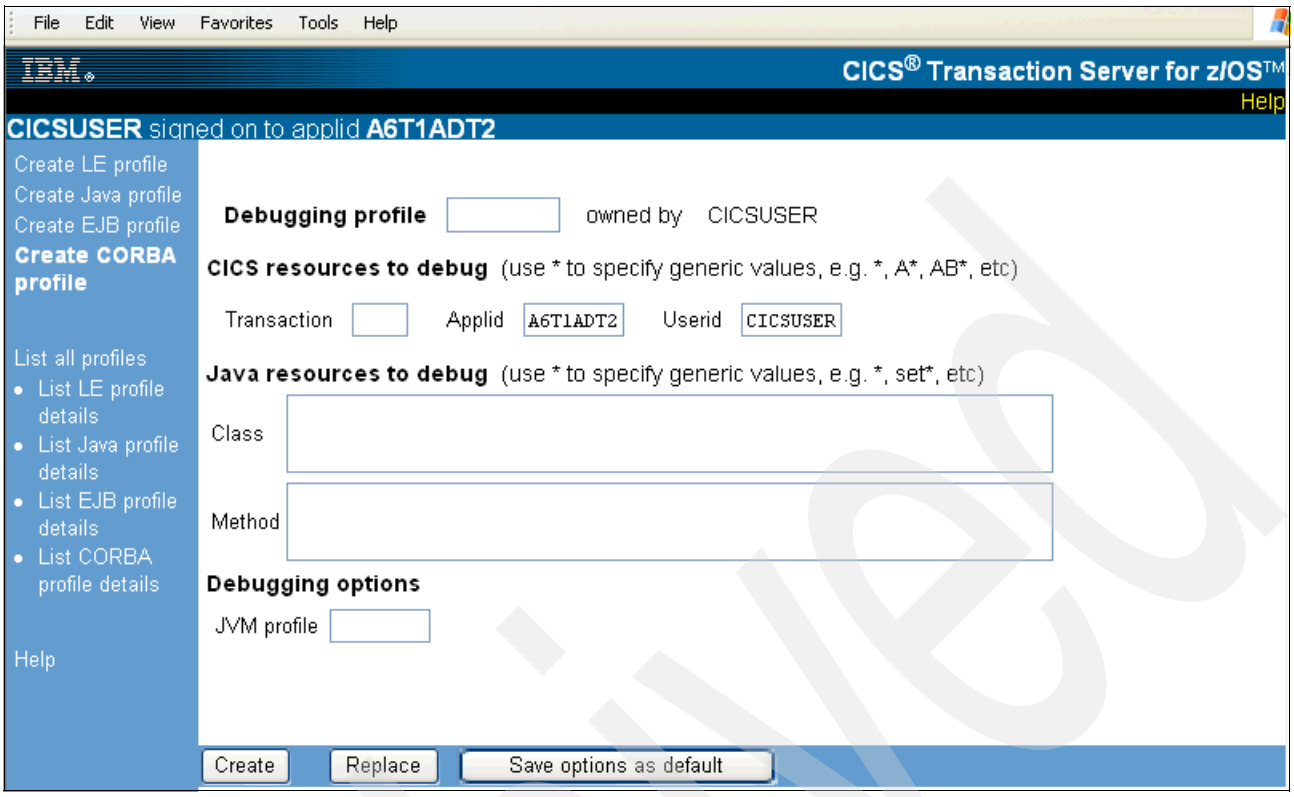


Figure 15-20 CADP - Web interface - Create CORBA profile page

- Fields on the previous three panels are:

  Debugging Profile
  Specify the name of the profile. If you change the name of an existing profile, CADP creates a new profile with the new name, and leaves the original profile unchanged.

- The following fields specify which programs should trigger the start of a debugging session when the profile is active. You can specify a generic value for any of these fields.

  Transaction
  Specify a value in this field when you want to debug only those programs that run under the specified transaction.

  Applid
  Specify a value in this field when you want to confine debugging to programs that run in the specified CICS region.

  Userid
  Specify a value in this field when you want to confine debugging to programs that are being run by the specified user.

- The following fields specify which Java resources should trigger the start of a debugging session when the profile is active:

  Type
  Specifies the type of Java resource that you want to debug:

  - **Java** Enter this value when you want to debug a Java program.
  - **EJB** Enter this value when you want to debug an enterprise bean.
  - **CORBA** Enter this value when you want to debug a stateless CORBA object.

  Class
  For Java programs and stateless CORBA objects only, specify a value in this field when

you want to debug only the specified class. You can specify a generic value if you want to debug a set of similarly-named classes.

Bean
For enterprise beans only, specify a value in this field when you want to debug only the specified bean. You can specify a generic value if you want to debug a set of similarly-named beans.

Method
For enterprise beans and stateless CORBA objects only, specify a value in this field when you want to debug only the specified method.

When an inbound request initiated by a Java remote method invocation is received, the value specified is compared with the mangled name in the inbound request to determine if the profile matches the request. If it is possible that mangling can take place, do not specify a method name in the debugging profile, but specify a generic method instead.

- The following field specifies the debugging options for this profile. You can save the value that you specify; the saved value is used by default each time you create a Java debugging profile.

  JVM profile
  Specifies the name of the JVM profile that is used for Java programs that match this profile. The profile should specify that the Java program is to run in debug mode. You cannot specify a generic value for this parameter.

Buttons on the Create LE profile page

The buttons on the three previous create profile pages are:

Create Create a new profile using the information entered on the page.

Replace Update an existing profile using the information entered on the page.

Save options as default
Save the contents of the following fields. The saved values are used by default each time you create a Java debugging profile.

How to start a Non-Language Environment Debug Tool under CICS

To debug CICS non-Language Environment programs, the Debug Tool Non-Language Environment Exits must have been previously started.

Passing runtime parameters into the non-LE debug session on startup

When you define your debugging profile using DTCN or CADP, you can pass a limited set of runtime options that will take effect in your non-Language Environment debugging session. These runtime options and their settings include the following:

- TEST/NOTEST: must be TEST
- TEST LEVEL: must be ALL
- Commands file
- Prompt Level: must be PROMPT
- Preference file
- You can also specify the following runtime options in a TEST string:
  - COUNTRY: Specifies a Country Code for Debug Tool. Valid country codes are:
    - **US** - United States of America
    - **JP** - Japan

– NATLANG: Specifies the National Language used to communicate with Debug Tool. Valid national language identifiers are:
  - **ENU** - English
  - **UEN** - Upper-case English
  - **JPN** - Japanese
– TRAP: to specify whether Debug Tool is to intercept Abends a valid value, is one of:
  - **ON** Enable Debug Tool to trap ABENDs.
  - **OFF** Prevent Debug Tool from trapping ABENDs; an ABEND causes abnormal termination of both Debug Tool and the program under test.

15.1.3 How to start Debug Tool under CICS

To use Debug Tool under CICS, you need to ensure that all of the required installation and configuration steps for CICS Transaction Server, Language Environment, and Debug Tool have been completed.

Choosing a debug mode

You can start Debug Tool in one of the following ways:

- **Single terminal mode**

  Debug Tool displays its screens on the same terminal as the application. This can be set up using CADP, DTCN, CEETEST, pragma, or CEEUOPT(TEST).

- **Dual terminal mode**

  Debug Tool displays its screens on a different terminal than the one used by the application. This can be set up with CADP, DTCN or CEDF.

  If you are using Debug Tool in a multiple-CICS region environment and sharing the EQADTCN2 temporary storage queue for DTCN or DFHDPFMB and DFHDPFMP for DCADP, set your profile to a debugging Display ID that is located in the same CICS region that the task you want to debug will run in.

- **Batch mode**

  Debug Tool does not have a terminal, but uses a commands file for input and writes output to the log. This can be set up using CADP, DTCN, CEETEST, pragma, or CEEUOPT(TEST).

- **Remote debug mode**

  Debug Tool works with a remote debugger to display results on a graphical user interface. This can be set up using CADP, DTCN, CEETEST, pragma, or CEEUOPT(TEST).

Methods for starting Debug Tool under CICS

There are several different mechanisms available to start Debug Tool under CICS. Each mechanism has its own advantages. The mechanisms are as follows:

- **DTCN** is a full-screen CICS transaction that allows you to dynamically modify any Language Environment TEST or NOTEST runtime option with which your application was originally link-edited. You can also use DTCN to modify other Language Environment runtime options that are not specific to Debug Tool.
- **CADP** is a CICS transaction that enables you to manage debugging profiles. This transaction is available with CICS Transaction Server for z/OS Version 2 Release 3. CADP has the following advantages over DTCN:
  – With CADP, multiple profiles with a single program name can be added from the same display device. There is no limit to the number of profiles supported. With DTCN, a

single profile, with up to eight program IDs, can be added from a single display device. In either case, the program names can be specified with wild cards.

  - CADP provides the same abilities as DTCN for managing debug profiles for Language Environment applications. CADP can also help manage debug profiles for Java applications, Enterprise Java Beans (EJBs), and CORBA stateless objects.
  - CADP profiles are persistent, and are kept in VSAM files. Persistence means that if the CADP profile was present before a CICS region is restarted, the CADP profile will be present after the CICS region is restarted. For DTCN profiles, if the CICS region that owns the temporary storage queue where the debugging profiles were defined is restarted, the DTCN profiles must be added again after the region is restarted.

- Language Environment **CEEUOPT** module link-edited into your application, containing an appropriate TEST option, tells Language Environment to start Debug Tool every time the application is run. This mechanism can be useful during initial testing of new code when you will want to run Debug Tool frequently.
- A **compiler directive** within the application, such as #pragma runopts(test) (for C and C++) or CALL CEETEST.

  These directives can be useful when you need to run multiple debug sessions for a piece of code that is deep inside a multiple enclave or multiple CU application. The application runs without Debug Tool until it encounters the directive, at which time Debug Tool is started at the precise point that you specify. With CALL CEETEST, you can even make the invocation of Debug Tool conditional, depending on variables that the application can test.
- **CEDF** utility where you can start a debug session in Dual Terminal mode alongside CEDF, using a special option on the CEDF command.

  This mechanism does not require you to change the application link-edit options or code, so it can be useful if you need to debug programs that have been compiled with the TEST option, but do not have invocation mechanisms built into them.

Starting Debug Tool by using DTCN

DTCN profiles contain the identifiers (IDs) of CICS resources to debug. These resource IDs can be Terminal, Transaction, Program, or User.

> **Note:** To debug a PL/I or C/C++ CICS program by using DTCN to start Debug Tool, update the link-edit step to include the EQADCCXT member from the Debug Tool library hlq.SEQAMOD into the program's main load module. If you are using CICS TS 2.3 and CADP, you do not need to include EQADCCXT.

When CICS programs are started, Debug Tool monitors the tasks to find a task that contains a combination of Terminal ID, Transaction ID, Program ID, and User ID that matches a repository profile item. Debug Tool selects the best matching profile (the one with the greatest number of resource IDs that match the active task).

If two tasks have an equal number of matching resource IDs, the older debug profile is selected. If this situation occurs, an error message is sent to the system console, suggesting that DTCN users should specify additional resource qualification.

DTCN not only provides the capability to specify *what* to debug by specifying debug resource IDs, DTCN also provides the capability to specify *how* the debug session will run, for example, whether a mainframe (MFI) or workstation (VAD) debug session is desired.

When a DTCN profile is active for a full-screen mode debugging session, Debug Tool preserves in the profile most of the breakpoint information for that session. When the DTCN profile is deleted, the breakpoint information is deleted.

How to end your CICS debugging session

After you have finished debugging your program, use DTCN again to turn off your debug profile by pressing PF6 to delete your debug profile and then pressing PF3 to exit. You do not need to remove EQADCCXT from the load module; in fact, it's a good idea to leave it there for the next time you want to start Debug Tool.

Starting Debug Tool using CAPD

CADP is an interactive transaction supplied by CICS Transaction Server for z/OS Version 2 Release 3. CADP helps you maintain persistent debugging profiles. When you start the CICS Transaction Server for z/OS Version 2 Release 3 region and you set the DEBUGTOOL system initialization parameter to YES, Debug Tool uses the CADP profile repository instead of the DTCN profile repository to find a matching debugging profile.

If you start the CICS region with DEBUGTOOL=YES, CADP will be the method for defining profiles; those defined in DTCN will not be used to start Debug Tool sessions.

You can use the CEMT transaction to set DEBUGTOOL dynamically as follows:

- To use CAPD:

  ```
  CEMT SET DEBUG
  ```

- To use DTCN:

  ```
  CEMT SET NODEBUG
  ```

Activating, inactivating, copying, and deleting profiles with the 3270 interface

To activate, inactivate, copy, or delete debugging profiles, start with the "List debugging profiles" screen, and follow these steps:

1. Press PF2 to ensure that the display includes the profiles you want to activate.
2. Press PF7 and PF8 to scroll to a profile that you want to activate.
3. Type **A** (for Activate), **I** (for Inactivate), **C** (for Copy) or **D** (for Delete) in the field to the left of the profile name.
4. Repeat steps 2 and 3 to select all the profiles you want to activate.
5. Press Enter.

By default, if any of the selected profiles to be activated is for a compiled language program, CICS displays the Set LE debugging display device screen.

If none of the selected profiles is for a compiled language program, CICS refreshes the List Debugging Profiles screen.

You can choose not to see the Set LE debugging display device screen when you activate profiles.

> **Note:** If you change a profile while it is active, the changes take effect immediately: the next time a program is started, the changed parameters are used to decide if the program should run under the debugger's control.

From the List Debugging Profiles screen, you can activate, inactivate, delete, and copy debugging profiles by typing the appropriate action character (`A`, `I`, `D`, and `C` respectively) in the field to the left of the profile name. You can combine these actions on the List Debugging Profiles screen as shown in Figure 15-21.

```
CADP      -    CICS Application Debugging Profile Manager   -     CICSC31F

List Debugging Profiles    (A=Activate,I=Inactivate,D=Delete,C=Copy)

  Owner    Profile  S Tran Program  Compile Unit Applid   Userid   Term Type
_ $EXAMPLE EJB      I *                          *        *             EJB
I CHABERT  ERC2#PGM A *    ERC2PGM* *            CICSC31F CHABERT  *    LE
D CHABERT  USERID   I *    *        *            CICSC31F ERIC     *    LE
A CHABERT  ERC1#TRN I ERC1 *        *            CICSC31F *        *    LE
_ $EXAMPLE CORBA    I T*                         *        IORWERTH      Corb
_ $EXAMPLE LE1      I T*   P*       *            CICSREG1 PANDREWS TTT1 LE
_ $EXAMPLE LE2      I TR   *        SAMPCOMPUN + CICSREG2 DRBEARD* TTT2 LE
C $EXAMPLE JAVA     I TR*                        *        PENFOLD*      Java
_ $EXAMPLE LE3      I TRN3 PROG3    *            CICSREG3 *        TTT2 LE




9   profile(s). All profiles shown
Enter=Process PF1=Help 2=Filter 3=Exit 4=View 5=Create LE 6=Create Java
                 9=Set display device 10=Edit 11=Sort
                                                                    14/002
```

Figure 15-21 CADP - 3270 interface - Profile management

CICS displays the Set LE Debugging Display Device to specify which display device you will use to interact with the debugger for the profile you want to activate, as shown in Figure 15-22.

=

```
CADP      -    CICS Application Debugging Profile Manager   -     CICSC31F

Set LE Debugging Display Device

Debugging Display Device
Session Type            ==>  g270                        (3270,TCP)
3270 Display Terminal   ==>  0026

TCP/IP Name Or Address
==>
==>
==>
==>
Port                    ==>  08000

Type of socket communication ==> Single                  (Single,Multiple)


Display this panel on LE profile activation ==> YES




Enter=Save and return PF1=Help 3=Exit 12=Cancel
                                                                    06/033
```

Figure 15-22 CADP - 3270 - Set LE display device

Complete the details of the display device that you want to associate with the profile. The fields on the Set LE debugging display device screen are:

Session Type - Specifies how you will interact with Debug Tool:

3270 You will interact with Debug Tool using a 3270 type terminal. Specify the terminal ID in the Display ID field. This is the default value.

TCP You will interact with Debug Tool using a debugging client on your workstation. The client will communicate with Debug Tool using TCP/IP. Specify the port number at which the client listens for a connection in the Port field.

3270 Display Terminal
When the session type is 3270, specify the terminal ID of the terminal with which you will interact with Debug Tool. The default value is the ID of the terminal at which you running CADP.

TCP/IP Name Or Address
When the session type is TCP, specify the IP address or name of the host where the debugging client is running.

Port
When the session type is TCP, specify the port number at which the debugging client listens for a connection. Specify a value in the range 0 to 65535. The default is 8000.

Type of socket communication
When the session type is TCP, specifies whether the debugging client and debugging server will communicate using a single socket or more than one socket.

Single Use a single socket for communication. This is the default value, and is the preferred value when you use a WebSphere Studio product as your debugging client.

Multiple Use more than one socket for communication. You must specify this value when you use a VisualAge product as your debugging client.

Display this panel on LE profile activation
Specifies whether you want to display the Set LE Debugging display device screen whenever you activate debugging profiles for compiled language programs:

YES Display the Set LE Debugging display device screen whenever debugging profiles are activated. This is the default behavior.

NO Do not display the Set LE debugging display device screen whenever debugging profiles are activated. The display device that you specify will be associated with all the profiles that you activate.

Activating debugging profile with the Web interface

Starting with the List profiles page, follow these steps:

1. Select the profile and click **Activate** if it is a profile for a compiled language program. The Set LE display device page is displayed as shown in Figure 15-23.

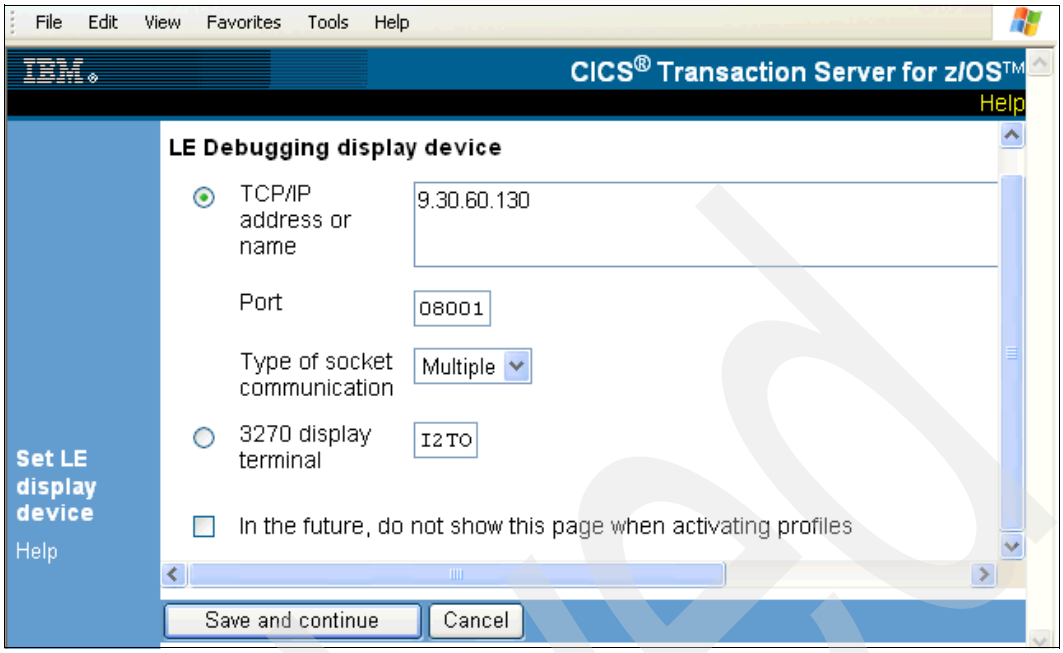


Figure 15-23 CADP - Web interface - Set LE display device page

2. Complete the details of the display device that you want to associate with the profile. You can use a 3270 terminal or a debugging tool on a workstation.

   Use the radio buttons to select how you will interact with the debugger. The fields on the Set LE display device page are:

   TCP/IP address or name
   Specifies that you will interact with the debugger using a debugging client on your workstation. Supply the following information:

   – The IP address or name of the host where the debugging client is running. By default, CICS inserts the IP address of the client which is running the browser, or—if there is a firewall between the browser and CICS—the IP address of the firewall.

   – The port number at which the debugging client listens for a connection. Specify a value in the range 0 to 65535. The default is 8000.

   Type of socket communication
   For a debugging client on your workstation, specifies whether the debugging client and debugging server will communicate using a single socket or more than one socket:

   – **Single** - Use a single socket for communication. This is the default value, and is the preferred value when you use a WebSphere Studio product as your debugging client.

   – **Multiple** - Use more than one socket for communication. You must specify this value when you use a VisualAge product as your debugging client.

   3270 display terminal
   Specifies that you will interact with the debugger using a 3270 type terminal. Supply the terminal ID of the terminal at which you will interact with the debugger.

3. Click **Save and continue**. CICS saves the display device settings, and displays the List profiles page.

Starting Debug Tool using CEDF

No specific preparation is required to use CEDF to start Debug Tool other than compiling the application with the appropriate compiler options and saving the source/listing.

CEDF has an **,I** option that starts Debug Tool. This option starts both EDF and Debug Tool in Dual Terminal mode. In Dual Terminal mode, EDF and Debug Tool screens are displayed on the terminal where you issue the CEDF command; application screens are displayed on the application terminal.

> **Note:** You need to know the ID of each terminal. One way to get this information is by using the CEOT transaction.

To start Debug Tool, enter the CEDF transaction as follows, where *xxxx* is the terminal on which you want to start the transaction to be debugged:

```
CEDF xxxx,ON,I
```

This terminal is where the application is started. CICS displays the following message:

```
TERMINAL C5TZ: OPTION I... REQUESTS THE DEBUG TOOL : EDF MODE ON. DEBUG TOOL ON
```

Then, on the *xxxx* terminal, enter: *TRAN*, where *TRAN* is the ID for the transaction being debugged.

Once the command is entered, Debug Tool will be started for all Language Environment-enabled programs that are running on the terminal where Debug Tool is started. Debug Tool will continue to be active on this terminal, even if you turn off EDF.

Using this approach, you are able to debug CICS statements using standard CEDF functions, and language statements with Debug Tool. On the terminal where CEDF has been issued, you can switch between EDF and Debug Tool displays as shown in Figure 15-24.

```
_TRANSACTION: ALAD PROGRAM: MYTRADMV TASK: 0002471 APPLID: A6T1ADT1 DISPLAY:  00
  STATUS:  PROGRAM INITIATION

     EIBTIME      = 235543
     EIBDATE      = 0104323
     EIBTRNID     = 'ALAD'
     EIBTASKN     = 2471
     EIBTRMID     = 'C5TZ'

     EIBCPOSN     = 4
     EIBCALEN     = 0
     EIBAID       = X'7D'                                        AT X'001400EA'
     EIBFN        = X'0000'                                      AT X'001400EB'
     EIBRCODE     = X'000000000000'                              AT X'001400ED'
     EIBDS        = '........'
+    EIBREQID     = '........'


 ENTER:  CONTINUE
 PF1 : UNDEFINED           PF2 : SWITCH HEX/CHAR     PF3 : END EDF SESSION
 PF4 : SUPPRESS DISPLAYS   PF5 : WORKING STORAGE     PF6 : USER DISPLAY
 PF7 : SCROLL BACK         PF8 : SCROLL FORWARD      PF9 : STOP CONDITIONS
 PF10: PREVIOUS DISPLAY    PF11: EIB DISPLAY         PF12: UNDEFINED
                                                                   01/001
```

Figure 15-24 CEDF - Debug Tool display (1)

All CEDF functions are available by viewing or scrolling down Figure 15-25.

```
COBOL     LOCATION: MYTRADMV :> 647.1
Command ===>                                                  Scroll ===> PAGE
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 5
****************************** TOP OF MONITOR ********************************
0001  1 DATA1   +000000000
0002  2 COMPANY-NAME-DATA   '                    '
0003  3 COMPANY-DATA-1   '                    '
0004  4 ********** AUTOMONITOR **********
0005    02 MYTRADMV:>DATA1    +000000000
****************************** BOTTOM OF MONITOR *****************************
SOURCE: MYTRADMV -1----+----2----+----3----+----4----+----5-- LINE: 643 OF 1616
     643                MOVE SPACES TO  COMPANY-DATA-1                      1
     644                MOVE SPACES TO  COMPANY-DATA-2                      1
     645                MOVE SPACES TO  COMPANY-DATA-3                      1
     646                MOVE SPACES TO  COMPANY-DATA-4                      1
     647                ADD 1 TO DATA1                                      0
     648        *       MOVE SPACES TO USER10                               .
LOG 0----+----1----+----2----+----3----+----4----+----5----+---- LINE: 48 OF 51
0048    LIST COMPANY-NAME-DATA ;
0049  MONITOR
0050    LIST COMPANY-DATA-1 ;
0051  SET AUTOMONITOR ON ;
PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEAR
PF  7:UP         8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEVE
                                                                     03/002
```

Figure 15-25 CEDF - Debug Tool display (2)

Starting Debug Tool using CEEUOPT

To request that Language Environment start Debug Tool every time the application is run, assemble a CEEUOPT module with an appropriate TEST runtime option. It is a good idea to link-edit the CEEUOPT module into a library and just add an INCLUDE LibDD(CEEUOPT) statement to the link-edit options when you link your application. Whenever the application program runs Debug Tool will be started.

Debug Tool runs in the mode defined in the TEST runtime option you supplied, normally Single Terminal mode, although you could provide a primary commands file and a log file and not use a terminal at all.

Starting Debug Tool using compiler directives

When compile directives are processed by your program, Debug Tool will be started in single terminal mode (this method supports only single terminal mode).

If you don't want to compile your program with hooks, you can use CEETEST calls to start Debug Tool at strategic points in your program. If you decide to use this method, you still need to compile your application so that symbolic information is created.

The syntax for CEETEST is as shown in Figure 15-26.

```
►►──void──CEETEST──(──┬──────────────────┬──,──┬────┬──)──;──────────────►◄
                      └─string_of_commands─┘     └─fc─┘

For COBOL:

►►──CALL──'CEETEST'──USING──string_of_commands──,──fc──;────────────────►◄

For PL/I

►►──CALL──CEETEST──(──┬──*──────────────┬──,──┬──*──┬──)──;──────────────►◄
                      └─string_of_commands─┘     └─fc─┘
```

Figure 15-26 CEETEST syntax

- **string_of_commands (input)**

  Halfword-length prefixed string containing a Debug Tool command list, string_of_commands is optional. If Debug Tool is available, the commands in the list are passed to the debugger and carried out. If string_of_commands is omitted, Debug Tool prompts for commands in interactive mode. For Debug Tool, remember to use the continuation character if your command exceeds 72 characters.

- **fc (output)**

  A 12-byte feedback code, optional in some languages, that indicates the result of this service. Language Environment provides a callable service called CEEDCOD to help you decode the fields in the feedback code. Requesting the return of the feedback code is recommended.

For C and C++ and COBOL, if Debug Tool was started through CALL CEETEST, the GOTO command is only allowed after Debug Tool has returned control to your program via STEP or GO.

15.1.4 How to debug CICS programs

The program source file (for C and C++ and Enterprise PL/I), the program listing (for COBOL and all other PL/I), or separate debug file (for COBOL) must be retained in a permanent data set for Debug Tool to read when you debug your program.

> **Note:** For C and C++, and Enterprise PL/I, it is the input to the compiler that needs to be retained. To enhance performance when using Debug Tool, use a large block size when saving these files

Debug modes under CICS

Debug Tool can run in several different modes, providing you with the flexibility to debug your applications in the way that suits you best. These modes include:

- **Single terminal mode**

  A single 3270 session is used by both Debug Tool and the application, swapping displays on the terminal as required. As you step through your application, the terminal shows Debug Tool screens, but when an EXEC CICS SEND command is issued, that screen will be displayed. Debug Tool holds that screen on the terminal for you to review; simply press

Enter to return to a Debug Tool screen. When your application issues EXEC CICS RECEIVE, the application screen again appears, so you can fill in the screen details.

- **Dual terminal mode**

  This mode can be useful if you are debugging screen I/O applications. Debug Tool displays its screens on 3270 session that is separate from the terminal displaying the application. You step through the application using the Debug Tool terminal and, whenever the application issues an EXEC CICS SEND, the screen is sent to the application display terminal. When the application issues an EXEC CICS RECEIVE, the Debug Tool terminal will wait until you respond to the application terminal.

- **Interactive batch mode**

  Use this mode if you are debugging a transaction that does not have a terminal associated with it. The transaction continues to run without a CICS principal facility, but Debug Tool screens are displayed on a 3270 session that you name.

- **Noninteractive batch mode**

  In this mode, Debug Tool does not have a terminal associated with it at all. It receives its commands from a command file and writes its results to a log file. This mode is useful if you want Debug Tool to debug a program automatically.

Preventing Debug Tool from stopping at EXEC CICS RETURN

Debug Tool stops at EXEC CICS RETURN and displays the following message:

```
CEE0199W The termination of a thread was signaled due to a STOP statement.
```

To prevent Debug Tool from stopping at every EXEC CICS RETURN statement in your application and suppress this message, set the TEST level to ERROR by using the SET TEST ERROR command.

Saving settings while debugging a pseudo-conversational program

If you change the Debug Tool display settings while you debug a pseudo-conversational CICS program, Debug Tool might restore the default settings. To ensure that your changes remain in effect every time your program starts Debug Tool, store your display settings in the preferences file or the commands file.

Saving and restoring breakpoints

When breakpoints are set in a CICS transaction, Debug Tool saves these breakpoint settings and restores them the next time this transaction is started. However, saving and restoring of breakpoints in assembler compilation units is not currently supported.

Restrictions when debugging under CICS

The following restrictions apply when debugging programs with the Debug Tool in a CICS environment.

- Debug Tool supports the use of CRTE terminals if both the application and Debug Tool share the terminal as a principal facility in single terminal mode. CICS does not permit the use of a CRTE terminal by Debug Tool if the terminal is not the application task's principal facility (which is the case in Dual terminal mode).
- The __ctest() function with CICS does nothing.
- The CDT# transaction is a Debug Tool service transaction used during Dual terminal mode debugging and is not intended for activation by direct terminal input. If CDT# is started via terminal entry, it will return to the caller (no function is performed).
- Applications that issue EXEC CICS POST cannot be debugged in Dual terminal mode.

- References to ddnames are not supported. All files, including the log file, USE files, and preferences file, must be referred to by their full data set names.
- The commands TSO, SET INTERCEPT, and SYSTEM cannot be used.
- CICS does not support an attention interrupt from the keyboard.
- The log file (INSPLOG) is not automatically started. You need to use the SET LOG ON command.
- Ensure that you allocate a log file big enough to hold all the log output from a debug session because the log file is truncated after it becomes full. (A warning message is not issued before the log is truncated.)
- Save files (INSPSAFE) are not used under CICS.
- You can start Debug Tool when a Non-Language Environment assembler program under CICS starts by defining a debug profile using CADP or DTCN. However, Debug Tool will only start on a CICS Link Level boundary, such as when the first program of the task starts or for the first program to run at a new Link Level. For profiles defined in CADP or DTCN which list a Non-Language Environment assembler program name that is dynamically called using EXEC CICS LOAD/CALL, Debug Tool will not start. Non-Language Environment assembler programs that are called in this way are identified by Debug Tool in an already-running debugging session and can be stopped by using a command like AT APPEARANCE or AT ENTRY. However, they cannot be used to trigger a Debug Tool session initially.

Support for CALL %CEBR and CALL %CECI (MFI only commands)

Call %CEBR Invoke CICS Temporary Storage Browser Program. It browses CICS Temporary Storage Queues (and supports “importing” of Transient Data Queues as TS queues for TD browsing.

Call %CECI Invoke CICS Command Level Interpreter Program. It gives the user the capability to dynamically execute EXEC CICS commands (such as READ of a VSAM file) while the transaction runs.

15.2 Debug Tool and DB2

In this section we discuss the following topics:

- Compiling source for both DB2 and Debug Tool
- Invoking the DB2 application specifying the TEST run-opt
- Using DTSU to invoke your DB2 application
- Considerations for DB2 stored procedures

15.2.1 Compiling for DB2 and Debug Tool

There is one basic issue when compiling DB2 programs for use with Debug Tool. This is the fact that the DB2 pre-compiler will expand your source code, and then the debug information will be generated based on that expansion.

There are several solutions to this problem:

1. As of DB2 V7, the DB2 product has provided utilities for compilers to use to do the DB2 expansion on the fly. When the compiler front ends do this expansion, you do not need to keep track of any extra code, and the code you are looking at when debugging matches the source files you normally work with. Look for options named DB2 or SQL on your compilers.
2. If you do not have a compiler that supports an integrated DB2 pre-compiler, you need to prepare your code for debugging in a two-step process:
   a. Pre-compile the source and save the generated source in a file. This file will be needed when debugging C and PL/I (and may potentially be useful for COBOL).
   b. Compile the source as usual for debugging.

15.2.2 Invoking a DB2 application using the TEST runtime option

To use the TSO command interface to start executing your application program, issue the DSN command to invoke DB2 a shown in Example 15-8.

Example 15-8 Debug Tool and DB2 - Using DSN command

```
DSN SYSTEM(DBA1) RUN PROGRAM(TRADERD2) PLAN(TRADERD2)
LIB('DAVINR2.DEMOS.PDPAK.LOAD')
PARMS('/NOTEST(ALL,*,PROMPT,MFI:INSPREF) ')
```

To use the TSO Call Access Facility (CAF), you link-edit the CAF interface module, DSNALI, with your application program. Then, issue the TSO CALL command for your application program, and include the TEST runtime option as a parameter as shown in Example 15-9.

Example 15-9 Debug Tool and DB2 - Using CAFF interface

```
CALL ‘change.mgmt.test.loadlib(progname)’ ‘/TEST’
```

15.2.3 Using DTSU to invoke your DB2 application

The panel shown in Figure 15-27 lets you select Initialize New setup file for DB2. This is done by selecting option **2** from the Debug Tool main panel.

```
------------------- Debug Tool Foreground - Edit Setup File ------------------
Command ===> ____________________________________________________________________


Setup File Library:
   Project . . . GRACINE
   Group . . . . TESTDT  . . . ________ . . . ________ . . . ________
   Type  . . . . DTSF
   Member  . . . ________        (Blank or pattern for member selection list)
                                 (or existing or new member name)

Other Data Set Name:
   Data Set Name . . . ____________________________________________________
   Volume Serial . . . ______    (If not cataloged)


/ Initialize New setup file for DB2 (/)




 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 15-27 The Debug Tool Edit Setup File Screen

Specify a file to allocate (if the given PDS does not exist) and you will be presented with the Edit Setup File panel, as shown Figure 15-28.

```
EDIT - Edit Setup File 'GRACINE.TESTDT.DTSF(DB2T)'            Row 1 to 1 of 1
Command ===> ___________________________________________ Scroll ===> PAGE

Modify information and use the  Run command to execute,
        or the  Submit command to submit to Batch.
     Press HELP for a list of all available commands.

DB2 System Id: ______
Program Name _________  Plan _________

Choose the format of your parameter string:

1 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs    - Run-time Options  / Program Arguments
_ Enter / to modify parameters ____________________________________________

Other DB2 options:
   DSN command options ___________________________________________________
   RUN command options ___________________________________________________

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)            DISP
 F1=Help      F3=Exit      F4=Run       F7=Backward  F8=Forward  F10=Submit
F12=Cancel
```

Figure 15-28 Creating a new setup file for DB2

On this panel, you can define your DB2 subsystem ID, the program, and the plan name. Eventually you can add DB2 DSN and DB2 RUN command options, and add the DD statements for the resources needed. Then enter the DTSU run command (PF4) to run your DB2 application.

If you intend to run your DB2 program using the TSO CAF facilities, do not select for DB2; instead, use DTSU for normal BATCH programs. (Remember to link DSNALI to your DB2 application).

If you are using DTSU, you do not need your own CLISTs for the file allocations. We recommend that you use DTSU.

15.2.4 DB2 stored procedures considerations

Debug Tool supports debugging of DB2 stored procedures. To define the runtime option TEST, you could use a CEEUOPT containing only this TEST option. However, a CEEUOPT will probably not work in most situations because most DB2 stored procedures may be defined as Type=SUB and CEEUOPT is ignored by Language Environment for non-MAIN programs. We recommend that you use the DB2-provided RUNOPTS field of the DB2 catalog, by using the appropriate DB2 SQL commands.

For DB2 Version 6 and later, the stored procedure is defined as follows:

- Create procedure: `traderb`
- Language: `cobol`
- External name: `traderb`
- Parameter style: `general`
- WLM environment: `wlmenv1`
- Run options: `'TEST(,,,vaDTCPIP&9.28.194.155%8001:INSPPREF'`
- Program type: `sub`

For DB2 Version 6 and later, to verify that the stored procedure is defined correctly, use the SQL SELECT command on the appropriate DB2 table as follows:

```
select * from sysibm.sysroutines;
```

If the definition is not correct, or if you want to remove the TEST option (we recommend you do not remove it, but change it from TEST to NOTEST), use the appropriate SQL command to modify the stored procedure, as follows:

```
alter procedure traderb run options
'TEST(,,,vaDTCPIP&9.28.194.155%8000:INSPPREF*);'
```

The TEST runtime option used in these examples assumes you will use Remote Debugger. If you want to use VTAM MFI, your TEST runtime option should look as follows:

```
'TEST(,,,MFI%LU0TCP08:INSPPREF*);'
```

You can then call your stored procedure.

15.3 Debug Tool and IMS

This section describes how to compile, link, and debug a program under IMS.

The first part describes the facilities available prior to IMS V8. The second part describes some new facilities available to purchasers of the Debug Tool Utilities and Advanced Features and IMS Version 8.

15.3.1 Traditional methods of debugging under IMS

Traditionally, the hardest part of preparing for debugging of an IMS program or transaction is setting the runtime options.

For compilation, the instructions given in the individual language chapters should be used.

For runtime options, there has traditionally never been a way to specify command line runtime options under IMS. In releases prior to IMS V8, you must use one of the following ways to set the runtime options for a transaction or program:

- An LE CEEUOPT
- The #pragma runopts (for C/C++)
- PLIXOPT for PL/I

MFI single terminal mode is not available under IMS. MFI VTAM or the Remote Debugger should be used.

15.3.2 Debugging with IMS V8 and above

If you have Debug Tool Utilities and Advanced Functions installed on your system and are using IMS version 8, there are some new DTU facilities that allow you to easily change your LE runtime options on the fly. The new V4.1 Debug Tool interface uses the IMS Single Point of Control facilities that are described in the IMS books, but provides an easy-to-use full featured interface tailored towards Debug Tool users.

In the past, the only way to provide the TEST runtime option that is required to bring up Debug Tool was to compile it into the program (with a CEEUOPT or using #pragma runopts with C/C++). This caused problems because:

- All users of the program were affected by this change, not simply the person debugging the program.
- Two people could not debug the same program simultaneously because only one VTAM LUNAME or TCP/IP address could be given.
- The CEEUOPT or program had to be recompiled and relinked every time your TCP/IP address or available VTAM terminal changed.

A second set of features are usable without IMS Version 8 and allow you to Create or Edit setup files for your IMS program using Debug Tool Utilities.

Linking an application to allow the managing of runtime options

IMS uses a customized version of the LE CEEBXITA to allow LE runopts to be specified. This exit is called DFSBXITA and is found in SDFSSMPL. This exit can be used in one of two ways:

1. You can assemble and link it with your application in order to allow dynamic changing of the LE runtime options for that application only.

2. Your system programmer can assemble and then link it into the SCEERUN(CEEBINIT) module. When this SCEERUN is used with your IMS region, it will allow the new functionality to be used with *all* applications running in the region. The JCL shown in Example 15-10 is the assembled DFSBXITA that was in HLQ.MY.OBJECT(CEEBXITA).

Example 15-10 JCL

```
//LKED EXEC PGM=IEWL,REGION=4M,
// PARM='CALL,LIST,XREF,LET,RENT,REFR,MAP,AMODE=24,RMODE=24'
//SYSLIB DD DSN=CEE.SCEELKED,DISP=SHR
// DD DSN=CEE.SCEERUN,DISP=SHR
//MYOBJ DD DSN=HLQ.MY.OBJECT,DISP=SHR
//SYSLMOD DD DSN=<>,DISP=SHR
//SYSUT1 DD DSN=&&SYSUT1,UNIT=SYSDA,SPACE=(CYL,(3,1)),
// DCB=BLKSIZE=1024
//SYSPRINT DD SYSOUT=*
//SYSLIN DD DDNAME=SYSIN
//SYSIN DD *
INCLUDE MYOBJ(CEEBXITA)
REPLACE CEEBXITA
INCLUDE SYSLIB(CEEBINIT)
ORDER CEEBINIT
MODE AMODE(24),RMODE(24)
ENTRY CEEBINIT
NAME CEEBINIT(R)
/*
```

DTU IMS facilities

Option **4** on the Debug Tool Utilities panel shown in Figure 15-29 provides the new IMS debug support facilities as shown in Figure 15-30.

```
---------------------------- Debug Tool Utilities ----------------------------
Option ===> 4_____________________________________________________________________

                                                                   More:     +
0  Manage Job Card
   For Program Preparation and Setup File Management

1  Program Preparation
   Compile old or new COBOL programs with newer compilers, convert old COBOL
   source into new COBOL source, use other compilers, and link edit.

2  Manage and Use Debug Tool Setup Files
   You can manage setup files and use them to run your program interactively
   with Debug Tool in TSO Foreground or submit your program to run in
   the background using MVS batch.

3  Code Coverage
   Measure code coverage in programs written in COBOL, PL/I, C/C++ and
   Assembler when compiled with specific IBM compilers and HLASM.

4  Manage IMS Programs

 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 15-29 Debug Tool Utilities option 4 selection

```
----------------------------- Manage IMS Programs ----------------------------
Option ===> ______________________________________________________________________



1  Manage LE Runtime Options

2  Manage Message Regions









 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 15-30 IMS Panel

Two options are available for customization.

The panel shown in Figure 15-31 requires you to enter the IMS Plex ID. All other fields are optional and can be used to reduce the number of entries on the following panel. Unless you have a large number of entries or know specifically which entry you are looking for, it may not be a good idea to specify any query qualifiers.

This facility allows the specification of runtimeruntime options to be as global as the whole IMS plex, or as specific as a given program in a given transaction from a given terminal submitted by a specific userid. Simply stating the transaction you are looking for may not find all profiles that apply to that transaction.

```
---------------------- Manage LE Runtime Options in IMS ----------------------

Press       to display matched entries.
Press        or      command to exit.

IMSPlex ID  . . . . . PLEX1___   Required

IMSPlex Member. . . . IMS1

Query qualifiers:
  Trancode. . . . . . ________
  Program . . . . . . ________
  Lterm . . . . . . . ________
  Userid. . . . . . . ________





Command ===> ____________________________________________________________
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 15-31 IMS PLEX ID

Figure 15-32 shows the LE runtime options profiles that match the given query parameters. In this example there are three.

```
------------------- Edit LE Runtime Options Entries in IMS --- Row 1 to 3 of 3

IMSPlex ID: PLEX1     IMSPlex Mbr: IMS1

Query qualifiers:
 Trancode:           Program:           Lterm:           Userid:

Use line command
 E to edit an entry       C to copy an entry       D to delete an entry

  Mbr  Trancode Program  Lterm    Userid   LE runtime options
_ IMS1 SDTCB                     HECKE    TEST(ALL,'*',PROMPT,'VADTCPIP&9.146
_ IMS1 SDTCB                              TEST(ALL,'*',PROMPT,'VADTCPIP&9.146
_ IMS1 I5TCB                              TEST(ALL,'*',PROMPT,'TCPIP&9.146.22
****************************** Bottom of data ******************************




Command ===> ___________________________________________________ Scroll ===> PAGE
 F1=Help      F3=Exit      F4=New       F7=Backward  F8=Forward  F10=Refresh
F11=RefreshA F12=Cancel
```

Figure 15-32 LE runtime options profiles

To add a new profile, the F4 key can be used.

The panel shown in Figure 15-33 allows you to provide qualifiers to identify the transaction these runtime options should apply to. This is a long page; press F8 to move forward through

the fields. Do not attempt to put VTAM or remote debugger parameters in the top part of the page. They are collected in the bottom part of the panel.

```
------------------------ Create LE Runtime Options Entry -----------------------
                                                                        More:     +

Provide subsystem information:

IMSPlex ID : PLEX1___            Required
IMSPlex Mbr: IMS1

Qualifiers:
 Trancode: IATCB___  Program: ________  Lterm: ________  Userid: ________

Select Test Options:

 Test Option. . . TEST__         Test/Notest
 Test Level . . . ALL__          All/Error/None
 Commands File                   *, DD name, data set name
  . . . . . . . . *
 Prompt Level . . PROMPT__
 Preference File                 *, DD name, data set name
  . . . . . . . . INSPPREF

Command ===> ____________________________________________________________
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 15-33 LE runtime option entry (part 1 of 2)

Press F8 to move forward and view the information shown in Figure 15-34.

The bottom half of the panel allows you to fill in the parameters to specify where debugging will occur:

- Select Full-screen mode if you have a VTAM Terminal LU that can be used for the MFI debugger.
- Select Remote debug mode if you do not have an available VTAM terminal and you have installed one of the remote debuggers:
  - Set the Type field to WSD if the WebSphere Enterprise Developer Compiled language debugger is in use.
  - Set the Type field to TCP if either the VisualAge Remote Debugger or the IBM Distributed Debugger is in use.

When all the data has been entered, press F3 Exit to create the new profile.

```
----------------------- Create LE Runtime Options Entry -----------------------
                                                                        More:   -
  . . . . . . . . . *
 Prompt Level . . PROMPT
 Preference File                  *, DD name, data set name
  . . . . . . . . . INSPPREF

Select (/) a session type and provide parameters:

 _ Full-screen mode
   Terminal LU. . ________        MFI VTAM Terminal LU

 / Remote debug mode
   Type . . . . . TCP             TCP/WSD
   Address. . . . 9.30.60.129
   Port . . . . . 8001

Other Language Environment run-time options:


Command ===>
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 15-34 Create LE runtime options (part 2 of 2)

Figure 15-35 shows the old and the new profile. The new profile is placed at the top of the list.

```
------------------- Edit LE Runtime Options Entries in IMS --- Row 1 to 4 of 4

IMSPlex ID: PLEX1     IMSPlex Mbr: IMS1

Query qualifiers:
 Trancode:           Program:           Lterm:           Userid:

Use line command
 E to edit an entry       C to copy an entry       D to delete an entry

  Mbr  Trancode Program  Lterm    Userid   LE runtime options
_ IMS1 IATCB                              TEST(ALL,'*',PROMPT,'VADTCPIP&9.30.
_ IMS1 SDTCB                     HECKE    TEST(ALL,'*',PROMPT,'VADTCPIP&9.146
_ IMS1 SDTCB                              TEST(ALL,'*',PROMPT,'VADTCPIP&9.146
_ IMS1 I5TCB                              TEST(ALL,'*',PROMPT,'TCPIP&9.146.22
****************************** Bottom of data ******************************




Command ===>                                                  Scroll ===> PAGE
 F1=Help      F3=Exit      F4=New       F7=Backward  F8=Forward  F10=Refresh
F11=RefreshA F12=Cancel
```

Figure 15-35 Edit LE runtime options

The Edit LE runtime options entries in the IMS menu provide 3 activity choices for each created entry.

- **E** - Edit an entry to make some modification.

  Editing will bring you to the same screens shown in Figure 15-33 and Figure 15-34. You can then make the required modifications.

- **C** - Copy an entry.

Copying an entry will bring you to the same screens as Figure 15-33 and Figure 15-34, where you can make modifications. An additional entry that incorporates your modifications is displayed in the next screen (shown in Figure 15-35).

- **D** Delete an entry that is no longer needed.

  As shown in Figure 15-36, deleting an entry will produce an additional menu on top of the one shown in Figure 15-34.

```
------------------- Edit LE Runtime Options Entries in IMS --- Row 1 to 5 of 5
IM  |            Delete Confirmation for IATCC,,,
    |
Qu  | Instructions:
 T  |    Press ENTER key to confirm delete request.          rid:
    |    (The table entry will be deleted.)
Us  |
 E  |    Enter END or EXIT command to cancel delete          te an entry
    |    request.
    |                                                        s
D   |                                                        PT,'VADTCPIP&9.30.
    |                                                        PT,'VADTCPIP&9.30.
    |                                                        PT,'VADTCPIP&9.146
    |                                                        PT,'VADTCPIP&9.146
    |                                                        PT,'TCPIP&9.146.22
**  |                                                        ******************
    | Command ===> _____________________________
    |  F1=Help      F2=Split     F3=Exit      F7=Backward
    |  F8=Forward   F9=Swap     F12=Cancel
Command ===>                                          Scroll ===> PAGE
 F1=Help      F3=Exit      F4=New       F7=Backward  F8=Forward  F10=Refresh
F11=RefreshA F12=Cancel
```

Figure 15-36 deleting a profile entry

Start the debugger in listening mode on the same port listed previously (8001) and go to IMS to start the transaction. The remote debugger is displayed as shown in Figure 15-37 and you can debug your program.

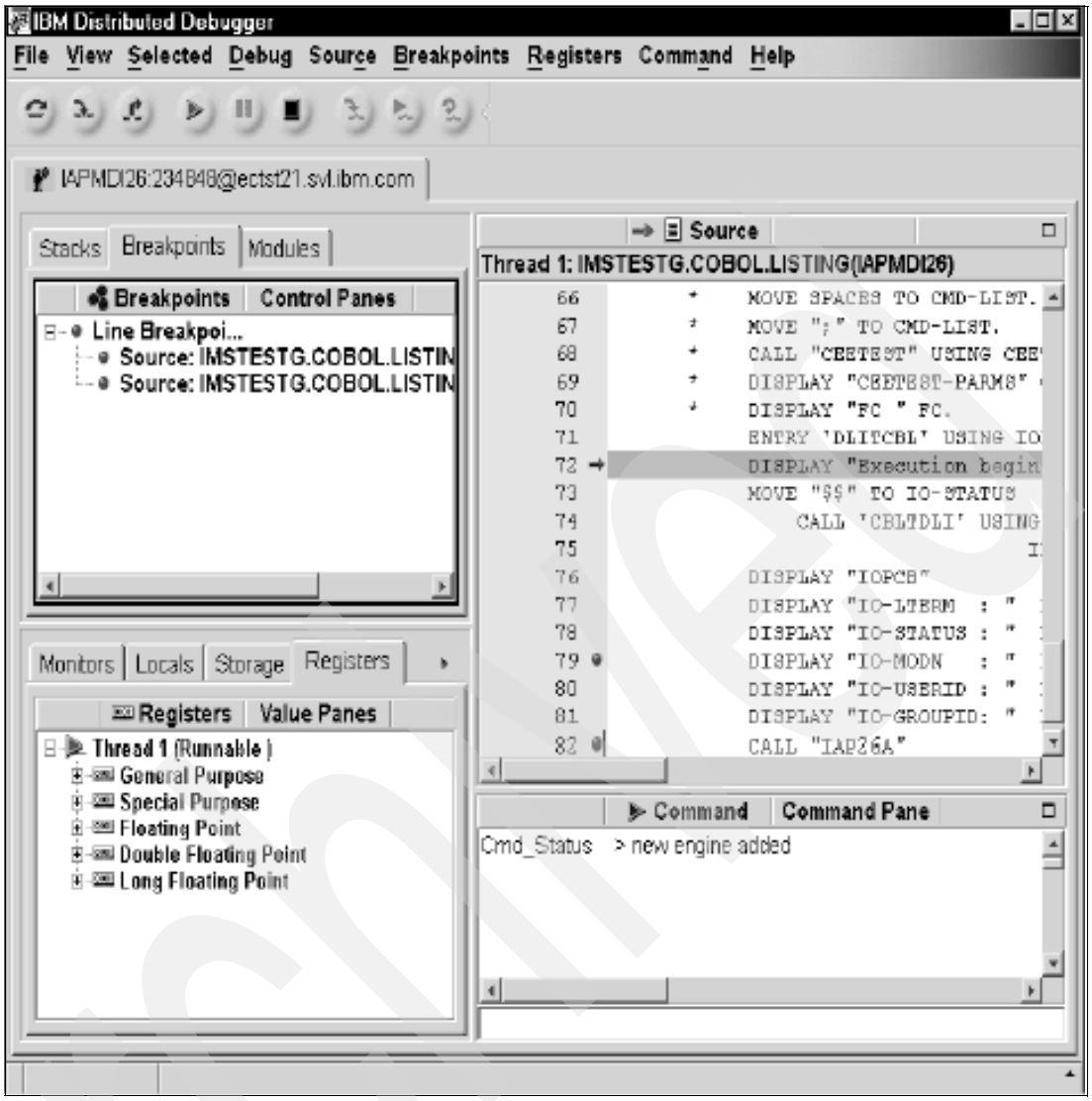


Figure 15-37 Distributed Debugger

If you stop the run by clicking the stop icon (square), you will receive the message shown in Figure 15-38 because you forced abnormal termination of the IMS transaction run.

Figure 15-38 Debugger message

Press F3 twice to go back to the Manage IMS Programs menu as shown in Figure 15-39. At this point, choose option **2 Manage Message Regions**.

```
----------------------------- Manage IMS Programs ----------------------------
Option ===>_2_____________________________________________________________________


1  Manage LE Runtime Options

2  Manage Message Regions


 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 15-39 Manage IMS

Specify your setup file as shown in Figure 15-40 and press Enter. You could optionally specify a new file (including member) here and create a new setup, but this example illustrates how to edit an existing member.

```
------------------ Manage Message Regions - Edit Setup File ------------------
Command ===> ______________________________________________________________________


Setup File Library:
   Project . . . ________
   Group . . . . ________ . . . ________ . . . ________ . . . ________
   Type  . . . . DTSF____
   Member  . . . ________        (Blank or pattern for member selection list)
                                 (or existing or new member name)

Other Data Set Name:
   Data Set Name . . . 'ZHONG.VM3.TEST.DTSF'_____________________________________
   Volume Serial . . . ______    (If not cataloged)




 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
  Menu  RefList  RefMode  Utilities  Help
```

Figure 15-40 Manage message region

This brings up a list of data set members to choose from, as shown in Figure 15-41. In the usual ISPF fashion, they are presented in alphabetical order, and F7 and F8 can be used to move through the list.

```
  Menu  Functions  Confirm  Utilities  Help
 ------------------------------------------------------------------------
LIBRARY          ZHONG.VM3.TEST.DTSF                      Row 00001 of 00087
Command ===> _____________________________________________ Scroll ===> PAGE
   Name     Prompt          Size    Created          Changed            ID
_ A                         108   2002/07/27   2002/07/27 15:55:15    ELIN
_ ABC                         8   2003/06/23   2003/06/23 15:04:32    USRT002
_ ATTRBBMP
_ ATTRBMPP
_ A1                        115   2002/08/01   2002/08/01 10:57:31    ELIN
_ A3                         65   2002/08/02   2002/08/02 13:52:00    ELIN
_ BMP1                        7   2003/02/28   2003/02/28 13:33:11    USRT004
_ BMP11                     194   2003/07/07   2003/07/07 16:55:51    USRT002
_ BMP11A                    194   2003/06/22   2003/06/22 16:32:38    USRT003
_ B1                          5   2002/08/07   2002/08/07 09:56:19    ELIN
_ DB2BAT1                    81   2002/09/24   2002/09/24 23:19:14    ELIN
_ DB2B001                    72   2002/11/27   2002/11/27 14:36:32    ELIN
_ DB2B001B                    0   2002/07/15   2002/07/16 13:23:14    ELIN
_ DB2B002                    67   2002/10/31   2002/11/01 08:40:38    ELIN
_ DB2B002I                   68   2002/10/31   2002/11/03 18:52:54    ELIN
 F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
F10=Left   F11=Right  F12=Cancel
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
  Menu  RefList  RefMode  Utilities  Help
```

Figure 15-41 Dataset member list

An **S** can then be used to select which file to edit and submit as shown in Figure 15-42.

```
  Menu  Functions  Confirm  Utilities  Help
 ------------------------------------------------------------------------
LIBRARY          ZHONG.VM3.TEST.DTSF                      Row 00046 of 00087
Command ===> _____________________________________________ Scroll ===> PAGE
   Name     Prompt          Size    Created          Changed            ID
_ IVP2                       55   2002/09/30   2002/09/30 18:31:32    ELIN
_ IVP3                       49   2002/10/01   2002/10/01 00:09:30    ELIN
_ JJOB                       32   2002/11/23   2002/11/23 16:33:38    ELIN
_ JLIB1                      37   2002/11/25   2002/11/25 13:02:47    ELIN
_ JLIB2                      37   2002/11/25   2002/11/25 14:25:15    ELIN
_ MPPCA1                      8   2003/08/29   2003/08/29 11:13:12    USRT010
_ MPPC1A                    212   2003/09/18   2003/09/18 14:28:03    USRT010
_ MPPC1B                    197   2003/09/02   2003/09/02 17:32:58    USRT010
S MPPC3A                    173   2003/10/15   2003/10/15 11:31:59    USRT010
_ MPPC3B                    172   2003/04/23   2003/04/23 16:26:35    USRT003
_ MPP166                    174   2003/03/14   2003/03/21 14:44:14    ELIN1
_ MPP166A                   172   2003/03/24   2003/04/15 13:16:26    USRT003
_ TC1                        79   2002/08/08   2002/08/08 14:41:31    ELIN
_ TC2                        64   2002/08/23   2002/08/23 12:47:22    ELIN
_ TC3                        65   2002/08/23   2002/08/23 12:48:42    ELIN
 F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
F10=Left   F11=Right  F12=Cancel
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
  Menu  RefList  RefMode  Utilities  Help
```

Figure 15-42 Selection

A **/** (slash) on the specified line will allow you to modify the IMS parameters as shown in Figure 15-43.

```
Edit - Edit Setup File 'ZHONG.VM3.TEST.DTSF(MPPC3A)'          Row 1 to 5 of 35
Command ===> ____________________________________________ Scroll ===> PAGE

Modify information and use the  Submit command to submit to Batch.
     Press HELP for a list of all available commands.

Batch Job Name. . . MPPC3A

Load Module Name. . DFSRRC00

IMS Subsystem ID. . IMS1

/ Enter / to modify parameters (MSG,00300300300 3,V00000040,,0,,,05,05,IMS1,,,,
RR,,,,)

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)          DISP
___ _______  ___ _ ***************** Top of Data ******************* ____
___ COMMDD   1   _ 'IMSTESTL.IMS.COMMDD'                            SHR
___ IMSERR   1   _ SYSOUT=A                                         ____
___ PRINTDD  1   _ SYSOUT=A                                         ____
___ PROCLIB  1   _ 'IMSVS.PROCLIB'                                  SHR
 F1-Help      F3-Exit      F7-Backward  F8-Forward  F10-Submit   F12-Cancel
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
  Menu  RefList  RefMode  Utilities  Help
```

Figure 15-43 Selection

The screen shown in Figure 15-44 pops up to make it easy to modify IMS Procedure parameters. You simply update the fields and DTU will generate the appropriate parameter string once you exit the page.

```
------------------------ Parameters for IMS Procedures ----------------------
Command ===> ____________________________________________________________________

TYPE. . . . . MSG       MSG or BMP


AGN . . . . . ________  ALTID . . . . ____
APARM . . . . ________________________________  CKPTID. . . . ___
CL1   . . . . 003       CL2 . . . . . 003      CL3 . . . . . 003
CL4   . . . . 003       CPUTIME . . . ____     DBLDL . . . . ____
DIRCA . . . . ___       IMSID . . . . IMS1     IN. . . . . . ____
LOCKMAX . . . _____     MBR . . . . . ________ NBA . . . . . 05
OBA . . . . . 05        OPT . . . . . W        OUT . . . . . ___
OVLA. . . . . 0         PARDLI. . . . _        PCB . . . . . 040
PREINIT . . . RR        PRLD. . . . . __       PSB . . . . . ________
SOD . . . . . _         SPIE. . . . . 0        SSM . . . . . ____
STIMER. . . . 0         TEST. . . . . _        TLIN. . . . . 00
VALCK . . . . 0         VFREE . . . . _        VSFX. . . . . __
WFI . . . . . ________

 F1-Help      F2-Split     F3-Exit      F7-Backward  F8-Forward   F9-Swap
F12=Cancel
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
  Menu  RefList  RefMode  Utilities  Help
```

Figure 15-44 Parameters for IMS procedures

The forward and backward keys can be used to move through the DDs. An asterisk (*) in the C column will comment out the line. Once you have made all the changes you wish to the DD information, press F10 to submit as shown in Figure 15-45.

```
Edit - Edit Setup File 'ZHONG.VM3.TEST.DTSF(MPPC3A)'        Row 11 to 15 of 35
Command ===> ___________________________________________ Scroll ===> PAGE

Modify information and use the  Submit command to submit to Batch.
     Press HELP for a list of all available commands.

Batch Job Name. . . MPPC3A __

Load Module Name. . DFSRRC00

IMS Subsystem ID. . IMS1

_ Enter / to modify parameters (MSG,003003003003,W00000040,,0,,,05,05,IMS1,,,,,
RR,,,,)

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)            DISP
___ ________ 3   _ 'VCATDGL'                                          SHR
___ STEPLIB  1   _ 'IMSTESTL.TNUC1'                                   SHR
___ ________ 2   _ 'IMSTESTG.V2R10M0.SCEERUN'                         SHR
___ ________ 3   * 'ELIN.VM3.SL3060.SEQAMOD'                          SHR
___ ________ 4   _ 'IMSTESTL.TNUC0'                                   SHR
 F1=Help      F3=Exit      F7=Backward  F8=Forward  F10=Submit   F12=Cancel
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
  Menu  RefList  RefMode  Utilities  Help
```

Figure 15-45 Edit

The submit function brings you into an edit statement where you can do a final review of the JCL and type **sub** to submit it, as shown in Figure 15-46 and Figure 15-47.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------------
VIEW       USRT010.DTSUTEMP.XJC1D043                        Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> PAGE
****** **************************** Top of Data ******************************
==MSG> -Warning- The UNDO command is not available until you change
==MSG>           your edit profile using the command RECOVERY ON.
=NOTE= Enter Submit to submit the JCL.
=NOTE= Press PF3 to exit.
=NOTE= -Warning- Changes will NOT be saved.
-NOTE-
000001 //MPPC3A     JOB 'IMS SYSTEM',CLASS-J,MSGCLASS-A,MSGLEVEL-(1,1),
000002 //         TIME-1440,REGION-0M
000003 //*
000004 //*
000005 //* Generated by DTSU
000006 //*
000007 //MPP EXEC PGM=DFSRRC00,
000008 // PARM='MSG,003003003003,W00000040,,0,,,05,05,IMS1,,,,,RR,,,,'
000009 //COMMDD DD DSN=IMSTESTL.IMS.COMMDD,DISP=(SHR,,)
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
  Menu  RefList  RefMode  Utilities  Help
```

Figure 15-46 Edit

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------------
VIEW       USRT010.DTSUTEMP.XJC1D043                        Columns 00001 00072
Command ===> sub__________________________________________ Scroll ===> PAGE
****** **************************** Top of Data ******************************
==MSG> -Warning- The UNDO command is not available until you change
--MSG>           your edit profile using the command RECOVERY ON.
=NOTE= Enter Submit to submit the JCL.
=NOTE= Press PF3 to exit.
=NOTE= -Warning- Changes will NOT be saved.
=NOTE=
000001 //MPPC3A     JOB 'IMS SYSTEM',CLASS=J,MSGCLASS=A,MSGLEVEL=(1,1),
000002 //         TIME=1440,REGION=0M
000003 //*
000004 //*
000005 //* Generated by DTSU
000006 //*
000007 //MPP EXEC PGM=DFSRRC00,
000008 // PARM='MSG,003003003003,W00000040,,0,,,05,05,IMS1,,,,,RR,,,,'
000009 //COMMDD DD DSN=IMSTESTL.IMS.COMMDD,DISP=(SHR,,)
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
  Menu  RefList  RefMode  Utilities  Help
```

Figure 15-47 Edit

Once the job has been processed, its status is updated in the list window as shown in Figure 15-48.

```
  Menu  Functions  Confirm  Utilities  Help
 ------------------------------------------------------------------------
LIBRARY          ZHONG.VM3.TEST.DTSF                      Row 00046 of 00087
Command ---> _                                             Scroll ---> PAGE
   Name     Prompt          Size    Created          Changed            ID
_ IVP2                       55   2002/09/30   2002/09/30 18:31:32    ELIN
_ IVP3                       49   2002/10/01   2002/10/01 00:09:30    ELIN
_ JJOB                       32   2002/11/23   2002/11/23 16:33:38    ELIN
_ JLIB1                      37   2002/11/25   2002/11/25 13:02:47    ELIN
_ JLIB2                      37   2002/11/25   2002/11/25 14:25:15    ELIN
_ MPPCA1                      8   2003/08/29   2003/08/29 11:13:12    USRT010
_ MPPC1A                    212   2003/09/18   2003/09/18 14:28:03    USRT010
_ MPPC1B                    197   2003/09/02   2003/09/02 17:32:58    USRT010
_ MPPC3A   *PROC'ED         173   2003/10/15   2003/10/15 11:31:59    USRT010
_ MPPC3B                    172   2003/04/23   2003/04/23 16:26:35    USRT003
_ MPP166                    174   2003/03/14   2003/03/21 14:44:14    ELIN1
_ MPP166A                   172   2003/03/24   2003/04/15 13:16:26    USRT003
_ TC1                        79   2002/08/08   2002/08/08 14:41:31    ELIN
_ TC2                        64   2002/08/23   2002/08/23 12:47:22    ELIN
_ TC3                        65   2002/08/23   2002/08/23 12:48:42    ELIN
 F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up     F8=Down    F9=Swap
F10=Left   F11=Right  F12=Cancel
 -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -
  Menu  RefList  RefMode  Utilities  Help
```

Figure 15-48 Updated list

15.3.3 Precedence of the information on the LE Runtime Options panel

When using the new DTU Language Environment (LE) runtime options support, two obvious questions arise.

If my program matches more than one profile, which one will be used?

The first match located is used. The list is searched in the order profiles are entered. Editing the entry does not change its position in the order.

Based on this, we recommend that this feature be used as little as possible for specifying runtime options other than TEST, and that all users on a given IMS plex have agreed upon a way of setting up the profiles (perhaps requiring the userid, or always stating the transaction). This facility is best used to simply specify the TEST runtime option. It should only be used to specify other runtime options as a limited time test of the given LE runtime options efficiency.

Where do these runopts fit in the LE runopt hierarchy?

This questions concerns which runopts have precedence (panel, ceeuopt, ceeropt, ceedopt).

Language Environment has several ways of specifying the runtime options for a program. The following list gives them in ascending order of precedence (that is, things lower on the list can override things higher on the list):

- CEEDOPT — Language Environment options specified at installation
- CEEROPT — region-wide CICS or IMS default options
- CLER under CICS
- CEEUOPT — also #pragma runopts, or PLIXOPTS

- Command Line, or _CEE_RUNOPTS
- CEEBXITA
- Language Environment Storage Tuning User Exit
- Options defined at installation time that have the nonoverridable attribute.

The usual Language Environment runtime option merging will occur (that is, options or suboptions from a lower precedence entity will be taken).

16

Debug Tool Utilities and Advanced Functions

This chapter describes the code-coverage utility of Debug Tool Utilities and Advanced Functions as well as the COBOL conversion and compilation functionality.

The code-coverage utility enables you to determine which code statements have been executed and which statements have not been executed. You can use this information to decide whether to adapt the application's input data and parameters to get full test coverage.

Debug Tool Coverage Utility (DTCU) supports all environments except UNIX System Services. In this part, we emphasize measuring code coverage in Enterprise COBOL for z/OS and OS/390.

COBOL conversion and compilation functionality also is available with the Debug Tool Utilities and Advanced Functions. This facility allows you to convert and compile older COBOL programs such as:

- Programs written for the OS/VS COBOL compiler (68/74 Standard COBOL).
- Programs written for the VS COBOL II (or later) compiler that was last compiled with the CMPR2 compiler option (74/85 Standard).

The conversion uses the COBOL and CICS Command Level Conversion Aid (CCCA) to perform the conversion, and creates code that is suitable for compiling with:

- 5648-G53 ENTERPRISE COBOL for z/OS
- 5648-A25 COBOL for OS/390 and VM

After compilation, you can debug your converted old COBOL source code with the Debug Tool using the object module created from the newer compiler.

16.1 Coverage Utility

To effectively use DTCU (Debug Tool Coverage Utility), you must customize it first.

16.1.1 DTCU customization

This section describes:

- Where to place certain Coverage Utility load modules so that the correct people have access to them
- How to install the monitor SVCs
- How to customize the product

Make the load modules in hlq.SEQAMOD accessible to all users

Certain Coverage Utility load modules must be placed in an APF-authorized data set that is accessible to all your users. The APF-authorized data set does not need to be in the link list. Perform the following steps:

1. Mark the hlq.SEQAMOD data set as APF-authorized and make it accessible to all users, or copy the following load modules to an APF-authorized data set that all users can access:
   - EQACUOCM (monitor interface)
   - EQACU9M0 (monitor messages)
   - EQACU9M1 (monitor messages)
   - EQACU9M2 (monitor messages, and only if the JPN feature is installed)
2. Add the EQACUOCM program to the AUTHPGM entry in the member IKJTSOxx of the SYS1.PARMLIB data set.
3. Issue the PARMLIB UPDATE(xx) command from TSO or IPL your system.
4. Edit the EQASTART member of the hlq.SEQAEXEC data set and set the INST_Auth_SEQAMOD variable to the name of the data set from step 1 that contains EQACUOCM.

Placing load modules in an APF data set not accessible to general users

Certain Coverage Utility load modules must be placed in an APF-authorized data set that is accessible only to system programmers. The APF-authorized data set must not be in the link list. Do one of the following:

- Mark the hlq.SEQAAUTH data set as APF-authorized and limit access to these load module.
- Do not mark the hlq.SEQAAUTH data set as APF-authorized. Copy the following load modules into an APF-authorized data set that only system programmers can access:
  - EQACUOIN (SVC installer)
  - EQACUOSV (SVCs)

Installing and enabling the monitor SVCs

The EQACUOIN module installs and enables the monitor SVCs. The monitor SVCs must be installed and enabled before a user starts a monitor session. The EQACUOIN module must be run:

- When the SVCs are initially installed
- After service is applied
- Any time you IPL your system

The monitor SVCs use some common system storage, as described later. In addition, each user session uses ECSA storage.

- **CSA** 13248 bytes
- **SQA** 25496 bytes

To perform the initial installation:

1. Reserve two free user SVC numbers. IEASVCxx does not need to be updated since these user SVCs are installed dynamically.
2. Customize the EQACUOIS member of the hlq.SEQASAMP data set.
3. Submit EQACUOIS on the system on which you intend to run the monitor.
4. To verify that the monitor was installed properly, run the following command from ISPF 6:

```
ex 'hlq.SEQAEXEC(EQACUOSE)' 'LEVEL'
```

Verify that the monitor release is V5R1M0 and the date is 2002.245 or later as shown in Example 16-1.

Example 16-1 EQACUOSE reply

```
********************************** Top of Data **********************************
Monitor Release: V5R1M0 Date: 2002.245
MAST: 00F9A620 PSA: 00F1C000 CPU: 00000000 SEST: 00F9A308 UNID: 00000000
******************************** Bottom of Data *********************************
```

To have the monitor SVCs installed automatically after a system IPL, make the following changes:

1. Add COM='S EQACUOIN' to the COMMNDxx member of the SYS1.PARMLIB data set.
2. Copy and customize hlq.SEQASAMP(EQACUOPS) to your SYS1.PROCLIB data set as member EQACUOIN.
3. The process started by EQACUOIN must have access to the EQACUOIN and EQACUOSV load modules.

Customizing the product defaults

Complete the following steps to edit hlq.SEQAEXEC(EQACUDFT):

1. Change all occurrences of EQAW to hlq.
2. In the execute step data entry for EXEJOBLB, change the name hlq.SEQAMOD to the name of the APF-authorized data set you used for the EQACUOCM program.
3. Enter the Coverage Utility Monitor SVC numbers (in hexadecimal notation) in the CUSVC2B and CUSVC4B entries.
4. When you create JCL, the *JOBLn lines become the first three lines of the JOB card for each respective job. Customize these lines.
5. If your site requires a specification for allocation parameters such as STORCLAS or UNIT on new or temporary data set allocations, look for the word SPACE in this EXEC and the 'hlq.SEQAS*' data sets and update the allocation specifications.
6. If you want Coverage Utility to generate or build each data set as sequential or partitioned, set the USEPRGNM variable to Y. To generate a data set as sequential, set the DSORG variable to SEQ. To generate a data set as partitioned, set the DSORG variable to PDS.
7. If you do not want Coverage Utility to generate or build any data set names automatically, set the USEPRGNM variable to N.

16.1.2 DTCU requirements

For coverage analysis, DTCU requires:

- The listings created by the compilers and the assemblers. These products offer options that enable you to include assembler statements in the listings. These assembler statements in the listing are needed by DTCU.
- The object module or load modules. DTCU will create copies with inserted breakpoints.

16.1.3 Running DTCU

To run DTCU, performing the following steps:

1. Setup:
   a. Compile the source code, using required compile or assembler options.
   b. Generate DTCU JCL using the DTCU ISPF dialog:
      - Edit the DTCU control file.
      - Create the setup JCL.
      - Create the start monitor JCL.
      - Create the report or summary JCL.
   c. Edit the link edit JCL to include the modified object.
   d. Edit the program invocation JCL to point to the modified load module.
2. Execution:
   a. Run the setup JCL.
   b. Run the link edit JCL.
   c. Run the JCL to start a monitor session.
   d. Run your application using the modified load modules.
   e. Stop the monitor session.
3. Report:
   a. Run the report or summary JCL.

Figure 16-1 shows a diagram of the entire process.

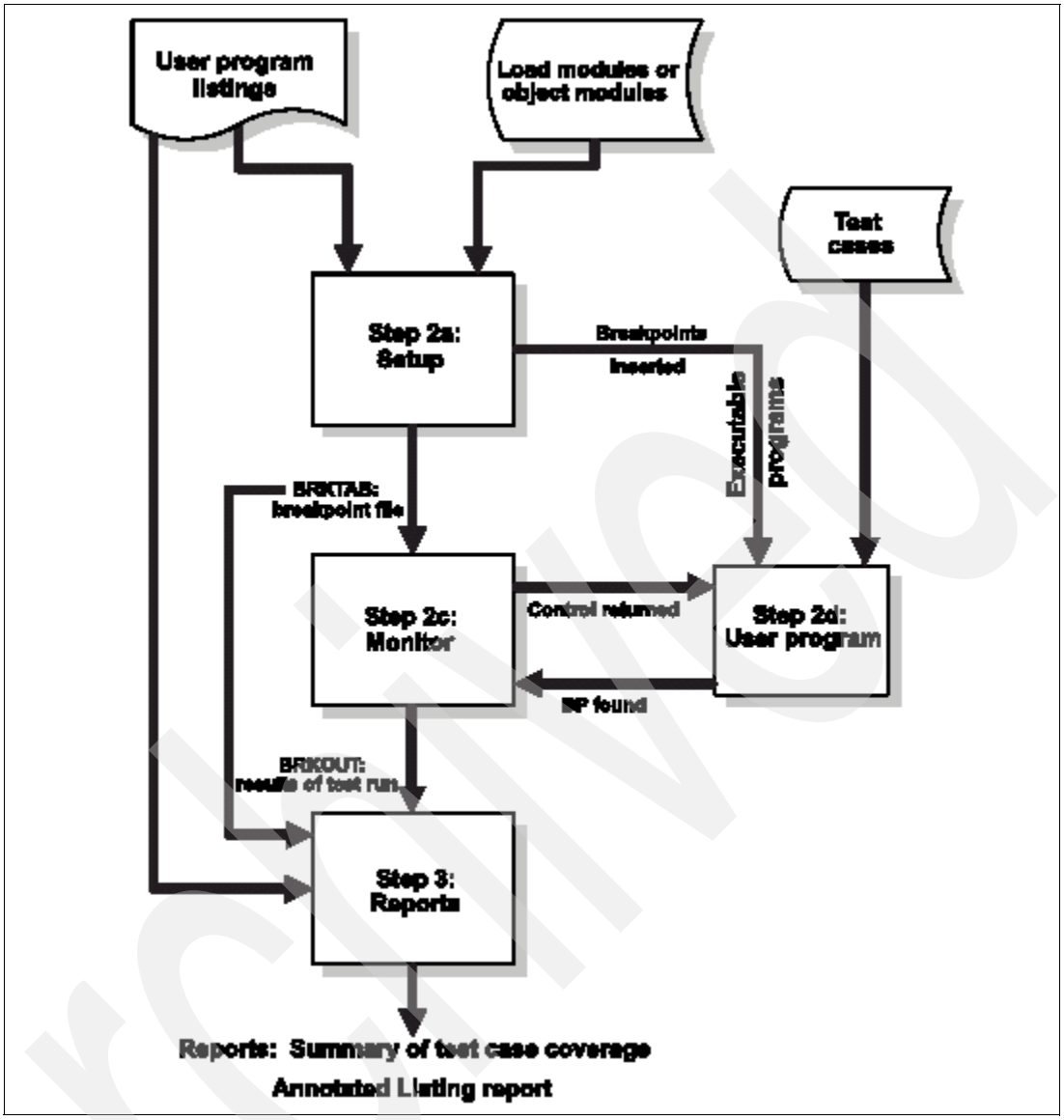


Figure 16-1 DTCU flow

16.1.4 Using DCTU

DTCU comes with an ISPF user interface where you can start all preparation steps needed, as follows:

1. Set up the DTCU defaults.
2. Set up your control file.
3. Create the JCL for the setup.
4. Create the JCL to start the monitor.
5. Create JCL for a report.

Starting the DTCU ISPF dialog

If an option was installed to access the Debug Tool Utilities from an existing ISPF panel, then select that option.

If the Debug Tool Utilities were not installed in your ISPF environment, enter the following command from ISPF option **6**:

```
EXEC 'hlq.SEQAEXEC(EQASTART)'
```

Next, select option **3** for DTCU. The next panel you see is the Debug Tool Coverage Utility, as shown in Figure 16-2.

```
------------------------- Debug Tool Coverage Utility -------------------------
Option ===> _____________________________________________________________________

0  Defaults      Manipulate defaults
1  CntlFile      Work with the Control File
2  Setup         Create JCL for Setup
3  StartMon      Create JCL to Start the Monitor
4  Reports       Create JCL for Reports
5  Monitor       Control the Monitor
6  FastPath      FastPath

Enter X to Terminate

                                                                        02/014
```

Figure 16-2 DCTU: Primary panel

16.1.5 Modifying your DTCU defaults

Select option **0 Defaults** from the Debug Tool Coverage Utility panel to specify your Coverage Utility user default. The Manipulate Defaults panel is displayed as shown in Figure 16-3.

```
----------------------------- Manipulate Defaults ------------ Variables saved
Option ===> _____________________________________________________________________

1  EDIT          Edit defaults
2  RESET         Reset defaults to site defaults
3  IMPORT        Import defaults from a sequential dataset
4  EXPORT        Export defaults to a sequential dataset

Enter END to Terminate

Import | Export Dataset (Options 3 and 4 only):
  Data Set Name . . . . ________________________________________________

                                                                        02/014
```

Figure 16-3 DTCU - Manipulate Defaults

To edit your user defaults perform these steps:

1. Select option **1** on the Manipulate Defaults panel. The scrollable Edit Defaults panel is displayed as shown in Figure 16-4 (use F7 and F8 to navigate).

```
-------------------------------- Edit Defaults --------------------------------
Command ===> ____________________________________________________________________

Enter END (to Exit and Save changes) or CANCEL (to Exit without saving)
                                                                   More:     +
------------------------------- General Defaults ------------------------------
Project Qualifier. . . . . CHABERT.COVERAGE
Use Pgm Name for File Name YES        (Yes|No)
Program Name . . . . . . . TRADERB
JCL Output Dsn . . . . . . 'CHABERT.COVERAGE.JCL'
  Type . . . . . . . . . . JCL
  DSORG. . . . . . . . . . PDS        (SEQ|PDS)
  Alloc Parms. . . . . . . LRECL(80) RECFM(F B) BLKSIZE(0)
                           TRACKS SPACE(10 10) UNIT(SYSALLDA)
1st JOBLIB Dsn . . . . . . 'ADTOOLS.DEBUG.TOOL51.SEQAMOD'
2nd Alternate JOBLIB Dsn . _
3rd Alternate JOBLIB Dsn .
4th Alternate JOBLIB Dsn .
5th Alternate JOBLIB Dsn .
6th Alternate JOBLIB Dsn .
REXX Dsn . . . . . . . . . 'ADTOOLS.DEBUG.TOOL51.SEQAEXEC'
Sample Dsn . . . . . . . . 'ADTOOLS.DEBUG.TOOL51.SEQASAMP'
Display Messages . . . . . I          (S|E|W|R|I)
Log Messages . . . . . . . I          (S|E|W|R|I)
                                                                     16/029
```

Figure 16-4 DTCU - Edit Defaults

2. Change the Project Qualifier value to the high-level qualifier that you want Coverage Utility to use to construct names for user and project data sets.
3. To generate or build any data set names automatically, ensure that Use Pgm Name For File Name is set to Yes. Coverage Utility uses the project qualifier, the program name, and the specified values for type and DSORG for each data set to build names of the following forms:

   Sequential data sets:

   'proj_qual.program_name.file_type'

   – Partitioned data sets:

   'proj_qual.file_type(program_name)'

   If you specify No for Use Pgm Name For File Name, Coverage Utility does not automatically build or change any data set names.

Once done, you can save (Export) your defaults. Later, you can either reset the defaults to your site defaults, or you can import the defaults that you have exported.

16.1.6 Compiling the program to be measured

To compile your source code, do these steps:

1. Edit your compile JCL.
2. Run the JCL.
3. Make sure that all steps run with a return code of zero and that the listings and object modules are created.

We used our program TRADERB, and made sure that the compile time option LIST was on. This option is available for all compilers. For the required compile time options, refer to

Coverage Utility User's Guide and Messages V5R1, SC18-9303. The SYSLIN output from the compiler, as well as the compiler listings, are now available as members of partitioned data sets.

Example 16-2 is a sample for the compile JCL we used.

Example 16-2 Sample JCL for code coverage

```
//COMPILE  EXEC PGM=IGYCRCTL,
//             PARM='NOTEST,SOURCE,LIST,OBJECT,NOOPT,NONUMBER,LIB'
//STEPLIB   DD DSN=IGY.V3R4M0.SIGYCOMP,DISP=SHR
//SYSIN     DD DSN=CHABERT.TRADER.SOURCE(TRADERB),DISP=SHR
//SYSPRINT  DD DSN=CHABERT.TRADER.COBLIST(TRADERB),DISP=SHR
//SYSLIN    DD DSN=CHABERT.TRADER.COBOBJ(TRADERB),DISP=SHR
//SYSUT1    DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSUT2    DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSUT3    DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSUT4    DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSUT5    DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSUT6    DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSUT7    DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSLIB    DD DSN=CHABERT.TRADER.COPYLIB,DISP=SHR
```

Note: For the compiler options required by Coverage Utility, refer to the *Debug Tool Version 5 Release 1 Coverage Utility User's Guide and Messages* SC18-9303, chapter 6.

16.1.7 Setting up your control file

Coverage Utility uses assembler statements from the compiler listings to determine where to insert breakpoints. You supply the names of the listing files in the Coverage Utility control file.

Select **1 CntlFile** from the main DTCU panel. The control file data set name will be based on the defaults you have defined. In the control file, describe the compile units you want to be analyzed.

In our test case, the control file name is 'userid.COVERAGE.DTCU(TRADERB)'

Select **1: Edit**

The data in the control file consists of the following information:

- The type of listing file (COBOL).
- The names of the listing files for the programs that you want to test.
- The names of the load modules that contain the code of each listing.
- The copy to or from information for making copies of the object modules into which the breakpoints are inserted.

We wanted to make this discussion easy to follow, so our TRADERB is really very simple, as shown in Figure 16-5.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------------
EDIT       CHABERT.COVERAGE.DTCU(TRADERB) - 01.05          Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> PAGE
****** ***************************** Top of Data ******************************
==MSG> -Warning- The UNDO command is not available until you change
==MSG>           your edit profile using the command RECOVERY ON.
000001 *
000002 * This is an example DTCU control file that was copied from
000003 * your system's SEQASAMP library. Make appropriate edits for your testin
000004 *
000005 * COBOL Example
000006 *
000007 * Statements required for coverage
000008 *
000009         Defaults ListDSN=CHABERT.TRADER.COBLIST(*),
000010                  LoadMod=TRADERB,
000011                  FromObjDSN=CHABERT.TRADER.COBOBJ,
000012                  ToObjDSN=CHABERT.TRADER.DTCUOBJ
000013
000014         COBOL ListMember=TRADERB
****** ***************************** Bottom of Data ***************************


                                                                        04/015
```

Figure 16-5 DTCU Control File

The last line `COBOL ListMember=TRADERB` defines what COBOL program will be prepared and measured. If you have an application with several Compile units, add a line for each of those compile units you want to be analyzed.

Example:

COBOL ListMember=TRADERB

PLI Listmember=TRADPLI

16.1.8 Create the setup JCL

When you run the setup JCL, the Coverage Utility setup program analyzes the assembler statements in the compiler listings and creates a table that contains breakpoint data (address, op code, and so on). Breakpoints are inserted in the instrumented object modules or load modules. If you instrumented object modules, you then link these modified object modules into a modified sample load module for Coverage Utility to use.

Select option **2 (Setup**) from the main DTCU panel and Figure 16-6 is displayed.

The generated JCL will be used to create a new object, which will have breakpoints inserted based on the assembler code (because of the LIST compile time option) found in the compiler listings.

```
----------------------------- Create JCL for Setup -----------------------------
Option ===> _____________________________________________________________________

1  Generate      Generate JCL from parameters
2  Edit          Edit JCL
3  Submit        Submit JCL

Enter END to Terminate


Use Program Name for File Name  YES (Yes|No) Program Name  TRADERB

Control File:
  Control File Dsn. . . 'CHABERT.COVERAGE.DTCU(TRADERB)'

JCL Library and Member:
  JCL Dsn . . . . . . . 'CHABERT.COVERAGE.JCL(STRADERB)'

Output Breakpoint Table:
  Breakpoint Table Dsn. 'CHABERT.COVERAGE.TRADERB.BRKTAB'



                                                                        02/014
```

Figure 16-6 DTCU - Create JCL for Setup

Select option **1 Generate.**

Informational messages are written to your screen as the JCL is created. The created JCL is put into the JCL library that is identified on the panel by the member name S*progname.*

16.1.9 Create JCL to start a monitor session

JCL is required to start a Coverage Utility monitor session.

To create the JCL to start a monitor session, do these steps:

1. Select option **3 StartMon** from the Debug Tool Coverage Utility panel and the Create JCL to Start the Monitor panel is displayed as shown in Figure 16-7.
2. If necessary, change the program name.
3. Select option **1 Generate.** Informational messages are written to your screen as the JCL is created. The created JCL is put into the JCL library that is identified on the panel by the member name X*progname.*

```
----------------------- Create JCL to Start the Monitor -----------------------
Option ===> _____________________________________________________________________

1  Generate      Generate JCL from parameters
2  Edit          Edit JCL
3  Submit        Submit JCL

Enter END to Terminate


Use Program Name for File Name  YES (Yes|No) Program Name  TRADERB

Session ID  . . . . . . CHABERT

Input File:
  Breakpoint Table Dsn. 'CHABERT.COVERAGE.TRADERB.BRKTAB'

JCL Library and Member:
  JCL Dsn . . . . . . . 'CHABERT.COVERAGE.JCL(XTRADERB)'

Output File:
  Breakout Dsn. . . . . 'CHABERT.COVERAGE.TRADERB.BRKOUT'


                                                                        02/014
```

Figure 16-7 DCTU - Create JCL for Monitor

16.1.10 Create JCL for a report

JCL is required to generate a summary report.

To create the summary report JCL:

1. Select option **4 Reports** from the Debug Tool Coverage Utility panel. The Create JCL for Reports panel is displayed.
2. Select option **1 Summary**. The Create JCL for Summary Report panel is displayed as shown Figure 16-8. You create the JCL for generating the sample summary report from this panel.

   You also can use option **2 Annotation** to create JCL for summary and annotation reports.

```
------------------------- Create JCL for Summary Report -------------------------
Option ===> _____________________________________________________________________

1  Generate      Generate JCL from parameters
2  Edit          Edit JCL
3  Submit        Submit JCL

Enter END to Terminate

Use Program Name for File Name  YES (Yes|No) Program Name  TRADERB

Input Files:
  Breakpoint Table Dsn. 'CHABERT.COVERAGE.TRADERB.BRKTAB'
  Breakout Dsn. . . . . 'CHABERT.COVERAGE.TRADERB.BRKOUT'

JCL Library and Member:
  JCL Dsn . . . . . . . 'CHABERT.COVERAGE.JCL(TTRADERB)'

Output Summary Type and File:
  Type. . . . . . . . . INTERNAL (Internal|External)
  Inline  . . . . . . . N        (I|N)
  Report Dsn  . . . . . 'CHABERT.COVERAGE.TRADERB.SUMMARY'
    (* for default sysout class)

                                                                          02/014
```

Figure 16-8 DTCU - Create a JCL for Summary report

3. Select option **1 Generate**. Informational messages are written to your screen as the JCL is created. The created JCL is put into the JCL library that is identified on the panel by the member name T*progname*.

16.1.11 Jobs generated so far

After you have generated the JCL, there are three jobs in your JCL data set as shown in Figure 16-9.

```
  Menu  Functions  Confirm  Utilities  Help
 ____________________________________________________________________________
EDIT              CHABERT.COVERAGE.JCL                     Row 00001 of 00003
          Name     Prompt       Size   Created          Changed          ID
_________ STRADERB
_________ TTRADERB
_________ XTRADERB
          **End**

Command ===> ____________________________________________ Scroll ===> CSR
                                                                        24/015
```

Figure 16-9 DTCU - Jobs after JCL generation

The three jobs are named with your program name prefixed by:

– S: JCL for the setup step

– T: JCL for the Summary report
– X: JCL to start up the monitor session
– R: JCL for the Create JCL for Summary and Annotation Report

Of course, you could add the JCL for the compile of your modules, and a job to link edit the modified object. For both, there are samples available in the hlq.SEQASAMP.

Next we start the coverage measurement process.

16.1.12 Perform the setup step

Submit the JCL (Sprogname) generated by the Create JCL for Setup step.

The job should end with RC=0. After that you will find two new data sets. In our sample they were called:

- CHABERT.COVERAGE.TRADERB.BRKTAB (a file of breakpoint-related information that can be reused later when the monitor is started and during the report printing).
- CHABERT.TRADER.DTCUOBJ (the modified object, which will be linked as a regular application)

16.1.13 Perform the link step

This is a regular link edit job as shown in Example 16-3.

Example 16-3 Sample Link JCL

```
//LINK     EXEC PGM=HEWL,PARM='MAP'
//SYSLIB    DD DSN=CEE.SCEELKED,DISP=SHR
//SYSPRINT  DD DSN=CHABERT.TRADER.LINKLIST(TRADERB),DISP=SHR
//SYSLIN    DD DSN=CHABERT.TRADER.DTCUOBJ(TRADERB),DISP=SHR
//SYSLMOD   DD DSN=CHABERT.TRADER.LOAD(TRADERB),DISP=SHR
//SYSUT1    DD UNIT=SYSDA,SPACE=(TRK,(10,10))
//*
```

After the link, you have a regular load module ready to be executed. Before you do that, you must start the monitor session.

16.1.14 Start the monitor session

For this, you submit the JCL (Xprogname) generated by the Create JCL to start a monitor session step.

16.1.15 Start your application

Although there is some JCL provided in the SEQASAMP (member EQACU1G9), you can use your original JCL. Eventually, the program will end with RC, so now you can look for the output.

16.1.16 Display statistics online

This is an optional step.

From ISPF option 6, run the command EXEC hlq.SEQAEXEC(EQACUOSA)

You will get some output depending on the number of source lines. This is only recommended for small test cases. An example is shown in Figure 16-10.

```
  Menu  Utilities  Compilers  Help
 ------------------------------------------------------------------------------
 BROWSE    CHABERT.MSGS.FILE                          Line 00000000 Col 001 080
********************************* Top of Data ***********************************
Num Listing                                     Date   Time     PAs   BPs
--------------------------------------------------------------------------------
001 CHABERT.TRADER.COBLIST(TRADERB)             04.326 13:29.59 00039 000422
PA    ADR      BPS    EVNTS      ACTVE
--------------------------------------------------------------------------------
00001 00000000 000019 0000000000 000019
00002 00000000 000012 0000000000 000012
00003 00000000 000001 0000000000 000001
00004 00000000 000023 0000000000 000023
00005 00000000 000030 0000000000 000030
00006 00000000 000001 0000000000 000001
00007 00000000 000028 0000000000 000028
00008 00000000 000001 0000000000 000001
00009 00000000 000034 0000000000 000034
00010 00000000 000001 0000000000 000001
00011 00000000 000007 0000000000 000007
00012 00000000 000001 0000000000 000001
00013 00000000 000001 0000000000 000001
00014 00000000 000001 0000000000 000001
 Command ===> _____________________________________________ Scroll ===> CSR
                                                                     24/015
```

Figure 16-10 Output from the EQACUOSA command

For each program area, the following fields are displayed:

Num	The sequential number of the listing
Listing	The name of the listing data set
Date	The date of the compile
Time	The time of the compile
PAs	The number of program areas in the listing
BPs	The number of break points in the listing
PA	The sequential number of the program area in the listing
ADR	When program area has run, this is the storage address of the program area
BPS	The number of break points that have been executed for the program area
EVNTS (events)	The number of break points that have executed for this program area
ACTVE (active)	The number of break points that are still in storage in the program area

16.1.17 Stop the monitor

From ISPF option **6**, run the command EXEC hlq.SEQAEXEC(EQACUOSP)

Example 16-4 is a sample reply.

Example 16-4 Output from the EQACUOSP command

```
************************** Top of Data **********************
Monitor session CHABERT stopped - session data written to disk
************************* Bottom of Data ********************
```

16.1.18 Create the summary report

For this example we submit the JCL (Tprogname) generated by the Create JCL for Reports step.

First part of the summary report

Example 16-5 is the first part of the summary report.

Example 16-5 DTCU - Summary report (Part 1 of 2)

```
*********  DTCU SUMMARY:                                          PROGRAM AREA DATA                    ********
              DATE: 11/21/2004
              TIME: 14:21:11
       TEST CASE ID:
|<--                    PROGRAM IDENTIFICATION                    -->|
|                          |                                        |  STATEMENTS:          | BRANCHES:          |
| PA LOAD MOD PROCEDURE    | LISTING NAME                           |   TOTAL     EXEC   %  |  CPATH   TAKEN   %  |
-------------------------------------------------------------------------------------------------------------------
   1 TRADERB  MAINLINE        CHABERT.TRADER.COBLIST(TRADERB)          12        0   0.0
   2          SETUP-FILES                                              10        0   0.0
   3          SETUP-FILES-EXIT                                          1        0   0.0
   4          CLOSEDOWN-FILES                                          19        0   0.0
   5          READ-TRANSACT-FILE                                       14        0   0.0
   6          READ-TRANSACT-FILE-EXIT                                   1        0   0.0
   7          BUY-SELL                                                 10        0   0.0
   8          BUY-SELL-EXIT                                             1        0   0.0
   9          BUY-SELL-BUY-FUNCTION                                    12        0   0.0
  10          BUY-SELL-BUY-FUNCTION-EXIT                                1        0   0.0
  11          CALCULATE-SHARES-BOUGHT                                   5        0   0.0
  12          CALCULATE-SHARES-BOUGHT-EXIT                              1        0   0.0
  13          CALCULATE-SHARES-SOLD                                     1        0   0.0
  14          CALCULATE-SHARES-SOLD-EXIT                                1        0   0.0
  15          BUY-SELL-SELL-FUNCTION                                   14        0   0.0
  16          BUY-SELL-SELL-FUNCTION-EXIT                               1        0   0.0
  17          VALIDATE-COMPANY-EXISTS                                   2        0   0.0
  18          VALIDATE-COMPANY-EXISTS-EXIT                              1        0   0.0
  19          GET-SHARE-VALUE                                           9        0   0.0
  20          READ-CUSTFILE                                            15        0   0.0
  21          READ-CUSTFILE-EXIT                                        1        0   0.0
  22          WRITE-CUSTFILE                                           15        0   0.0
  23          WRITE-CUSTFILE-EXIT                                       1        0   0.0
  24          REWRITE-CUSTFILE                                         12        0   0.0
  25          REWRITE-CUSTFILE-EXIT                                     1        0   0.0
  26          READ-COMPFILE                                            13        0   0.0
  27          READ-COMPFILE-EXIT                                        1        0   0.0
  28          BUILD-NEW-CUSTOMER                                       13        0   0.0
  29          BUILD-NEW-CUSTOMER-EXIT                                   1        0   0.0
  30          SET-DUMMY-CUST-RECORD                                     7        0   0.0
  31          GENERATE-CUSTOMER-REPORT                                  9        0   0.0
  32          GENERATE-CUSTOMER-REPORT-EXIT                             1        0   0.0
  33          CALCULATE-SHARE-VALUE                                     8        0   0.0
  34          START-CUSTFILE                                            8        0   0.0
  35          READ-CUSTFILE-NEXT                                        8        0   0.0
  36          WRITE-HEADER                                              7        0   0.0
  37          WRITE-DETAILS                                             5        0   0.0
  38          WRITE-TRANSACTION-REPORT                                 15        0   0.0
  39          WRITE-TRANSACTION-REPORT-EXIT                             1        0   0.0
-------------------------------------------------------------------------------------------------------------------
Summary for all PAs:                                                 258        0   0.0       0       0 100.0
```

The summary report tells you how much of the procedures (blocks) has been executed. If a procedure shows 100%, it means that all the statements have been executed at least one time. If a block or procedure shows less then 100%, it tells you how many are in that block, and how many statements have been executed.

Each section of the report includes a program identification area. This information gives the load module name, procedure name, and listing name for the program.

The columns in the PROGRAM IDENTIFICATION area are:

PA	The number of the program area.
LOAD MOD	The name of the load module.
PROCEDURE	COBOL: Paragraph name. PL/I: Procedure, ON-unit, or Begin-block name. C/C++: Function name. ASM: CSECT name.
LISTING NAME	The name of the listing. If the listing name is longer than 40 characters, only the right-most 40 characters are shown.

The section of the report called PROGRAM AREA DATA contains coverage statistics in addition to the program identification information. The columns in the coverage statistics area are:

STATEMENTS: TOTAL	The statements of code for this test case run.
STATEMENTS: EXEC	The statements of code that executed.
STATEMENTS: %	The percentage of statements that executed
BRANCHES: CPATH	The number of conditional branch paths.
BRANCHES: TAKEN	The number of conditional branch paths that executed.
BRANCHES: %	The percentage of conditional branch paths that executed.

Second part of the summary report

It shows exactly what statements in their correspondence blocks have not been executed, as shown in Example 16-6.

Example 16-6 DTCU - Summary report (Part 2 of 2)

```
*********  DTCU SUMMARY:                    UNEXECUTED CODE                    ********
              DATE: 11/21/2004
              TIME: 14:21:11
      TEST CASE ID:
|<--                     PROGRAM IDENTIFICATION                     -->|           UNEXECUTED CODE
|                            |                                          |
| PA LOAD MOD PROCEDURE      | LISTING NAME                             | start    end     start    end     start   end
-------------------------------------------------------------------------------------------------------------------------
  1 TRADERB  MAINLINE        CHABERT.TRADER.COBLIST(TRADERB)              328    346
  2          SETUP-FILES                                                  352    377
  3          SETUP-FILES-EXIT                                             381    381
  4          CLOSEDOWN-FILES                                              384    409
  5          READ-TRANSACT-FILE                                           417    439
  6          READ-TRANSACT-FILE-EXIT                                      444    444
  7          BUY-SELL                                                     448    460
  8          BUY-SELL-EXIT                                                464    464
  9          BUY-SELL-BUY-FUNCTION                                        472    489
 10          BUY-SELL-BUY-FUNCTION-EXIT                                   493    493
 11          CALCULATE-SHARES-BOUGHT                                      499    506
 12          CALCULATE-SHARES-BOUGHT-EXIT                                 510    510
 13          CALCULATE-SHARES-SOLD                                        515    515
 14          CALCULATE-SHARES-SOLD-EXIT                                   519    519
 15          BUY-SELL-SELL-FUNCTION                                       526    547
 16          BUY-SELL-SELL-FUNCTION-EXIT                                  551    551
 17          VALIDATE-COMPANY-EXISTS                                      554    555
 18          VALIDATE-COMPANY-EXISTS-EXIT                                 558    558
 19          GET-SHARE-VALUE                                              561    573
 20          READ-CUSTFILE                                                580    598
 21          READ-CUSTFILE-EXIT                                           602    602
 22          WRITE-CUSTFILE                                               606    623
 23          WRITE-CUSTFILE-EXIT                                          627    627
 24          REWRITE-CUSTFILE                                             631    644
 25          REWRITE-CUSTFILE-EXIT                                        648    648
 26          READ-COMPFILE                                                651    666
 27          READ-COMPFILE-EXIT                                           670    670
 28          BUILD-NEW-CUSTOMER                                           675    687
 29          BUILD-NEW-CUSTOMER-EXIT                                      690    690
 30          SET-DUMMY-CUST-RECORD                                        695    701
 31          GENERATE-CUSTOMER-REPORT                                     706    718
```

```
 32          GENERATE-CUSTOMER-REPORT-EXIT                          725    725
 33          CALCULATE-SHARE-VALUE                                  728    737
 34          START-CUSTFILE                                         742    752
 35          READ-CUSTFILE-NEXT                                     757    766
 36          WRITE-HEADER                                           771    777
 37          WRITE-DETAILS                                          781    785
 38          WRITE-TRANSACTION-REPORT                               789    809
 39          WRITE-TRANSACTION-REPORT-EXIT                          811    811
-------------------------------------------------------------------------------------------------------------------
```

The section of the report called UNEXECUTED CODE contains information for unexecuted code segments in addition to the program identification information. The columns for this area are:

start The line or statement number of the first unexecuted instruction in this unexecuted segment.

end The line or statement number of the last unexecuted instruction in this unexecuted segment.

The number that appears for start and end is the number that is used to identify each line or statement in the compiler listing.

16.1.19 Create the summary and annotated listing report

The first part of this report is the summary report described in the previous section. Example 16-7 is an abstract of the annotated listing.

Example 16-7 DTCU - Annotated listing

```
 000520                 ******************************************************************
 000521                  BUY-SELL-SELL-FUNCTION SECTION.
 000522                 * CHECK WE HAVE A RECORD FOR THIS CUSTOMER.COMPANY, IF NOT EXIT
 000523                 * CHECK THAT WE CAN MEET THE SELL REQUEST, IF NOT EXIT
 000524                 * CALCUATE NEW NUMBER OF SHARES AND UPDATE CUSTFILE
 000525                 * CALCULATE NEW SHARE TOTAL SHARE VALUE
 000526 ^                   MOVE 'ENTRY FOR SELL' TO COMMENT-FIELD                          82
 000527                 * CHECK WHETHER WE HAVE ANY SHARES TO SELL
 000528 ^                   PERFORM READ-CUSTFILE                                           579
 000529                     EVALUATE RETURN-VALUE                                           178
 000530 ^                     WHEN CLEAN-RETURN                                             180
 000531 ^    1                     IF TR-NO-OF-SHARES IS GREATER THAN DEC-NO-SHARES         154 118
 000532      1                     THEN
 000533 ^    2                       MOVE INVALID-SALE TO RETURN-VALUE                      190 178
 000534 ^    2                       MOVE TOO-MANY-SHARES-MSG TO COMMENT-FIELD              101 82
 000535      1                     ELSE
 000536 ^    2                       PERFORM CALCULATE-SHARES-SOLD                          512
 000537 ^    2                       PERFORM REWRITE-CUSTFILE                               629
 000538 ^    2                       IF RETURN-VALUE = CLEAN-RETURN                         178 180
 000539 ^    3                          PERFORM WRITE-TRANSACTION-REPORT                    788
 000540      2                       END-IF
 000541                 * @TEST 2 LINES
 000542      1                     END-IF
 000543 ^                     WHEN CUSTOMER-NOT-FOUND                                       194
 000544 ^    1                     MOVE INVALID-SALE TO RETURN-VALUE                        190 178
 000545 ^    1                     MOVE NO-SHARES-MSG TO COMMENT-FIELD                      103 82
 000546                       WHEN OTHER
 000547 ^    1                     MOVE BAD-CUST-READ TO RETURN-VALUE                       184 178
 000548                     END-EVALUATE
 000549                     .
 000550                  BUY-SELL-SELL-FUNCTION-EXIT.
 000551 ^                   EXIT.
 000552                 ******************************************************************
```

DTCU annotation symbols

Each instruction line of the listing has a character to the right of the statement number to indicate what happened during the test run:

& A conditional branch instruction that has executed both ways

> A conditional branch instruction that has branched but not fallen through

V A conditional branch instruction that has fallen through but not branched

:	Non-branch instruction that has executed
^	Instruction that has not executed
@	Data area in the assembler listing
%	An unconditional branch instruction that has executed in the assembler listing

16.1.20 Use of the monitor panel

Some of the monitor-related functions can also been done from a DTU panel.

Select option **5** (**Monitor**) from the Coverage Utility panel, and the panel shown in Figure 16-11 is displayed.

```
------------------------------ Control the Monitor ------------------------------
Option ===> _____________________________________________________________________

1  Start         Create JCL to Start the Monitor

2  Stop          Stop monitor execution normally                  (EQACUOSP)

3  SessDisplay   Display all active sessions                      (EQACUOSE)

4  Listings      Display listings                                 (EQACUOSL)
5  Statistics    Display statistics                               (EQACUOSA)
6  BPDisplay     Display Breakpoint status                        (EQACUOBP)

7  AddId         Specify a unique testcase id                     (EQACUOID)
8  Snapshot      Take snapshot of data                            (EQACUOSN)
9  Reset         Reset all data in monitor                        (EQACUORE)

10 Quit          Terminate monitor without saving breakpoint data (EQACUOQT)

Enter END to Terminate

                                                                   02/014
```

Figure 16-11 DTCU - Control the Monitor

The options are:

- Option **1, Start**: This will generate the JCL to start the monitor (similar to 16.1.9, “Create JCL to start a monitor session” on page 374).
- Option **2, Stop** the monitor: This is the ISPF version of the command line version of the function shown in 16.1.17, “Stop the monitor” on page 378.
- Option **3, Display all active sessions**: This shows the currently active monitor sessions.
- Option **4, Listings**: This shows the listing data sets that were actually used (compiler listings).
- Option **5, Statistics**: These are the ISPF versions of the command line versions shown in 16.1.16, “Display statistics online” on page 377.
- Option **6, BPDisplay**: This shows the breakpoint status, the corresponding assembler hooks, and their offsets.
- Option **7, Specify a unique test case ID**: This lets you specify an ID for this test case.
- Option **8, Snapshot**: This shows the breakpoint ID, if any, and the name of the saved statistical data.
- Option **9, Reset**: This is used to reset all monitor data.

- Option **10, Quit**: This is used to quit.

16.2 COBOL and CICS Command Level Conversion Aid

The ability to compile old COBOL programs using the convert and compile option of Debug Tool Utilities and Advanced Functions provides you with an easy way of debugging old COBOL programs, while giving you the option of using the converted source and new compiler, or continuing to use the old source and old compiler. This section describes what the convert and compile option needs, and what steps are required to set it up.

16.2.1 Requirements for the conversion tools

To successfully convert an old program to ANSI85 standard, you need:

- The source code composing your application
- Any copy books associated with the application

16.2.2 Setting up for convert and compile

In this part, we show the steps needed to convert an OS/VS COBOL Program to the ANSI85 standard. We start from DTU, either by selecting DTU directly from ISPF, or by selecting from ISPF, option **6** (TSO commands) and issuing the following command:

```
EXEC 'D/T_high_lvl_qualifier.SEQAEXEC(EQASTART)'
```

From the main DTU panel, select **1: Program Preparation** as shown in Figure 16-12.

```
---------------------------- Debug Tool Utilities ----------------------------
Option ===> _____________________________________________________________________

                                                                   More:     +
0  Manage Job Card
   For Program Preparation and Setup File Management

1  Program Preparation
   Compile old or new COBOL programs with newer compilers, convert old COBOL
   source into new COBOL source, use other compilers, and link edit.

2  Manage and Use Debug Tool Setup Files
   You can manage setup files and use them to run your program interactively
   with Debug Tool in TSO Foreground or submit your program to run in
   the background using MVS batch.

3  Code Coverage
   Measure code coverage in programs written in COBOL, PL/I, C/C++ and
   Assembler when compiled with specific IBM compilers and HLASM.

4  Manage IMS Programs

                                                                     02/014
```

Figure 16-12 CCCA - Getting to the COBOL Convert and Compile interface

Note: If you intend to do your conversion and compilation in batch, choose option 0 and verify your job card before selecting option 1 from the main menu.

The Debug Tool Program Preparation Menu shown in Figure 16-13 is displayed.

```
------------------------ Debug Tool Program Preparation ------------------------
Option ===> _____________________________________________________________________

                                                                    More:     +
1  COBOL Compile
   Using 5655-G53 IBM Enterprise COBOL for z/OS and OS/390

2  COBOL Convert and Compile
   Using 5648-B05 COBOL and CICS Command Level Conversion Aid
   and   5655-G53 IBM Enterprise COBOL for z/OS and OS/390

3  PL/I  Compile
   Using 5655-H31  IBM(R) Enterprise PL/I for z/OS

4  C and C++ Compile
   Using 5694A01 z/OS C/C++

5  Assemble
   Using High Level Assembler

L  Link Edit
   Using z/OS Binder

                                                                    02/014
```

Figure 16-13 CCCA - Debug Tool Program Preparation

Ensure that the Compiler and Converter Settings are correct

1. Verify the settings for the Enterprise COBOL Compiler and the COBOL and CICS Command Level Conversion Aid. To do this you simply page forward (F8) and select option **S** as shown in Figure 16-14.

```
------------------------ Debug Tool Program Preparation ------------------------
Option ===> _____________________________________________________________________

                                                                    More:   -

4  C and C++ Compile
   Using 5694A01 z/OS C/C++

5  Assemble
   Using High Level Assembler

L  Link Edit
   Using z/OS Binder

F  Fault Analyzer Side file Create
   Using 5655-G74 IBM Fault Analyzer for z/OS and OS/390

C  Convert old 68/74 Std COBOL to 85 Std COBOL
   Using 5648-B05 COBOL and CICS Command Level Conversion Aid

S  Manage System and User Settings

                                                                    02/014
```

Figure 16-14 CCCA - Manage System and User Settings selection

2. On the panel shown in Figure 16-14, set up the COBOL and Conversion Utilities appropriately for your system using selection **C** and selections 2 and 9 from the screen shown in Figure 16-15.

```
-------------- Debug Tool Program Preparation - Manage Settings --------------
Command ===> _____________________________________________________________

                                                                   More:     +
System Settings for Program Preparation (compiler product datasets, etc):
These settings are refreshed at the start of every session.
  1  COBOL Compile
  2  COBOL Conversion (CCCA)
  3  PL/I  Compile
  4  C and C++ Compile
  5  HLASM
  6  Link Edit
  7  Fault Analyzer IDILANGX

User Settings for Program Preparation (input and output datasets, etc):
These settings are saved between sessions until RESET.
  8  COBOL Compile
  9  COBOL Convert and Compile
 10  PL/I  Compile
 11  C and C++ Compile
 12  HLASM
 13  Link Edit
 14  Fault Analyzer IDILANGX

                                                                    02/015
```

Figure 16-15 CCCA - Debug Tool Program Preparation - Manage Settings

3. Select **2: COBOL Conversion (CCCA)** to define the system settings as follows:
   a. Define the CCCA library, or LINKLIST if the library is in the MVS linklist.
   b. Define the shared and private VSAM and non-VSAM data set qualifiers.
   c. We recommend that you do not change the allocation sizes for the different data sets.
   d. Press PF3 to save.
4. Select **9: COBOL Convert and Compile** to define the user settings.
5. Change the data set qualifiers and their allocation sizes and DB2 precompiler and CICS translator option.

Note: Remember the DTU's help function. Select the fields using the cursor and press F1.

Once you have finished the configuring the settings, you can use F3 until you get to the Debug Tool Program Preparation Menu.

16.2.3 Start the conversion process

1. To actually do the conversion, select option **2** from the Debug Tool Program Preparation Menu as shown in Figure 16-16.

```
------------------------ Debug Tool Program Preparation ------------------------
Option ===> 2_____________________________________________________________________

                                                                        More:     +
1  COBOL Compile
   Using 5655-G53 IBM Enterprise COBOL for z/OS and OS/390

2  COBOL Convert and Compile
   Using 5648-B05 COBOL and CICS Command Level Conversion Aid
   and   5655-G53 IBM Enterprise COBOL for z/OS and OS/390

3  PL/I  Compile
   Using 5655-H31  IBM(R) Enterprise PL/I for z/OS

4  C and C++ Compile
   Using 5694A01 z/OS C/C++

5  Assemble
   Using High Level Assembler

L  Link Edit
   Using z/OS Binder

                                                                          02/015
```

Figure 16-16 CCCA - COBOL Convert and Compile

2. This will bring up the Debug Tool Program Preparation - Convert and Compile page as shown in Figure 16-17.

```
-----------Debug Tool Program Preparation - COBOL Convert and Compile----------
Command ===> ______________________________________________________________________

Convert using 5648-B05 COBOL and CICS Command Level Conversion Aid
Compile using 5655-G53 IBM Enterprise COBOL for z/OS and OS/390

F Foreground or Batch Processing  (F B)
                                                                        More:     +
Specify primary input data set for conversion and compilation.
Source Library:
   Project . . . ________
   Group . . . . ________ . . . ________ . . . ________ . . . ________
   Type  . . . . ________
   Member  . . . ________        (Blank or pattern for member selection list)

Other Partitioned or Sequential Data Set:
   Data Set Name . . . ______________________________________________________
   Volume Serial . . . ______    (If not cataloged)

Test options, data set name patterns, CICS and SQL processing options:
_ Enter / to edit options and data set name patterns
N CICS    I Integrated CICS translator, S Separate CICS translator, N None
N DB2/SQL I Integrated SQL coprocessor, S Separate DB2 precompiler, N None

                                                                          02/015
```

Figure 16-17 CCCA - The top part of Convert and Compile panel

Figure 16-17 looks very similar to the other program preparation panel for COBOL compile. The steps to follow are:

1. Select a foreground or background convert/compile.
2. Enter the name of the OS/VS COBOL (or other COBOL) source data set.
3. Page forward to see the second half of the panel as shown in Figure 16-18.

```
-----------Debug Tool Program Preparation - COBOL Convert and Compile----------
Command ===> _____________________________________________________________________

Convert using 5648-B05 COBOL and CICS Command Level Conversion Aid
Compile using 5655-G53 IBM Enterprise COBOL for z/OS and OS/390

F Foreground or Batch Processing  (F B)
                                                                   More:   - +

Compiler Options (TEST compiler option is automatically added):
_____________________________________________________________________________

Specify secondary input data sets for Copy processing:
DDname         Syslib data set Names
  SYSLIB  . . . ____________________________________________
  ________ . . . ____________________________________________
  ________ . . . ____________________________________________
  ________ . . . ____________________________________________
  ________ . . . ____________________________________________
  ________ . . . ____________________________________________
  ________ . . . ____________________________________________
  ________ . . . ____________________________________________
  ________ . . . ____________________________________________
                                                          02/015
```

Figure 16-18 CCCA - The bottom part of the Convert and Compile panel

4. Eventually, you need to:
   - Mark for DB2 and /or CICS if EXEC SQL or EXEC CICS statements are in source.
   - Add or change compiler options.
   - Add SYSLIB data sets for your copybooks.
5. Press Enter and select the member to be converted/compiled.

The next panel shows the input data set, compiler options, and all data sets for the convert, compile, and precompile/translate step. You can modify any of these data set names and options.

We used the sample OS/VS COBOL program, which includes the EXHIBIT statement as shown in Example 16-8.

Example 16-8 Original OS/VS COBOL source

```
IDENTIFICATION DIVISION.
      PROGRAM-ID. MINI.
      ENVIRONMENT DIVISION.
      DATA DIVISION.
      WORKING-STORAGE SECTION.
        77 X PIC 99.
      PROCEDURE DIVISION.

          DISPLAY "MINI!".

          EXHIBIT X
          DISPLAY "MINI!".

          GOBACK.
```

1. Press Enter to start the foreground convert/compile. If you selected for background compile check the upcoming JCL and press Enter.

2. If any of the steps did not work, browse the corresponding conversion listing files directly from the next panel. Edit the appropriate input data sets. Example 16-9 shows the converted source.

Example 16-9 OS/VS program after conversion

```
000010 IDENTIFICATION DIVISION.
000020 PROGRAM-ID. MINI.
000030*              PROGRAM CONVERTED BY
000040*              CCCA FOR OS/390 & MVS & VM 5648-B05
000050*              CONVERSION DATE 10/03/02 15:48:24.
000060 ENVIRONMENT DIVISION.
000070 DATA DIVISION.
000080 WORKING-STORAGE SECTION.
000090   77 X PIC 99.
000100 PROCEDURE DIVISION.
000110
000120     DISPLAY "MINI!".
000130
000140     DISPLAY
000150             "X = " X " "
000160     DISPLAY "MINI!".
000170
000180     GOBACK.
```

Now, how does convert/compile know the source and target language level and other options? It inherits these options from the settings you provide in the Convert old 68/74 Standard COBOL to 85 Standard COBOL panel. Select option **1** from the DTU panel. In this sample we used:

Source language level: 3 OS/VS COBOL LANGLVL(1)

Target language level: 4 COBOL for OS/390

You can also overwrite the source language level in the Edit data set name patterns and other options panel. Use the help function to find out what the numbers mean, for example, 3. You get to the Edit Data Set Name Patterns and Other Options panel by inserting a slash (/) just in front of the Enter / to edit data set name patterns, and DB2 or CICS translation options line.

Now exit out to the Program Preparation panel and select **L link** to link your program.

Finally, run the program and use the Debug Tool to debug your converted application.

16.2.4 Convert old 68/74 COBOL to 85 COBOL

The CCCA conversion tool can be selected from the Program Preparation panel option **C,** Convert old 68/74 Std COBOL to 85 Std COBOL as shown in Figure 16-19.

```
------------------------ Debug Tool Program Preparation ------------------------
Option ===> _____________________________________________________________________

                                                                    More:   -

4  C and C++ Compile
   Using 5694A01 z/OS C/C++

5  Assemble
   Using High Level Assembler

L  Link Edit
   Using z/OS Binder

F  Fault Analyzer Side file Create
   Using 5655-G74 IBM Fault Analyzer for z/OS and OS/390

C  Convert old 68/74 Std COBOL to 85 Std COBOL
   Using 5648-B05 COBOL and CICS Command Level Conversion Aid

S  Manage System and User Settings

                                                                      02/014
```

Figure 16-19 Choosing the option to do conversion only

Figure 16-20 shows the CCCA main panel.

```
----------------------------- CCCA Master Menu -------------------------------
Option ===> ▌
                                                  Userid    - REDBK1
                                                  Terminal  - 3278A
   1  CONVERT    - Convert COBOL source programs  Time      - 10:12
                                                  PF Keys   - 12
   2  CUSTOMIZE  - LCP Development Aid            Applid    - ISR

   0  OPTIONS    - Set environment and conversion options





   COBOL and CICS Command Level Conversion Aid for OS/390 & MVS & VM
                  5648-B05  Version 2  Release 1
        Copyright (C) IBM Corp 1982, 1998 - All rights reserved


 F1=Help      F2=Split     F3=Exit      F5=Refresh   F7=Up        F8=Down
F12=Cancel
```

Figure 16-20 CCCA Master Menu

Before you start converting an OS/VS COBOL program, select option **1.1.1** for Environment. You will find the option (like jobcard and data set prefixes) preset with what you already entered in system and user settings.

Now go back to the CCCA Master Menu and choose 0.2. This brings you to the menu shown in Figure 16-21. You can define the source language level and the target language level on this panel.

```
---------------------------- CCCA Language Level ------------------------------
Command ===> _

   Source language level ===> 3   1. DOS/VS COBOL LANGLVL(1)
                                  2. DOS/VS COBOL LANGLVL(2)
                                  3. OS/VS COBOL LANGLVL(1)
                                  4. OS/VS COBOL LANGLVL(2)
                                  5. VS COBOL II Release 1.0  1.1  2.0, or
                                     any COBOL with the CMPR2 option
                                  6. VS COBOL II NOCMPR2 Release 3.0  3.1  3.2
                                  7. VS COBOL II NOCMPR2 Release 4.0
                                  8. COBOL/370 NOCMPR2
                                  9. COBOL/VSE NOCMPR2
                                 10. COBOL for MVS and VM NOCMPR2
                                 11. COBOL for OS/390 and VM NOCMPR2


   Target language level ===> 4   1. VS COBOL II
                                  2. COBOL/VSE
                                  3. COBOL for MVS and VM
                                  4. COBOL for OS/390
 F1=Help      F2=Split     F3=Exit      F5=Refresh   F7=Up        F8=Down
F12=Cancel
```

Figure 16-21 Language level options

The Conversion Options 1 panel displays options that are output-related, like list formatting and program generation options. Consider customizing the following options for your installation, as shown in Figure 16-22:

- Re-sequence source lines
- Reserved word suffix
- Flag report writer statements (affects Report Writer programs)
- Remove value clauses in file/linkage sections (affects Report Writer programs).

```
-------------------------- CCCA Conversion Options 1 --------------------------
Command ===> _

   Lines per report page . . . . . ===> 60      01 to 99

   Resequence source lines . . . . ===> Y       Y/N
   Sequence number increment . . . ===> 0010    0001 to 9999

   Reserved word suffix. . . . . . ===> 74      Default value 74
   Generate new program. . . . . . ===> Y       Y/N
   Generate new copy members . . . ===> Y       Y/N
   Replace like-named copy members ===> Y       Y/N
   Print old source lines. . . . . ===> Y       Y/N
   Print copy members. . . . . . . ===> Y       Y/N
   Print diagnostics of level >= . ===> 00      00 to 99
   Report Heading. . . . . . . . . ===> SAMPLE REPORT
   Generate tokenization listing . ===> Y       Y/N



 F1=Help      F2=Split     F3=Exit      F5=Refresh   F7=Up        F8=Down
F12=Cancel
```

Figure 16-22 CCCA - Conversion option 1

The options included on the Conversion Options 2 panel shown in Figure 16-23 are conversion-specific.

```
------------------------- CCCA Conversion Options 2 ---------------------------
Command ===> _
Option
     1. Check procedure names . . . . . . . . . . . . ===> Y   Y/N
     2. Flag Report Writer statements . . . . . . . . ===> Y   Y/N
     3. Remove obsolete elements. . . . . . . . . . . ===> Y   Y/N
     4. Negate implicit EXIT PROGRAM. . . . . . . . . ===> Y   Y/N
     5. Generate END PROGRAM header . . . . . . . . . ===> N   Y/N
     6. Compile after converting. . . . . . . . . . . ===> Y   Y/N
     7. Flag manual changes in new source programs. . ===> N   Y/N
     8. Add DATE FORMAT clause to date fields . . . . ===> N   Y/N
     9. Remove VALUE clauses in File/Linkage Sections ===> Y   Y/N
    10. Flag IF FILE-STATUS (NOT) = "00". . . . . . . ===> Y   Y/N
    11. Flag BLL cell arithmetic. . . . . . . . . . . ===> Y   Y/N
    12. BLL cell conversion method. . . . . . . . . . ===> A   A/B
    13. Search source for literal delimiter . . . . . ===> Y   Y/N
    14. Literal delimiter (QUOTE or APOST). . . . . . ===> Q   Q/A
    15. . . . . . . . . . . . . . . . . . . . . . . . ===> N   Y/N


  Note: Option numbers appear on the Program/File report
 F1=Help      F2=Split     F3=Exit      F5=Refresh   F7=Up        F8=Down
F12=Cancel
```

Figure 16-23 CCCA - Conversion option 2

Note: Option 6 (Compile after converting) means that the program is compiled but there is no output saved.

Returning to the CCCA Master Menu, choose selection **2**, CUSTOMIZE - LCP Development Aid (Language Conversion Program). This selection allows you to:

- Change the reserved word table
- Compile LCP source
- Delete LCP or activate/deactivate debugging for an LCP
- Generate a directory of the LCP library
- Update message file
- Set environment and conversion options
- Convert COBOL source programs

Setting environment and conversion options selection from this menu is identical to selecting options from the CCCA Master Menu.

Selecting Convert COBOL Source Programs from this menu is identical to selecting Conversion from the main CCCA panel.

Now select **7 CONVERT PROGRAM - Convert COBOL source programs**

The first panel presents you with the job statement information. Verify the data and press Enter.

The next panel lets you enter the source input library, the program and copy library, and the options (like SQL, CICS, DLI).

After pressing Enter, you can select one or more members from the source input library. Return from this panel with F3 Exit.

The conversion submission panel lets you either select more members to convert (press Enter), submit the JCL with or without return, or cancel the conversion.

Press PF3 to submit the job (or jobs) and to return to the CCCA converter menu.

To know the results and statistics of your conversions, select from the CCCA convertor menu options 3 to 8, and L. Option E lets you delete the conversion statistics.

16.2.5 Summary

Debug Tool Utilities and Advanced Functions provides you with an easy way of dealing with old COBOL programs:

- You can continue to use the old source and just use the convert and compile options for debugging. (This would allow you to continue to use the old compiler for the production version of the program if you wish.)
- You can convert your old COBOL source to 85 COBOL source on the fly and debug or just use the facility to create your new production drivers.
- You can convert your old COBOL source to 85 COBOL and keep the converted source (recommended) for use in your normal production and debugging.

Part 5

Fault Analyzer for z/OS

Part 5 covers Fault Analyzer for z/OS (FA). Among the topics covered are the following:

- Settings and customizations
- Interactive analysis
- Real-time analysis report
- Batch analysis report
- FA and subsystems

17

Fault Analyzer settings and customization

This chapter describes how to set up and customize Fault Analyzer (FA). The audience for this information is systems programmers and application programmers.

17.1 Invocation exits

For FA to analyze an abend, it must be set up to be invoked through the appropriate abend processing exit. The following exits are required for FA to function effectively:

- CICS global user exit, IDIXrf CX52 or IDIXCX53
- MVS pre-dump exit, IDIXDCAP
- Language Environment batch abnormal termination exit IDIXCEE
- Language Environment batch abnormal termination exit IDIXCCEE

17.2 Fault Analyzer ISPF interface

At any time after an abend you can, as a TSO user, start the Fault Analyzer ISPF interface to review the fault.

Using this interface you can:

- View the stored real-time analysis report
- Start a batch reanalysis
- Start an interactive reanalysis
- View information about the fault
- Delete the fault entry

The ISPF interface also permits you to review the Fault Analyzer User's Guide and Reference on-line by using BookManager® Read against the current softcopy version of the book.

You can only perform reanalysis of a fault if either a minidump or a SYSMDUMP was written.

Compiler listing or side file data sets that were allocated or specified via the DataSets option when the real-time analysis took place will automatically be reused if performing reanalysis (if they are available in the reanalysis environment).

To make the reanalysis different from the initial real-time analysis, you must do one (or more) of the following:

- Supply compiler listings (or side files) for the programs involved in the abend (if they were not available for the initial real-time analysis).
- Change analysis options.
- Use the interactive reanalysis to review dump storage.

The main differences between the batch and interactive reanalysis steps are:

- Interactive reanalysis always provides full detail, and lets you look at storage locations that might not be included in the analysis report, whereas batch reanalysis provides the level of detail you ask for through the Detail option, and does not let you look at storage locations.
- Interactive reanalysis ties up the use of an ISPF session, whereas once you submit batch reanalysis jobs you can get on with other things.

If you want to supply a listing or side file so that Fault Analyzer can provide source line information when it performs the fault reanalysis, you must compile the program and then store the compiler listing or side file.

To create a side file from a compiler listing, a program named IDILANGX is used. The sample JCL shown in Figure 17-1 does the following:

- Compiles a COBOL program.
- Executes IDILANGX to process the listing and store it as a side file where Fault Analyzer can access it.

> **Note:** You can only compile one program per compile step in order to name the compiler listing PDS(E) member (if using a partitioned data set), and to ensure that only one compiler listing is written to the output file.

```
//IDISCMPS JOB (GSF),'GENERATE.SIDE.FILE',NOTIFY=&SYSUID.,
//         MSGCLASS=X,CLASS=A,MSGLEVEL=(1,1)
//         JCLLIB ORDER=(IGY.V2R1M0.SIGYPROC) <-- INSTALLATION
//*                                                IGYWCLG PROC
//*
//*********************************************************/
//* THIS JOB RUNS A COBOL COMPILE PLUS PRODUCES A SIDE FILE  */
//* FROM A PROGRAM LISTING THAT FAULT ANALYZER CAN USE FOR   */
//* OBTAINING Source INFORMATION.                            */
//* THE COMPILE OUTPUT IS THEN WRITTEN TO SYSUT2 IN THE      */
//* IEBGENER STEP.                                           */
//*********************************************************/
//*
//CBLRUN   EXEC IGYWC,PARM.COBOL='LIST,MAP,Source,XREF'
//COBOL.SYSIN DD DATA,DLM='##'
 CBL APOST,NOOPT,DYNAM,SSRANGE,RENT
       IDENTIFICATION DIVISION.
       PROGRAM-ID. IDISCBL1
       ENVIRONMENT DIVISION.
       INPUT-OUTPUT SECTION.
       FILE-CONTROL.
       DATA DIVISION.
       FILE SECTION.

       WORKING-STORAGE SECTION.
       01  FILLER                PIC X(20)  VALUE 'WORKING-STORAGE'.
       01  NUMBERX PIC 999999 COMP-3.
       01  ERROR-FLD.
           05  ERROR-COUNT PIC 999999 COMP-3.
           05  FLDY REDEFINES ERROR-COUNT.
               07 FLDZ PIC XXXX.
       01  BAD-RESULT PIC 99 COMP-3.

       PROCEDURE DIVISION.
       MAIN SECTION.
           DISPLAY '*** IDISCBL1 - START OF PROGRAM'.
       LOOP SECTION.
       START000.
           MOVE 3 TO ERROR-COUNT.
           ADD 986885 TO ERROR-COUNT GIVING NUMBERX.
           MOVE 'ABCD' TO FLDZ.
           IF NUMBERX > 0 THEN PERFORM CLEAR.
           DISPLAY '*** IDISCBL1 - END OF PROGRAM'.
           GOBACK.
       CLEAR SECTION.
       START001.
           DIVIDE NUMBERX BY ERROR-COUNT GIVING BAD-RESULT.
           EXIT.
       END PROGRAM IDISCBL1.
##
//COBOL.SYSPRINT DD DSN=&&COBLIST(IDISCBL1),
//         DISP=(,PASS),SPACE=(TRK,(10,5,5),RLSE),
//         DCB=(RECFM=FBA,LRECL=133,BLKSIZE=0)
//*
//IDILANGX EXEC PGM=IDILANGX,REGION=4096K,
// PARM='IDISCBL1 (COBOL ERROR'
//STEPLIB  DD  DISP=SHR,DSN=IDI.LOAD   1
//LISTING  DD  DISP=(OLD,PASS),DSN=&&COBLIST
//IDILANGX DD  DISP=SHR,DSN=IDI.IDILANGX
//SYSUDUMP DD  SYSOUT=*
```

Figure 17-1 Sample JCL to compile a cobol program and store a side file

17.2.1 Invoking the interface

How you invoke the Fault Analyzer ISPF interface depends on how it was customized. One way is to add an option to one of your ISPF selection panels. A systems programmer or the person who customized Fault Analyzer would be responsible of setting this up. To display the on-line help while in the interface, press the Help function key (PF1).

Note: Multiple concurrent invocations of Fault Analyzer (for example, using the split screens feature) by a single TSO/ISPF user is not supported.

The Fault Entry List display is shown when the Fault Analyzer ISPF interface is started. Figure 17-2 shows an example of a Fault Entry List display.

```
  File  Options  View  Services  Help
 ------------------------------------------------------------------------
 IBM Fault Analyzer - Fault Entry List                      Line 1 Col 1 80
 Command ===> _____________________________________________ Scroll ===> CSR

 Fault History File or View  : 'IDI.HIST'

 {The following line commands are available: ? (Query), V (View real-time
 report), I (Interactive reanalysis), B (Batch reanalysis), D (Delete).}

    Fault_ID Job/Tran User_ID  Sys/Job  Abend Date       Time
 __   F00007 JMONITOR n/a      STLABF6  U4093 2004/10/29 07:34:48
 __   F00006 CONOVERY CONOVER  STLABF6  SB37  2004/10/25 09:47:46
 __   F00005 CONOVERF CONOVER  STLABF6  S0CB  2004/10/21 11:05:18
 __   F00004 TIMOTHY  TIMOTHY  STLABF6  n/a   2004/10/11 16:21:13
 __   F00003 TIMOTHY  TIMOTHY  STLABF6  n/a   2004/10/11 14:31:39
 __   F00002 ZHONG    ZHONG    STLABF6  n/a   2004/10/11 10:53:17

 *** Bottom of data.


  F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
  F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 17-2 Fault entry list display

Note: If your Fault Entry List display does not show the PF keys, and you would like to see them, enter the ISPF command: `FKA ON`.

The history file or view that was last selected while using the Fault Analyzer ISPF interface is shown by default. The first time the interface is used, the initial history file name is obtained using the IDIHIST suboption of the DataSets option in effect. Another history file or view can be selected if desired. If a view was selected the last time the ISPF interface was used, and the view contains errors, then it is possible that an error display is presented prior to the Fault Entry List display. An example of an error display is shown in Figure 17-3. To exit from the error display, press PF3.

```
---------------------------------- Error ----------------------------------
                                                          Line 1 Col 1 76
 Command ===> ____________________________________________ Scroll ===> CSR
 The following problems were found while processing the view in
 DA.VIEWS(SWBAD1):

 * -HistCols syntax error: Missing starting parenthesis. The -HistCols
   specification has been ignored.

 * Data set 'xyz' open error: EDC5049I The specified file name could not be
   located.

 * -Match syntax error: The subcommand entered for the "FRED" command was
   invalid. The -Match specification has been ignored.

 Press PF3 to continue.

 *** Bottom of data.



  F1=Help     F3=Exit     F7=Up       F8=Down     F12=Cancel
```

Figure 17-3 Error display prior to the Fault Entry List display

The panel in Figure 17-3 shows:

- Each time the incorrect view member is read.
- When the Fault Entry List Column Configuration display is presented.

The identified errors in the view should be corrected to avoid this display. Entries in the Fault Entry List display are presented in reverse chronological order, with the most recent fault entry (based on abend date and time) shown at the top.

Each fault entry in the list occupies a single line and is identified by a fault ID on the left side of the display.

The default information is displayed if no HistCols option has been specified and no customization has been done by the user. You can use the displayed fields to identify the faults you are interested in, or reduce the display to only a subset of the faults using the MATCH command.

As shown at the top of the display if help text is enabled (shown in Figure 17-2), a number of line commands are available against individual history file entries.

You can show or hide help text using the VIEW selection on the Action field as shown in Figure 17-4.

```
  File  Options  View  Services  Help
 ------------ ┌──────────────────────────────┐ ──────────────────────────────
IBM Fault Anal │ _  *. Add Blank Lines        │        Line 1 Col 1 80
Command ===>   │    2. Remove Blank Lines     │       Scroll ===> CSR
               │    *. Add Help Text          │
Fault History  │    4. Remove Help Text       │
               │    *. Add Detail Information │
{The following  │    6. Remove Detail Information │ V (View real-time
report), I (In │    7. Preferred formatting Width... │ s), D (Delete).}
               │    8. Column Configuration...│
   Fault_ID Jo │    9. Refresh                │  Time     History_File_
     F00007 JM └──────────────────────────────┘  07:34:48 IDI.HIST
     F00006 CONOVERY CONOVER  STLABF6  SB37  2004/10/25 09:47:46 IDI.HIST
     F00005 CONOVERF CONOVER  STLABF6  S0CB  2004/10/21 11:05:18 IDI.HIST
     F00004 TIMOTHY  TIMOTHY  STLABF6  n/a   2004/10/11 16:21:13 IDI.HIST
     F00003 TIMOTHY  TIMOTHY  STLABF6  n/a   2004/10/11 14:31:39 IDI.HIST
     F00002 ZHONG    ZHONG    STLABF6  n/a   2004/10/11 10:53:17 IDI.HIST

*** Bottom of data.


 F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 17-4 VIEW selections to show or hide help text,

This screen responds to the standard Up, Down, Left, and Right commands, which by default are assigned to the PF7, PF8, PF10, and PF11 function keys respectively.

These can be used to scroll the display horizontally or vertically as needed to see all of the information available.

In the top right corner of the screen is the current top-most line number and indication of the left-most and right-most columns currently displayed.

The end of the fault entry list is indicated by the line:

```
*** BOTTOM OF DATA.
```

This line is used to indicate the bottom of all Fault Analyzer ISPF interface scrollable displays. You exit from the Fault Analyzer ISPF interface by issuing the Exit command (PF3) from the Fault Entry List display, or by selecting the Exit Fault Analyzer option from the Fault Entry List display File menu.

17.2.2 Using views

When it would be useful to concurrently view fault history file entries from more than a single fault history file, a view name can be specified instead of a fault history file name on the Fault Entry List display, or one can be selected via the File menu List Views option.

The definition of these views must be set up in the PDS(E) data set identified by the IDIVIEWS suboption of the DataSets option in the IDICNF00 parmlib config member. Apart from being able to see fault entries from multiple history files simultaneously, views can also be used to provide a specific column layout for the Fault Entry List display, or to provide a selection criteria for the initially displayed list of fault entries.

17.2.3 Setting up views

This section explains how to set up views for an installation and general information about using views.

To do the most effective setup of views for an installation, an understanding of the varied groups of users of Fault Analyzer is required. The main aspects of each user group that need to be determined are:

- Their data set security profiles for data sets with write access
- Their need to review or work on faults from jobs submitted by other user groups
- The need for read access only, or no access at all, to fault entries between groups. These requirements will already be understood to some degree by the installation security administrators, since they are the ones primarily dealing with data set security requirements.

An additional consideration is any requirement for fault visibility between the groups, for example:

- The security of PDS(E) fault history files is the data set security in your system.
- For users' jobs to record faults in the fault history file, they need write (update) access to the fault history file that the job will use.

All users in the same group will have read and delete access to the faults in a fault history file set up for the group. This will generally mean that it is better to maintain separate fault history files for different user groups, providing write (update) access only to the history file their jobs will use.

While this makes it harder for a user to look at the faults across the environment if they are in many different history files, a view can be created to let a user see a composite view of many fault history files in one Fault Analyzer ISPF display.

The list of fault history file data set names to make up a view are simply placed into a member in another PDS that has been created specifically to hold the view definitions. This is generally an installation-wide data set to which all users have read access, and that has the following characteristics:

- It is pointed to by the IDIVIEWS data sets in the DataSets option.
- The names of the members are used as the view names.
- The users are given a list of the view names when using the ISPF options pull down to change the fault history file and pressing F4 to list views.
- The first line of these members can be an optional comment starting with an asterisk in column one.
- If this comment line exists in a view member it is displayed beside of the view name in the selection screen. This allows the comment line to provide a description of the view to assist the users choice.

An example of a view data set member is shown in Example 17-1.

Example 17-1 Example of a view data set member

```
* List of History Files (1)
PO05.IFA.GROUP01.HIST (2)
PO05.IFA.GROUP02.HIST (3)
PO05.IFA.GROUP03.HIST (4)
* IMS history files (11)
IMS1.HIST (5)
```

```
IMS2.HIST (6)
* CICS history files (12)
CICS.REG01.HIST (7)
CICS.REG02.HIST (8)
* Including the system wide history file (9)
IDI.HIST (10)
```

Example 17-1 notes:

- (1) is a comment (indicated by the asterisk in column 1). Because this comment is on the first line of the view member, it will also be used as the view description when displaying the list of views using the Fault Analyzer ISPF interface.
- Lines identified as (2), (3), (4), (5), (6), and (7) are the fully qualified fault history file names that will be displayed simultaneously through the Fault Analyzer ISPF interface when selecting this view. All data set names must start in column 1 and be specified on a single line each. There is no limit on the total number of data set names that can be specified in a single view.
- The comments identified as (10), (11), and (12) will simply be ignored by the Fault Analyzer ISPF interface.

Important:

No specific view data set attributes are required, except that it must be a PDS(E).

The logical record length must be large enough to contain the longest data set name. The view member must not contain sequence numbers.

When creating the PDS(E), make sure the profile NUMBER is OFF or it will not work. Any sequential number at the end of any lines will result to failure as an invalid statement.

For easier reference to fault IDs across multiple history files, the ability to set up to three alphabetic fault prefix characters for individual fault history files is available. Using different prefix characters between different groups will help users recognize the owning group when viewing faults in a composite view display.

The fault prefix characters can be set or changed using the IDIUTIL batch utility SETFAULTPREFIX command.

A walk through the menus is presented beginning with Figure 17-5, using the VIEWS list, and adding the IDIVIEWS DDname to the IDICNG00 of the system parmlib.

Tip: Usually SYS1.PARMLIB if not re-pointed to another PARMLIB of your choice by receiving / applying SYSMOD IDISCNF using SMP/E job sample IDISCNF of install PDS suffix SIDISAM1.

```
  File  Options  View  Services  Help
 ------------------------------------------------------------------------
 IBM Fault Analyzer - Fault Entry List                        Line 1 Col 1 80
 Command ===> VIEWS_                                        Scroll ===> CSR

 Fault History File or View  : 'IDI.HIST'

 {The following line commands are available: ? (Query), V (View real-time repor
 D (Delete).}

    Fault_ID History_File_DSN                                System   Job/Tran Abe
__    F00010 IDI.HIST                                        STLABF6  IMWEBF6  S42
__    F00009 IDI.HIST                                        STLABF6  IMWEBF6  U40
__    F00008 IDI.HIST                                        STLABF6  IMWEBF6  U40
__    F00007 IDI.HIST                                        STLABF6  JMONITOR U40
__    F00006 IDI.HIST                                        STLABF6  CONOVERY SB3
__    F00005 IDI.HIST                                        STLABF6  CONOVERF S0C
__    F00004 IDI.HIST                                        STLABF6  TIMOTHY  n/a
__    F00003 IDI.HIST                                        STLABF6  TIMOTHY  n/a
__    F00002 IDI.HIST                                        STLABF6  ZHONG    n/a

 *** Bottom of data.
  F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
  F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 17-5 VIEWS typed on the command line

As the default, the history file is displayed first on the Main Menu of Fault Analyzer, the Fault Entry List menu display.

To activate the list members of the PDS(E) data set, customized to the IDICNF00 member of the PARMLIB for DDname IDIVIEWS, the view can be selected in either of two ways:

1. Type `VIEWS` on the command line as shown in Figure 17-5.
2. On the Fault Entry List display panel, click **File** and press Enter; then from the selection menu provided by the File menu, select option **4** and press Enter, as shown in Figure 17-6.

```
  File  Options  View  Services  Help
 ------------------------------------------------------------------------
 | 4 _1. Last Accessed Fault History Files or Views...|        Line 1 Col 1 80
 |    2. Last Accessed Fault History File Entries...  |       Scroll ===> CSR
 |    3. Clear Last Accessed Information              |
 |    4. List Views...                                |
 |    5. Analyze MVS Dump Data Set...                 |
 |    6. Fault History File Properties...             |  (View real-time repor
 |    7. Exit Fault Analyzer                          |
 ------------------------------------------------------

    Fault_ID History_File_DSN                                System   Job/Tran Abe
      F00010 IDI.HIST                                        STLABF6  IMWEBF6  S42
      F00009 IDI.HIST                                        STLABF6  IMWEBF6  U40
      F00008 IDI.HIST                                        STLABF6  IMWEBF6  U40
      F00007 IDI.HIST                                        STLABF6  JMONITOR U40
      F00006 IDI.HIST                                        STLABF6  CONOVERY SB3
      F00005 IDI.HIST                                        STLABF6  CONOVERF S0C
      F00004 IDI.HIST                                        STLABF6  TIMOTHY  n/a
      F00003 IDI.HIST                                        STLABF6  TIMOTHY  n/a
      F00002 IDI.HIST                                        STLABF6  ZHONG    n/a

 *** Bottom of data.
  F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
  F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 17-6 List VIEWS by using the selection menu of the File Menu

The View List panel shown in Figure 17-7 is displayed. From this panel you can:

1. Browse one of the member names of the PDS(E) DDname IDIVIEWS of the IDICNF00 in the PARMLIB.
2. Select one of the member names.

```
  File  Options  View  Services  Help
 ┌──────────────────────────────── View List ────────────────────────────────┐
 │                                                        Row 1 to 3 of 3    │
 │ Command ===> _                                       Scroll ===> CSR      │
 │ Line commands: S (select) B (browse).                                     │
 │    Name     Description                                                   │
 │    BATCHIST USER GRACINE BATCH HISTORY FILES                              │
 │    CICSHIST CICS HISTORY FILES FOR USER GRACINE                           │
 │    VLSTCICS CICS LIST OF HISTORY FILES                                    │
 │ ************************** Bottom of data **************************      │
 │                                                                           │
 │  F1=Help    F3=Exit    F7=Up      F8=Down   F12=Cancel                   │
 └───────────────────────────────────────────────────────────────────────────┘
 F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 17-7 View list

Browsing the view list

Browsing a member of the view list provides the name of the history file customized and can be very specific to a selection of history files, as shown in Figure 17-8.

```
  File  Options  View  Services  Help
 ┌──────────────────────────────── View List ────────────────────────────────┐
 │                                                        Row 1 to 3 of 3    │
 │ Command ===>                                         Scroll ===> CSR      │
 │ Line commands: S (select) B (browse).                                     │
 │    Name     Description                                                   │
 │    BATCHIST USER GRACINE BATCH HISTORY FILES                              │
 │  b CICSHIST CICS HISTORY FILES FOR USER GRACINE                           │
 │    VLSTCICS CICS LIST OF HISTORY FILES                                    │
 │ ************************** Bottom of data **************************      │
 │                                                                           │
 │                                                                           │
 │                                                                           │
 │                                                                           │
 │                                                                           │
 │                                                                           │
 │                                                                           │
 │                                                                           │
 │                                                                           │
 │                                                                           │
 │  F1=Help    F3=Exit    F7=Up      F8=Down   F12=Cancel                   │
 └───────────────────────────────────────────────────────────────────────────┘
 F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 17-8 Selecting to browse member CICSHIST

Place a `b` in front of the member name you want to browse and press Enter. The contents of the member is displayed as shown in Figure 17-9.

```
File  Options  View  Services  Help
                    View List
                    View Member Browse
Contents of 'GRACINE.VIEWS(CICSHIST)'                    Row 1 to 5 of 5
Command ===> _____________________________________ Scroll ===> CSR
* CICS HISTORY FILES FOR USER GRACINE
IDI.HIST.CICSC22F
IDI.HIST.CICSC23G
IDI.HIST.CICSC31F
IDI.HIST.CICSC31G
************************** Bottom of data **************************

F1=Help    F3=Exit    F7=Up      F8=Down    F12=Cancel
```

Figure 17-9 View member browse display

Selecting the view list

Selecting a member of the view list will display the abends of the history files included in the member selected, as shown in Figure 17-10 and Figure 17-11.

```
File  Options  View  Services  Help
                    View List
                                                   Row 1 to 3 of 3
Command ===>                                     Scroll ===> CSR
Line commands: S (select) B (browse).
   Name     Description
s_ BATCHIST USER GRACINE BATCH HISTORY FILES
   CICSHIST CICS HISTORY FILES FOR USER GRACINE
   VLSTCICS CICS LIST OF HISTORY FILES
************************** Bottom of data **************************

F1=Help    F3=Exit    F7=Up      F8=Down    F12=Cancel

F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 17-10 Member selection from view list

```
  File  Options  View  Services  Help
 ------------------------------------------------------------------------
 IBM Fault Analyzer - Fault Entry List                        Line 1 Col 1 80
 Command ===> _____________________________________________ Scroll ===> CSR

 Fault History File or View  : (BATCHIST) USER GRACINE BATCH HISTORY FILES

 {The following line commands are available: ? (Query), V (View real-time repor
 D (Delete).}

    Fault_ID History_File_DSN                                System   Job/Tran Abe
__    F00011 IDI.HIST                                       STLABF6  FBIIDRUN S52
__    F00002 IDI.HIST.TEST                                  STLABF6  $TSTFACB S0C
__    F00001 IDI.HIST.TEST                                  STLABF6  $TSTFACB S0C
__    F00010 IDI.HIST                                       STLABF6  IMWEBF6  S42
__    F00009 IDI.HIST                                       STLABF6  IMWEBF6  U40
__    F00008 IDI.HIST                                       STLABF6  IMWEBF6  U40
__    F00007 IDI.HIST                                       STLABF6  JMONITOR U40
__    F00006 IDI.HIST                                       STLABF6  CONOVERY SB3
__    F00005 IDI.HIST                                       STLABF6  CONOVERF S0C
__    F00004 IDI.HIST                                       STLABF6  TIMOTHY  n/a
__    F00003 IDI.HIST                                       STLABF6  TIMOTHY  n/a
 F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 17-11 Fault entry list display using Views List member selection

17.2.4 Changing the history file or the view displayed

When the Fault Analyzer ISPF interface is started initially, the history file or view last displayed is shown.

To select a different history file or view, do the following:

- To specify a history file or view name to be displayed, type its name on the Fault History File or View line as the example shown in Figure 17-12.

```
  File  Options  View  Services  Help
 ------------------------------------------------------------------------
 IBM Fault Analyzer - Fault Entry List                        Line 1 Col 1 80
 Command ===> _____________________________________________ Scroll ===> CSR

 Fault History File or View  : 'IDI.HIST'

 {The following line commands are available: ? (Query), V (View real-time
 report), I (Interactive reanalysis), B (Batch reanalysis), D (Delete).}

    Fault_ID Job/Tran User_ID  Sys/Job  Abend Date       Time     History_File_
__    F00007 JMONITOR n/a      STLABF6  U4093 2004/10/29 07:34:48 IDI.HIST
__    F00006 CONOVERY CONOVER  STLABF6  SB37  2004/10/25 09:47:46 IDI.HIST
__    F00005 CONOVERF CONOVER  STLABF6  S0CB  2004/10/21 11:05:18 IDI.HIST
__    F00004 TIMOTHY  TIMOTHY  STLABF6  n/a   2004/10/11 16:21:13 IDI.HIST
__    F00003 TIMOTHY  TIMOTHY  STLABF6  n/a   2004/10/11 14:31:39 IDI.HIST
__    F00002 ZHONG    ZHONG    STLABF6  n/a   2004/10/11 10:53:17 IDI.HIST

 *** Bottom of data.


 F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 17-12 History file value name in the Fault Entry List

  - After typing the history file or view name, press Enter to show the fault entries.

Use the following rules for naming history files and views:

- For history file names, the standard TSO naming convention applies, that is, the name typed is automatically prefixed by the TSO prefix if not enclosed in single quotes as shown in Example 17-2.

Example 17-2 How history file is typed

If TSO prefix is set to IDI and the specified history file name is IDI.HIST, 2 choices to enter the history file name ON THE 'FAULT HISTORY FILE OR VIEW' LINE:

type `HIST`

type `'IDI.HIST'`

If missing, the ending quote is automatically added.

- View names are member names in one of the data sets associated with the IDIVIEWS DDname. These are specified by enclosing them in parenthesis.
- To specify that the view member ABC is to be displayed, type (ABC) on the Fault history file or view line. If missing, the closing parenthesis is automatically added.
- A record is maintained of the last 10 history files or views displayed. To select a previously used history file or view:
  a. Select **File** → **Last Accessed Fault History Files or Views**. This will bring up the Last Accessed Fault History Files or Views display as the example shown in Figure 17-13.

```
  File  Options  View  Services  Help
 ┌──────────────── Last Accessed Fault History File Entries ────────────────┐
 │                                                                          │ 1 1 80
 │ Enter the number corresponding to one of the following                   │ > CSR
 │ previously accessed fault history file entries and press Enter:          │
 │ __  1.  'IDI.HIST(F00007)'                                               │
 │     2.  'IDI.HIST(F00002)'                                               │
 │     3.                                                                   │
 │     4.                                                                   │
 │     5.                                                                   │
 │     6.                                                                   │ _File_
 │     7.                                                                   │ T
 │     8.                                                                   │ T
 │     9.                                                                   │ T
 │     10.                                                                  │ T
 │                                                                          │ T
 │  F1=Help    F3=Exit   F12=Cancel                                         │ T
 └──────────────────────────────────────────────────────────────────────────┘
*** Bottom of data.


 F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 17-13 Sample last accessed Fault history file or Views display

  b. From the list displayed, type the number corresponding to the desired history file or view name at the initial cursor position and press Enter to display the entries for the selected file or view.
  c. To return to the Fault Entry List display without making any changes, press either PF3 or PF12.
- To display a a list of views available to you, select **File** → **List Views**. This will bring up the panel shown in Figure 17-14.

```
File  Options  View  Services  Help
                          View List
                                                   Row 1 to 9 of 9
Command ===> _____________________________________ Scroll ===> CSR
Line commands: S (select) B (browse).
   Name     Description
   APC      Sample view of APC history files
   BATCH    Batch History Files
   DB2      SAMPLE VIEW OF DB2 HISTORY FILES
   JOHNS    View of CICS and IMS History files
   SUBS     SubSystem History Files
   SW       Test
   TOM      Tom's Views
   VIEW2    TEST VIEW
   ZZZZZZZZ (No description available)
************************* Bottom of data *************************

 F1=Help    F3=Exit    F7=Up      F8=Down    F12=Cancel
F8=Down     F10=Left     F11=Right
```

Figure 17-14 Views list display

- From here, you can enter one of the following line commands against each view:
  - B (browse): This will permit you to browse the view member.

    For example, if If the view member JOHNS was selected for browse, the contents of this member would be displayed as shown in Figure 17-15.

    To return from the View Member display to the View List, press either PF3 or PF12.

```
File  Options  View  Services  Help
                          View List
                       View Member Browse
C  Contents of 'DA.VIEWS(JOHNS)'                    Row 1 to 3 of 3
L  Command ===> ____________________________________ Scroll ===> CSR
   * View of CICS and IMS History files
   CTEST.DUMPA.DACICS.DCAT
   CTEST.DUMPA.DAIMS.DCAT
   ************************* Bottom of data *************************
B

*

 F1=Help    F3=Exit    F7=Up      F8=Down    F12=Cancel
F8=D
```

Figure 17-15 Views member browse display

- S (select): This will select the view for display and automatically return to the previous display with the selected view name specified on the Fault History File or View line.

To display the chosen view, press PF3. To return to the previously displayed history file or view without making any changes, press PF12.

When a different history file or view is selected, the column configuration of the Fault Entry List display might change.

17.2.5 Fault entry list column configuration

The fault information shown on the Fault Entry List display is determined by the HistCols option in effect.

If no HistCols option is used, the default is as illustrated in Figure 17-5 on page 403.

Individual users are able to alter the Fault Entry List display information by either entering the COLS command or by selecting **View** → **Column Configuration**.

This will bring up the Fault Entry List Column Configuration display; an example is shown in Figure 17-16.

```
  File  View  Services  Help
 ____________________________________________________________________________
Fault Entry List Column Configuration                        Line 1 Col 1 80
Command ===> ____________________________________________ Scroll ===> CSR

Current Fault Entry List Column Configuration (Sample Data):

   Fault_ID Job/Tran User_ID  Sys/Job  Abend Date       Time     History_File
__   F00249 IDIVPCOB FRED     MVSA     S0C7  2001/11/22 15:29:03 IDI.HIST

Column Configuration Settings:

{Below, you may change your Fault Entry List display column configuration. To
make a column visible, or to change its relative display position, enter a
non-zero positive value in the Order column; to hide a column, enter 0. The
resulting column configuration will be shown above.}

Order Column
1____ Fault_ID
2____ Job/Tran
3____ User_ID
4____ Sys/Job
 F1=Help      F3=Save      F4=Reset     F6=Actions   F7=Up        F8=Down
F10=Left     F11=Right    F12=Cancel
```

Figure 17-16 Sample Fault Entry List Column Configuration display

The Fault Entry List Column Configuration display is divided into two sections:

- The top portion of the panel is the Current Fault Entry List Column Configuration section, which shows the current column configuration with headings and sample data. This permits you to see which of the selected columns will be visible on the Fault Entry List display without first requiring you to scroll the display horizontally.
- The lower portion is the Column Configuration Settings section, where you can specify which columns are displayed in the Fault Entry List panel.

To make a column visible, or to change its relative display position, enter a non-zero positive value in the Order column; to hide a column, enter 0. Press Enter. The column configuration you specified will be shown in the Current Fault Entry List Column Configuration section.

The Fault_ID column cannot be hidden. If it is not given a specific display position, then it will default to being the first column.

When the Fault Entry List Column Configuration display is first presented, a message is issued which identifies from where the current column configuration was read. There are four possibilities:

- Default column configuration used

  This indicates that a history file, or a view without a valid -HistCols specification, is selected from the Fault Entry List display, and no changes to the default configuration have been saved in the user's ISPF profile.

  The default column configuration is determined by the HistCols option in effect. If changes are made to this configuration, then they will be saved in the user's ISPF profile as the general column configuration. Entering the RESET command (PF4) has no effect.

- General column configuration read from user profile

  This indicates that a history file, or a view without a valid -HistCols specification, is selected from the Fault Entry List display, and a general column configuration exists in the user's ISPF profile. If changes are made to this configuration, then it will replace the general configuration in the user's ISPF profile. Enter the RESET command (PF4) to reset the column configuration to the default.

- Column configuration read from view member *member-name*

  This indicates that a view with a valid -HistCols specification is selected from the Fault Entry List display, and no changes to this configuration have been saved in the user's ISPF profile.

  If changes are made to this configuration, then they will be saved in the user's ISPF profile as a view-specific column configuration.

  Entering the RESET command (PF4) has no effect.

- Specific column configuration for view member-name read from user profile

  This indicates that a view with a valid -HistCols specification is selected from the Fault Entry List display, and a view-specific column configuration exists in the user's ISPF profile.

  If changes are made to this configuration, then it will replace the view-specific configuration in the user's ISPF profile.

  Enter the RESET command (PF4) to reset the column configuration to the -HistCols specification in the view.

The SAVE command (PF3) is used to save the current column configuration in the ISPF profile and return to the Fault Entry List display with the new configuration active. All subsequent interactive Fault Analyzer sessions will use this configuration until it is changed by a subsequent modification, reset to the default using the RESET command, or the ISPF profile is deleted.

The CANCEL command (PF12) can be used to return from the Fault Entry List Column Configuration display without saving any changes made.

Available columns

Figure 17-17, Figure 17-18, and Figure 17-19 display the available columns that can be selected.

```
  File  View  Services  Help
 ------------------------------------------------------------------------
Fault Entry List Column Configuration                        Line 1 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR

Current Fault Entry List Column Configuration (Sample Data):

   Fault_ID Job/Tran User_ID  Sys/Job  Abend Date       Time     History_File_
__   F00249 IDIVPCOB FRED     MVSA     S0C7  2001/11/22 15:29:03 IDI.HIST

Column Configuration Settings:

{Below, you may change your Fault Entry List display column configuration. To
make a column visible, or to change its relative display position, enter a
non-zero positive value in the Order column; to hide a column, enter 0. The
resulting column configuration will be shown above.}

Order Column
1____ Fault_ID
2____ Job/Tran
3____ User_ID
4____ Sys/Job
 F1=Help      F3=Save      F4=Reset     F6=Actions   F7=Up        F8=Down
F10=Left     F11=Right    F12=Cancel
```

Figure 17-17 First screen of the Fault entry list column configuration display

```
  File  View  Services  Help
 ------------------------------------------------------------------------
Fault Entry List Column Configuration                       Line 19 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR
5____ Abend
6____ Date
7____ Time
8____ History_File_DSN
9____ Appl_ID
10___ Job_ID
11___ EXEC_Pgm
12___ IMS_Pgm
13___ Class
14___ Job_Type
15___ Jobname
16___ Program
17___ Stepname
18___ System
0____ Dup_Count
0____ Minidump
0____ Module
0____ MD_Pages
 F1=Help      F3=Save      F4=Reset     F6=Actions   F7=Up        F8=Down
F10=Left     F11=Right    F12=Cancel
```

Figure 17-18 Second screen of the Fault entry list column configuration display

```
  File  View  Services  Help
 ------------------------------------------------------------------------------
Fault Entry List Column Configuration                          Line 29 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR
15___ Jobname
16___ Program
17___ Stepname
18___ System
0____ Dup_Count
0____ Minidump
0____ Module
0____ MD_Pages
0____ MVS_Dump
0____ Netname
0____ Offset
0____ Task
0____ Term_ID
0____ Tran_ID
0____ User_Title
0____ Username

*** Bottom of data.
 F1=Help      F3=Save      F4=Reset     F6=Actions   F7=Up        F8=Down
F10=Left     F11=Right    F12=Cancel
```

Figure 17-19 Third screen of the Fault entry list column configuration display

17.2.6 Matching and selecting fault

You can match faults, so that the display shows only the faults that share a common value for one of the fields (for example, a similar job name, or failing with the same abend code).

This is useful when you are looking for faults with a similar pattern, or if you want to collect entries into a contiguous group so that you can apply a range delete.

The MATCH command is limited to matching the value in one field. However, you can apply a second match to the entries that are displayed, to create a smaller selection, and build a compound match condition.

As you build a compound match condition, the fields you have matched on are displayed in the status line. The value you have matched on is displayed in each entry.

There are three ways of matching: cursor-selecting a matching value, over-typing existing values, or using the MATCH command.

Cursor-selecting a matching value

The first way of matching is to move the cursor under the value you want matched, then press PF4 (MatchCSR). Fault Analyzer refills the fault history window with faults that share this value for this field.

For example, if on the sample screen shown in Figure 17-14 on page 408 you moved the cursor under the Abend value on the last visible entry and pressed PF4, the new display shows only those faults that had an abend of S0CB. The resulting list of entries is shown in Figure 17-20.

```
  File  Options  View  Services  Help
IBM Fault Analyzer - Fault Entry List                          Line 1 Col 1 80
Command ===>                                                 Scroll ===> CSR
MATCH (Abend) [1]
Fault History File or View  : 'IBMUSER.DEMO.HIST'

{The following line commands are available: ? (Query), V (View real-time
report), I (Interactive reanalysis), B (Batch reanalysis), D (Delete).}
                                          [2]
   Fault ID Job/Tran User ID  Sys/Job  Abend Date       Time
__   F00294 DB2NL2   IBMUSER  MVS2     S0CB  2001/02/20 14:42:29
__   F00292 DB2LE2   IBMUSER  MVS2     S0CB  2001/02/20 14:38:25
__   F00049 DACBB001 JCULLEN  MVS2     S0CB  2001/02/01 08:56:27

*** Bottom of data.

 F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 17-20 Fault entry list after one match

Figure 17-20 notes:

[1] A message will be issued that shows the currently selected MATCH columns. This message will only remain until a function key or the Enter key is pressed.

[2] Column headings on which a MATCH is currently active are highlighted. They remain so until the MATCH is reset.

If you now move the cursor under the User ID value on the second entry and press PF4, the new display is as shown in Figure 17-21.

```
  File  Options  View  Services  Help
IBM Fault Analyzer - Fault Entry List                          Line 1 Col 1 80
Command ===>                                                 Scroll ===> CSR
MATCH (Abend User_ID)
Fault History File or View  : 'IBMUSER.DEMO.HIST'

{The following line commands are available: ? (Query), V (View real-time
report), I (Interactive reanalysis), B (Batch reanalysis), D (Delete).}

   Fault ID Job/Tran User ID  Sys/Job  Abend Date       Time
__   F00294 DB2NL2   IBMUSER  MVS2     S0CB  2001/02/20 14:42:29
__   F00292 DB2LE2   IBMUSER  MVS2     S0CB  2001/02/20 14:38:25

*** Bottom of data.

 F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 17-21 Fault Entry List after two matches

Matching is restrictive. If you apply a second match, the selection of faults is restricted to those faults that have already satisfied the first match.

For example, if you match by a userid, and then match by a dump status, the resultant display shows only those entries for one owner with a particular status.

If instead, you just matched by status, the display shows all the entries with this status for all user IDs.

Over-typing existing values

The second way of matching is by over-typing existing values.

For example, if the Abend column contains the value S0C4 for a fault entry, then by over-typing the 4 with a 1, making the value S0C1, and pressing the Enter key, a match will be performed to show only those fault entries that have an abend value of S0C1.

Wildcard characters (*) can be used to specify generic match values.

If nothing other than a single wildcard character is specified, then any column value, except n/a, will match. This can be used to select all fault entries for which a value exists in the particular column. Any number of values can be over-typed before pressing the Enter key.

However, if values for the same column are over-typed on multiple rows, then only the last over-typed value for that column is used.

Over-typing an existing value is particularly useful when you want to match on values that are similar to ones already displayed. By simply changing the displayed value to the desired target, and pressing the Enter key, a MATCH is performed with a minimum of typing required.

Using the MATCH command

The third way of matching is by entering the MATCH command. The syntax of this command is shown in Figure 17-22.

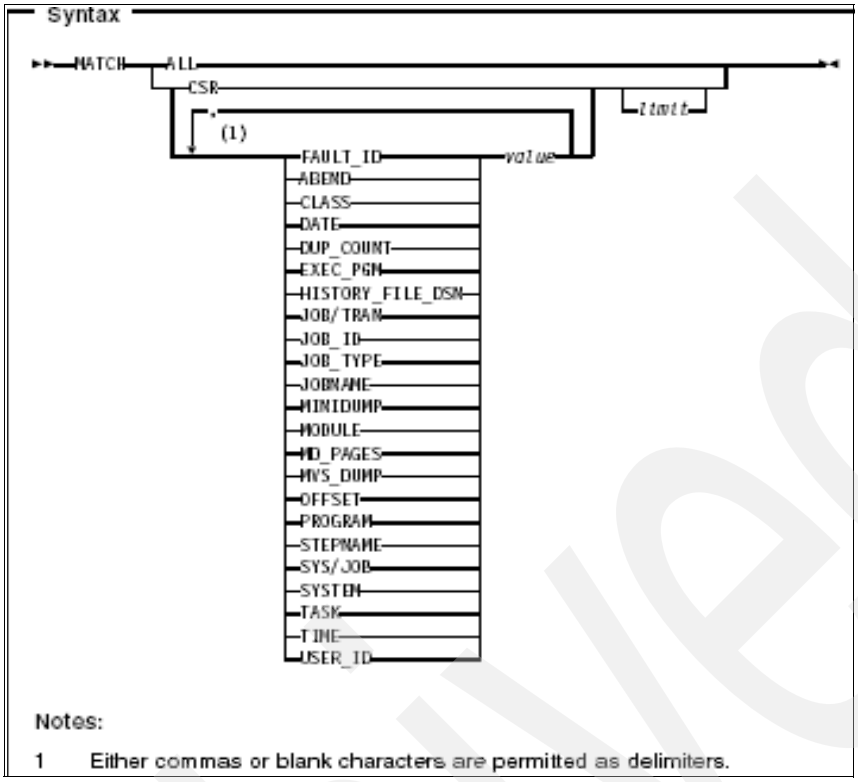


Figure 17-22 Match command syntax

MATCH ALL (which is the same as PF12) removes all match conditions including the limit, so the display shows the same entries it had when you first displayed the fault history file.

This is not the same as doing a REFRESH, which looks at the fault history file, and so may display new entries that have been written to the fault history file since you first started Fault Analyzer.

When you refresh, you also remove all match conditions.

MATCH CSR is the cursor match. To make this work you have to move the cursor under a value and press Enter, which is essentially the same as placing the cursor and pressing PF4.

The other keywords correspond to fields, and you follow the field name with a value. When matching, values are case insensitive (Fault Analyzer turns all values into upper case, and matches them with the upper case values displayed in the fault history file). An asterisk (*) can be used as a wildcard. When you append it to a value, Fault Analyzer matches all values starting with the value you entered before the *. Since all values are strings, you could, for example, enter:

```
MATCH DATE 2000/07*
```

This would display all entries for the month of July, for the year 2000.

If nothing other than a single wildcard character is specified, then any column value, except n/a, will match. This can be used to select all fault entries for which a value exists in the particular column.

The *limit* is the maximum number of entries that will be displayed. This is particularly useful if you just want to find the first entry with a particular value; set the limit to 1.

You do not have to be able to see a value to enter it as part of a MATCH command. That is, you can MATCH on column values that are in the visible area of the screen, as well as column values that are outside of the current scroll window.

You can also MATCH on columns that are not currently selected for display. If you apply a match value, and no entries satisfy this value, then Fault Analyzer displays the message `No matches`, and does not change the entries that are currently displayed. This means you cannot get a display with no entries in it.

You can also enter the REFRESH command (by typing it or by selecting **View** → **Refresh**). This command refreshes the display from the fault history file. The new display includes any faults recorded since you entered the fault history file or most recently performed a REFRESH, and all matching is removed.

A REFRESH is automatically performed when the Enter key is pressed from the Fault Entry List display and no other commands, selections, or over-typing is being performed, and no MATCH command is currently active.

17.2.7 Applying an action against a particular fault

You can apply an action to a particular fault by entering a line command against the entry. The available actions are:

B Batch reanalysis
Submit a batch job to perform reanalysis against the selected fault entry. The analysis report is written to SYSPRINT.

D Delete
Delete the fault entry from the fault history file. After you delete an entry, it is immediately removed from the Fault Entry List display, and is not displayed by any subsequent refresh.

I Interactive reanalysis
Run interactive reanalysis against the selected fault. After a little while, the interactive report is displayed. The interactive report does not replace the real-time analysis report.

V (or **S**) View report
View the dump analysis report about the fault entry that was generated in real time.

? View fault entry information
View the fault entry information. In particular, this shows the associated MVS dump data set name, if there is one.

If you enter a line command against an entry, and Fault Analyzer is unable to complete the command (for example, you attempt to run a batch dump reanalysis against a fault that has no associated dump data set, or the dump data set is unavailable), then the line command is not cleared from the line.

You can type line commands against many entries before you press Enter. In this case, Fault Analyzer attempts to honor each command, starting with the entry at the top. If Fault Analyzer is unable to honor a command, it stops processing. It clears the line commands from each entry it was able to process, but leaves the line commands for each entry it failed to process or the entry at which it could not honor the command.

17.2.8 History file properties

To display attributes and statistics for the currently selected fault history file, select **File** → **Fault History File Properties**. This will bring up the Fault History File Properties display as shown in Figure 17-23 and Figure 17-24.

```
  File  Options  View  Services  Help
 ┌──────────────────────────────────────────────────────────────────────────
 │ 6 _1. Last Accessed Fault History Files or Views...  │      Line 1 Col 1 80
 │    2. Last Accessed Fault History File Entries...    │     Scroll ===> CSR
 │    3. Clear Last Accessed Information                │
 │    4. List Views...                                  │
 │    5. Analyze MVS Dump Data Set...                   │
 │    6. Fault History File Properties...               │  (View real-time
 │    7. Exit Fault Analyzer                            │ ), D (Delete).}
 └──────────────────────────────────────────────────────┘
   Fault_ID Job/Tran User_ID  Sys/Job  Abend Date       Time     History_File_
     F00007 JMONITOR n/a      STLABF6  U4093 2004/10/29 07:34:48 IDI.HIST
     F00006 CONOVERY CONOVER  STLABF6  SB37  2004/10/25 09:47:46 IDI.HIST
     F00005 CONOVERF CONOVER  STLABF6  S0CB  2004/10/21 11:05:18 IDI.HIST
     F00004 TIMOTHY  TIMOTHY  STLABF6  n/a   2004/10/11 16:21:13 IDI.HIST
     F00003 TIMOTHY  TIMOTHY  STLABF6  n/a   2004/10/11 14:31:39 IDI.HIST
     F00002 ZHONG    ZHONG    STLABF6  n/a   2004/10/11 10:53:17 IDI.HIST

*** Bottom of data.


 F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 17-23 Selection #6 from File option

```
                         Fault History File Properties
 Attributes and Statistics                                  Line 1 Col 1 76
 Command ===> _____________________________________________ Scroll ===> CSR

 Enter the Exit command (PF3) to return to the fault history file display.

 Fault History File Type . . : PDSE (Library)
 Fault ID Prefix . . . . . . : F
 Maximum Fault Number. . . . : n/a
 Overwrite Existing Fault
   Entries . . . . . . . . . : No
 Total Number of Entries . . : 6
 Number of Entries With
   Minidump. . . . . . . . . : 6 (100.00% of Total)
 Maximum Minidump Size . . . : 150 Pages
 Minimum Minidump Size . . . : 53 Pages
 Average Minidump Size . . . : 102 Pages
 Number of Entries With
   Associated MVS Dump . . . : 1 (16.67% of Total)

 *** Bottom of data.
  F1=Help     F3=Exit     F5=RptFind  F7=Up       F8=Down    F10=Left
 F11=Right
```

Figure 17-24 Fault history file properties

17.2.9 Resetting history file access information

To reset all information about previously accessed fault history files or views, and previously accessed fault history file entries, select **File** → **Clear Last Accessed Information**.

Immediately after selecting this option, no entries are available when selecting the Last Accessed Fault History File Entries option. However, the Last Accessed Fault History Files or Views option will show a single entry for the currently active history file or view.

17.2.10 Refreshing fault entry information

While displaying a history file or view, it is possible that new entries are being added, for example due to real-time analysis of abending jobs.

To re-read the history file or view to include any such entries, you can either issue the REFRESH command or select **View** → **Refresh**.

The refresh will cause the display to be reformatted and scrolled to the top line and left-most column. Any MATCH command filtering that was active at the time of the refresh will be reset.

17.2.11 Actions command

The ACTIONS ISPF command (by default mapped to PF6) can be used to place the cursor at the left-most action available.

Depending on ISPF settings, you might then be able to move the cursor to other actions by pressing the Tab key. Alternatively, you can simply use the up, down, left, and right arrow keys to place the cursor on the action of your choice.

Using a PF key to issue the ACTIONS command is advantageous because the cursor will be automatically repositioned in the display at the location where it was before the action was selected. Once the cursor is placed on an action-bar item, press Enter to show the associated pull-down menu.

17.2.12 Action bar pull-down menus

Most of the displays used by the Fault Analyzer ISPF interface include an action bar located at the top of the panel. The selectable actions on the Fault Entry List display action bar are File, Options, View, Services, and Help. Selecting any of these items results in the display of a pull-down menu of additional options. The options provided for the main menu choices are illustrated in Figure 17-25 through Figure 17-29.

Use the following procedures to access any of the main menu and drop-down menu items:

- To display any of the pull-down menus, place your cursor on the action bar item of interest and press Enter.
- Options in each pull-down menu can be selected by entering the associated option number at the initial cursor position *or* placing the cursor (using the up, down, left, and right arrow keys) anywhere on the line of the option; and pressing Enter. Any options not available for selection are indicated by an asterisk (*) instead of a numerical option number.

```
  File  Options  View  Services  Help
 ┌──────────────────────────────────────────────────┐ ──────────────────────────────
 │ _  1. Last Accessed Fault History Files or Views...│        Line 1 Col 1 80
 │    2. Last Accessed Fault History File Entries...  │       Scroll ===> CSR
 │    3. Clear Last Accessed Information              │
 │    4. List Views...                                │
 │    5. Analyze MVS Dump Data Set...                 │
 │    6. Fault History File Properties...             │  (View real-time
 │    7. Exit Fault Analyzer                          │ ), D (Delete).}
 └──────────────────────────────────────────────────┘

   Fault_ID Job/Tran User_ID  Sys/Job  Abend Date       Time     History_File_
     F00007 JMONITOR n/a      STLABF6  U4093 2004/10/29 07:34:48 IDI.HIST
     F00006 CONOVERY CONOVER  STLABF6  SB37  2004/10/25 09:47:46 IDI.HIST
     F00005 CONOVERF CONOVER  STLABF6  S0CB  2004/10/21 11:05:18 IDI.HIST
     F00004 TIMOTHY  TIMOTHY  STLABF6  n/a   2004/10/11 16:21:13 IDI.HIST
     F00003 TIMOTHY  TIMOTHY  STLABF6  n/a   2004/10/11 14:31:39 IDI.HIST
     F00002 ZHONG    ZHONG    STLABF6  n/a   2004/10/11 10:53:17 IDI.HIST

*** Bottom of data.


 F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 17-25 File Entry List display File menu

```
  File  Options  View  Services  Help
 ──────────────────────────────────────────────────────────────────────────────
IBM F  ┌──────────────────────────────────┐                Line 1 Col 1 80
Comma  │ _  1. Fault Analyzer Preferences...│               Scroll ===> CSR
       │    2. Batch Reanalysis Options...  │
       │    3. Interactive Reanalysis Options...
Fault  └──────────────────────────────────┘

{The following line commands are available: ? (Query), V (View real-time
report), I (Interactive reanalysis), B (Batch reanalysis), D (Delete).}

   Fault_ID Job/Tran User_ID  Sys/Job  Abend Date       Time     History_File_
     F00007 JMONITOR n/a      STLABF6  U4093 2004/10/29 07:34:48 IDI.HIST
     F00006 CONOVERY CONOVER  STLABF6  SB37  2004/10/25 09:47:46 IDI.HIST
     F00005 CONOVERF CONOVER  STLABF6  S0CB  2004/10/21 11:05:18 IDI.HIST
     F00004 TIMOTHY  TIMOTHY  STLABF6  n/a   2004/10/11 16:21:13 IDI.HIST
     F00003 TIMOTHY  TIMOTHY  STLABF6  n/a   2004/10/11 14:31:39 IDI.HIST
     F00002 ZHONG    ZHONG    STLABF6  n/a   2004/10/11 10:53:17 IDI.HIST

*** Bottom of data.


 F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 17-26 Fault entry list Option menu

```
  File  Options  View  Services  Help
IBM Fault Anal  _  *. Add Blank Lines                    Line 1 Col 1 80
Command ===>       2. Remove Blank Lines                Scroll ===> CSR
                   *. Add Help Text
Fault History      4. Remove Help Text
                   *. Add Detail Information
{The following     6. Remove Detail Information        V (View real-time
report), I (In     7. Preferred formatting Width...    s), D (Delete).}
                   8. Column Configuration...
   Fault_ID Jo     9. Refresh                           Time     History_File_
     F00007 JM                                          07:34:48 IDI.HIST
     F00006 CONOVERY CONOVER  STLABF6  SB37  2004/10/25 09:47:46 IDI.HIST
     F00005 CONOVERF CONOVER  STLABF6  S0CB  2004/10/21 11:05:18 IDI.HIST
     F00004 TIMOTHY  TIMOTHY  STLABF6  n/a   2004/10/11 16:21:13 IDI.HIST
     F00003 TIMOTHY  TIMOTHY  STLABF6  n/a   2004/10/11 14:31:39 IDI.HIST
     F00002 ZHONG    ZHONG    STLABF6  n/a   2004/10/11 10:53:17 IDI.HIST

*** Bottom of data.

 F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 17-27 Fault entry list View menu

```
IBM Fault Analyzer -  _  1. Message ID Look-Up...                 1 Col 1 80
Command ===>          2. Copy Current Display to Data Set...  1 ===> CSR

Fault History File or View  : 'IDI.HIST.TEST'

{The following line commands are available: ? (Query), V (View real-time repor
D (Delete).}

   Fault_ID System   Job/Tran Abend Date       Time     User_ID  History_File_
     F00001 STLABF6  $TSTFACB S0C7  2004/11/03 13:14:50 GRACINE  IDI.HIST.TEST

*** Bottom of data.

 F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 17-28 Fault entry list Services menu

```
 File  Options  View  Services  Help
IBM Fault Analyzer - Fau  _  1. Fault Analyzer User's Guide and Reference...
Command ===>              _  2. About Fault Analyzer...

Fault History File or View  : 'IDI.HIST.TEST'

{The following line commands are available: ? (Query), V (View real-time repor
D (Delete).}

   Fault_ID System   Job/Tran Abend Date       Time     User_ID  History_File_
     F00001 STLABF6  $TSTFACB S0C7  2004/11/03 13:14:50 GRACINE  IDI.HIST.TEST

*** Bottom of data.

 F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 17-29 Fault entry list Help menu

18

Fault Analyzer interactive reanalysis

Interactive reanalysis provides several advantages over batch reanalysis, including the following:

- The sections of the report that are of interest can be selected and examined separately.
- Any storage area that is included in the associated minidump or SYSMDUMP can be displayed, regardless of whether it is included in the Fault Analyzer report.
- Source code information (if provided via compiler listing or side file) can be viewed in its entirety.
- This is the only way to analyze CICS system abends.

This chapter describes how to use FA to perform interactive reanalysis.

18.1 Interactive reanalysis options

To specify general interactive reanalysis options that apply to your interactive reanalysis sessions only, select **Options** → **Interactive Reanalysis Options** from the Fault Entry List display as shown in This will bring up the Interactive Reanalysis Options display as shown in Figure 18-2.

```
  File  Options  View  Services  Help
 ──────┌──────────────────────────────────────┐──────────────────────────────
IBM F  │ 3 _1. Fault Analyzer Preferences...  │            Line 1 Col 1 80
Comma  │    2. Batch Reanalysis Options...    │           Scroll ===> CSR
       │    3. Interactive Reanalysis Options...│
Fault  └──────────────────────────────────────┘ INE BATCH HISTORY FILES

{The following line commands are available: ? (Query), V (View real-time repor
D (Delete).}

   Fault_ID History_File_DSN                          System   Job/Tran Abe
     F00011 IDI.HIST                                  STLABF6  FBIIDRUN S52
     F00002 IDI.HIST.TEST                             STLABF6  $TSTFACB S0C
     F00001 IDI.HIST.TEST                             STLABF6  $TSTFACB S0C
     F00010 IDI.HIST                                  STLABF6  IMWEBF6  S42
     F00009 IDI.HIST                                  STLABF6  IMWEBF6  U40
     F00008 IDI.HIST                                  STLABF6  IMWEBF6  U40
     F00007 IDI.HIST                                  STLABF6  JMONITOR U40
     F00006 IDI.HIST                                  STLABF6  CONOVERY SB3
     F00005 IDI.HIST                                  STLABF6  CONOVERF S0C
     F00004 IDI.HIST                                  STLABF6  TIMOTHY  n/a
     F00003 IDI.HIST                                  STLABF6  TIMOTHY  n/a
 F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 18-1 Interactive reanalysis options display

The following settings can be specified using this display:

- Options line for interactive reanalysis

  You can specify options here that will apply to all interactive reanalysis sessions you initiate.

  These options, which are the equivalent of the PARM field options used by batch reanalysis jobs, take precedence over any options specified through an options file (Options data set name).

- Redisplay this panel before each reanalysis

  If this option is set to Y, then the Interactive Reanalysis Options display will be shown each time an interactive reanalysis is requested. Make any necessary changes and press PF3 (to continue with the current options) or PF12 (to undo any changes made). The result is determined by how the next field is set: if the Display panel to alter allocated data sets option is set to N, then the Interactive Reanalysis Options display will not be shown.

- Display panel to alter allocated data sets

  If this option is set to Y, you will be presented with an ISPF EDIT display screen of the pseudo JCL stream generated by Fault Analyzer. This edit screen is shown in Figure 18-3.

```
  File  Options  View  Services  Help
 ┌──────────────────────── Interactive Reanalysis Options ────────────────────────┐
I │                                                                                │ 0
C │ Press PF3 to save options or PF12 to cancel.                                   │
  │                                                                                │
F │ General Options:                                                               │
  │  Options line for interactive reanalysis:                                      │
{ │  LA(ENU)                                                                       │ r
D │  Redisplay this panel before each reanalysis Y  (Y/N)                          │
  │  Display panel to alter allocated data sets  Y  (Y/N)                          │
  │  Prompt before opening a SYSMDUMP  . . . . . Y  (Y/N)                          │ e
I │                                                                                │ C
  │ Reanalysis Options Data Set Control:                                           │ C
  │  Options data set name 'GRACINE.SIDIOPTS'                                      │
* │  Options member name   OPTS1___                                                │
  │  Use this data set during reanalysis . . . . Y  (Y/N)                          │
  │  Edit the options data set before reanalysis Y  (Y/N)                          │
  │                                                                                │
  │  F1=Help    F3=Exit   F12=Cancel                                               │
  └────────────────────────────────────────────────────────────────────────────────┘

 F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 18-2 Interactive reanalysis options display

Making any necessary changes in accordance with the instructions displayed, then enter the Exit (PF3) or Cancel (PF12) command as appropriate to initiate the interactive reanalysis. If this option is set to N, then the interactive reanalysis commences without first displaying the pseudo JCL EDIT screen shown in Figure 18-3.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ──────────────────────────────────────────────────────────────────────────────
EDIT       GRACINE.SIDIOPTS(OPTS1) - 01.00                Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> PAGE
****** ***************************** Top of Data ******************************
000001 DATASETS(IDIEXEC(&SYSUID..SIDIEXEC))
000002 EXIT(LISTING(REXX(LX)))
****** **************************** Bottom of Data ****************************

 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 18-3 Options file EDIT for interactive reanalysis

- Prompt before opening a SYSMDUMP

  If this field is set to Y and, during the interactive reanalysis or as a result of displaying storage locations from within the interactive report, access is required to a storage location that is not contained in the saved minidump, a display is shown before opening an associated SYSMDUMP or SVC dump data set to look for the missing storage. An example of this display is shown in Figure 18-4.

```
 File  Options  View  Services  Help
 ------------------------- Confirm SYSMDUMP Open -------------------------
 Command ===> ____________________________________________________________

 Fault Analyzer has determined the need to open SYSMDUMP data set:
 JKATNIC.CICS53.SOS.DUMP

 Permitting this might cause delays, however, if the open is not permitted,
 Fault Analyzer cannot access important storage information.

 Press Enter to confirm the data set open.

 Press Cancel or Exit to cancel the data set open and attempt to continue
 without access to missing storage locations.

  F1=Help     F3=Exit    F12=Cancel

     F00286 CICS53   n/a      MVS2     S122  2001/05/22 10:49:44
     F00325 DAAMB022 n/a      MVS2     S0C6  2001/04/27 11:03:48
     F00111 CICS53   n/a      MVS2     S08E  2001/03/22 13:12:23
     F00272 CICS53   n/a      MVS2     S08E  2001/03/22 13:12:23
i    F00328 CICS53   n/a      MVS2     S08E  2001/03/22 13:12:23
 F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 18-4 Confirm SYSMDUMP open display

You will only be prompted, at most, once during any interactive reanalysis session. If the open is cancelled by entering CANCEL or EXIT, no further attempts will be made to open the SYSMDUMP data set. Likewise, if the open is confirmed, Fault Analyzer will check the SYSMDUMP for all references to storage locations not contained in the minidump. If this field is set to N, then the associated dump data set will be opened if required without prompting you first.

- Options data set name

  This field can optionally specify the name of a PDS(E) data set in which a member contains Fault Analyzer options. The data set and member name will be used as the IDIOPTS user options file. This data set can, for example, be used if more options than will fit on the options line at the top of this display are required.

  Note: On the user options file shown in Figure 18-2:

  - The options data set will only be used if the Use this data set during reanalysis option is set to Y.
  - Options specified on the options line take precedence over options specified in this data set.

- Options member name

  This is the member name of the data set specified in ‘Options data set name’.

- Use this data set during reanalysis

  If this option is set to Y, then the data set and member name specified previously will be used by Fault Analyzer during the interactive reanalysis. If it is set to N, then the data set and member name will not be used.

- Edit the options data set before reanalysis

  If this field is set to Y, then an ISPF EDIT display screen of the member in the options data set specified previously is presented prior to commencing the interactive reanalysis.

Having made your changes to the options data set (if any), enter the Exit command (usually mapped to PF3).

18.2 Initiating interactive reanalysis

To initiate interactive reanalysis, enter **I** against the fault history entry.

The interactive reanalysis report looks similar to the real-time fault analysis report, but has more functions that let you look further into the cause of the problem.

Figure 18-5 shows an example of the first interactive report display, from which all other parts of the interactive report can be selected.

```
  File  View  Services  Help
 ____________________________________________________________________________
 Interactive Reanalysis Report                                Line 1 Col 1 80
 Command ===> ____________________________________________ Scroll ===> CSR_
 JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7          STLABF6   2004/11/08  21:58:09

 Fault Summary:
 Module IDISCBL1, program IDISCBL1, source line # 31 : Abend S0C7 (Data Excepti

 Select one of the following options and press Enter to access further fault in
 _  1.  Synopsis
    2.  Event Summary
    3.  System-Wide Information
    4.  Abend Job Information
    5.  Options in Effect

 {Fault Analyzer maximum storage allocated: 1.43 megabytes.}

 *** Bottom of data.


  F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
  F8=Down     F10=Left     F11=Right
```

Figure 18-5 Interactive reanalysis report display

A fault summary is provided at the top of the initial display which is equivalent to the summary provided in message IDI0002I that is issued during the real-time analysis of any fault.

The individual options that can be selected from the initial display are explained in the sections that follow.

Options that are available are cursor tab-capable and shown with their option number. Options that are not available cannot be tabbed to.

If the option to show help text is selected, then information about the maximum amount of allocated storage that Fault Analyzer used during the analysis will be included at the bottom of the display. This amount of storage is for explicit allocations only and does not include storage for loaded modules, and so forth.

18.3 General information about the interactive report

The interactive report will be formatted differently depending on the logical screen size used. All examples in this book are based on a screen size which is 24 lines by 80 columns; however, if your screen is larger, Fault Analyzer will format the report accordingly.

This is true also if the screen size is dynamically resized. Just press the Enter key and the report section viewed will be reformatted to match the screen size. Anywhere in the interactive report, you can use the Up (PF7), Down (PF8), Left (PF10), and Right (PF11) commands to scroll the display so you can see the entire report section that is currently selected.

Note: PF10 and PF11 in the Dump Storage display are instead mapped to the Prev and Next commands respectively, since this display does not necessitate horizontal scrolling.

Throughout the interactive report are tab-capable fields. These are point-and-shoot fields which respond to placing the cursor on them and pressing the Enter key. What is displayed next depends on the type of information selected.

Some will take you to other parts of the report while others will display detailed information about the item selected.

For example:

- Source code line or statement number

  Displays the source code for the entire program as obtained from the compiler listing or side file with the selected line or statement number highlighted. In addition, disassembly of machine instructions is provided.

- Storage address

  Displays the storage at this location in both hexadecimal and translated EBCDIC.

Although the point-and-shoot fields are defined using the ISPF color attribute YELLOW, they might actually be displayed with a different color depending on user settings.

18.4 Exit from the interactive report

When you exit from the interactive report using an Exit or Cancel command from the Interactive Reanalysis Report display, you are presented with a confirmation prompt.

To abort the exit and return the interactive reanalysis report, press PF12.

The exit prompt panel is only displayed if the interactive reanalysis elapsed time exceeds, or is equal to, the number of seconds in effect for the Interactive ExitPromptSeconds option. Using the ISPF jump command will exit the interactive report.

18.5 Synopsis

Selecting option **1** from the initial interactive report display results in the display of the Synopsis section of the report as shown in Figure 18-6.

```
  File  View  Services  Help
 ------------------------------------------------------------------------
Synopsis                                                   Line 1 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR
JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7            STLABF6   2004/11/08  21:58:09

A system abend 0C7 occurred in module IDISCBL1 program IDISCBL1 at offset X'42

A program interruption code 0007 (Data Exception) is associated with this aben

  A decimal digit or sign was invalid.

The cause of the failure was program IDISCBL1 in module IDISCBL1.  The COBOL s
was:

  Source
  Line #
  000029         CLEAR SECTION.
  000030         START001.
  000031             DIVIDE NUMBERX BY ERROR-COUNT GIVING BAD-RESULT.

The COBOL source code for data fields involved in the failure:
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right
```

Figure 18-6 Synopsis display

18.6 Event summary

Select option **2** from the initial interactive report display to view the Event Summary section of the report as shown in Figure 18-7.

```
  File  View  Services  Help
 ------------------------------------------------------------------------
Event Summary                                              Line 1 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR
JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7            STLABF6   2004/11/08  21:58:09

{The following events are presented in chronological order.}

Event            Fail  Module   Program  EP
#  Type          Point Name     Name     Name     Event Location (*)      Loaded
 1 Abend S0C7    ***** IDISCBL1 IDISCBL1 IDISCBL1 L#31 P+422              SYS043
 2 Call                CEEPLPKA n/a      CEEHDSP  E+3986                  CEE.SC
 3 Abend U4039         CEEPLPKA n/a      CEEHSDMP E+D4                    CEE.SC


(*) One or more of the following abbreviations might appear in the "Event Loca

    F#n  Source file number (refer to detailed event information for file iden
    L#n  Source file line number
    S#n  Listing file statement number (refer to detailed event information fo
    M+x  Offset from start of load module
    P+x  Offset from start of program
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right
```

Figure 18-7 Event summary display

Individual events can be selected from this summary by placing the cursor on the event number and pressing Enter. The detailed event display that is presented if you do this is described in the next section.

Point-and-shoot fields are provided for most of the information in the Event Location column:

- If selecting offset-type information (M+x, P+x, or E+x), the Dump Storage display is presented for the corresponding address.
- If selecting source or listing information (L#n or S#n), the Compiler Listing display is presented for the appropriate line or statement.

18.6.1 Detailed event information

An Event Details display is presented when an event is selected from the Event Summary display. The next five figures, Figure 18-8 through Figure 18-12 present an example.

```
  File  View  Services  Help
 ------------------------------------------------------------------------
Event 1 of 3: Abend S0C7 *** Point of Failure ***                Top of data
Command ===> _____________________________________________ Scroll ===> CSR
JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7           STLABF6   2004/11/08  21:58:09


Abend Code. . . . . . . . . : S0C7
Program Interruption Code . : 0007 (Data Exception)
  A decimal digit or sign was invalid.

COBOL Source Code:
  Source
  Line #
  000029         CLEAR SECTION.
  000030         START001.
  000031             DIVIDE NUMBERX BY ERROR-COUNT GIVING BAD-RESULT.

Data Field Declarations:
  Source
  Line #
  000011         01  NUMBERX PIC 999999 COMP-3.
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right
```

Figure 18-8 Event details display for point of failure (part 1 of 5)

```
  File  View  Services  Help
 ------------------------------------------------------------------------
Event 1 of 3: Abend S0C7 *** Point of Failure ***            Line 18 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR
JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7           STLABF6   2004/11/08  21:58:09
  000013             05  ERROR-COUNT PIC 999999 COMP-3.
  000016         01  BAD-RESULT PIC 99 COMP-3.

Data Field Values:
  BAD-RESULT     = X'0000'
  ERROR-COUNT    = X'C1C2C3C4'  *** Cause of error ***
  NUMBERX        = 986888

The listing or side file used for the above was found in GRACINE.IDI.IVPCB.LIS

Load Module Name. . . . . . : SYS04313.T215804.RA000.$TSTFACB.GOSET.H01(IDISCB
  At Address. . . . . . . . : 0BF00D30
  Load Module Length. . . . : X'12D0'
  Link-Edit Date and Time . : 2004/11/08  21:58:06

Program and Entry Point Name: IDISCBL1
  At Address. . . . . . . . : 0BF00D30 (Module IDISCBL1 offset X'0')
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right
```

Figure 18-9 Event details display for point of failure (Part 2 of 5)

```
  File  View  Services  Help
 ------------------------------------------------------------------------
Event 1 of 3: Abend S0C7 *** Point of Failure ***            Line 35 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR
JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7           STLABF6   2004/11/08  21:58:09
  Program Length. . . . . . : X'688'
  Program Language. . . . . : COBOL (Compiled using IBM Enterprise COBOL for z

Machine Instruction . . . . : FD73D108D0F8  DP    264(8,R13),248(4,R13)
  At Address. . . . . . . . : 0BF01152 (Program IDISCBL1 offset X'422')
  AMODE . . . . . . . . . . : 31
  Failing Operand . . . . . : Second operand
  First Operand Address . . : 0C151138 (8466120 bytes of storage addressable)
  First Operand Length. . . : 8
  First Operand Storage . . : 00000000 0986888C   *.....fh.*
  Second Operand Address. . : 0C151128 (8466136 bytes of storage addressable)
  Second Operand Length . . : 4
  Second Operand Storage. . : C1C2C3CF   *ABC.*

Program Status Word (PSW) . : 078D2000 8BF01158

General Purpose Registers:
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right
```

Figure 18-10 Event details display for point of failure (Part 3 of 5)

```
  File  View  Services  Help
 ------------------------------------------------------------------------
Event 1 of 3: Abend S0C7 *** Point of Failure ***            Line 52 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR
JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7           STLABF6   2004/11/08  21:58:09
  R0:  0C151140 (8466112 bytes of storage addressable)
  R1:  0BF00F55 (Module IDISCBL1 program IDISCBL1 + X'225')
  R2:  000167FC (51204 bytes of storage addressable)
  R3:  0BF0111C (Module IDISCBL1 program IDISCBL1 + X'3EC', source line # 26 )
  R4:  0BF00D68 (Module IDISCBL1 program IDISCBL1 + X'38')
  R5:  0BF0BBC8 (10847288 bytes of storage addressable)
  R6:  00000000 (2048 bytes of storage addressable)
  R7:  00000000 (2048 bytes of storage addressable)
  R8:  0C1710D0 (Module IDISCBL1 program IDISCBL1 WORKING-STORAGE SECTION BLW=
  R9:  0C14D448 (8481720 bytes of storage addressable)
  R10: 0BF00E5C (Module IDISCBL1 program IDISCBL1 + X'12C')
  R11: 0BF00F80 (Module IDISCBL1 program IDISCBL1 + X'250')
  R12: 0BF00E2C (Module IDISCBL1 program IDISCBL1 + X'FC')
  R13: 0C151030 (8466384 bytes of storage addressable)
  R14: 8BF010CC (Module IDISCBL1 program IDISCBL1 + X'39C', source line # 23 )
  R15: 8C11ED70 (Module IGZCPAC + X'43480')

 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right
```

Figure 18-11 Event details display for point of failure (Part 4 of 5)

```
  File  View  Services  Help
 ------------------------------------------------------------------------
Event 1 of 3: Abend S0C7 *** Point of Failure ***            Line 62 Col 1 80
Command ===> ___________________________________________ Scroll ===> CSR
JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7           STLABF6   2004/11/08  21:58:09
  R10: 0BF00E5C (Module IDISCBL1 program IDISCBL1 + X'12C')
  R11: 0BF00F80 (Module IDISCBL1 program IDISCBL1 + X'250')
  R12: 0BF00E2C (Module IDISCBL1 program IDISCBL1 + X'FC')
  R13: 0C151030 (8466384 bytes of storage addressable)
  R14: 8BF010CC (Module IDISCBL1 program IDISCBL1 + X'39C', source line # 23 )
  R15: 8C11ED70 (Module IGZCPAC + X'43480')

Associated Messages

CEE3207S Job-specific text not available

Associated Storage Areas


Next Event Details

*** Bottom of data.
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right
```

Figure 18-12 Event details display for point of failure (Part 5 of 5)

All information associated with the currently selected event is either already included in the displayed information, or a point-and-shoot link to the information is provided.

Such links include message and abend codes, for which an explanation can be provided if selected, and associated storage areas.

To select the previous or next event from the one currently displayed, point-and-shoot links are provided at the bottom of the current display. Simply place the cursor on one of these and press Enter.

18.7 System-wide information

Select option **3** from the initial interactive report display to view the System-Wide Information section of the report as shown in Figure 18-13.

```
  File  View  Services  Help
System-Wide Information                                        Line 1 Col 1 80
Command ===> ____________________________________________ Scroll ===> CSR
TRANID: CD01      CICS ABEND: DSNC                CICS41   2003/04/28 13:22:19

Open Files

CICS Information

DB2 Information

Storage Areas

Java Information

WebSphere Information

Language Environment Heap Analysis

*** Bottom of data.

 F1=Help      F3=Exit      F5=Rpt Find   F6=Actions   F7=Up        F8=Down
F10=Left     F11=Right
```

Figure 18-13 System-Wide information display

The System-Wide Information display contains links to additional sections of the report.

Point-and-shoot links that are available for selection are shown. Placing the cursor on one of these links and pressing Enter displays the detailed information.

The following types of information are available from here (if applicable to the fault analyzed):

- CICS Information

  This provides point-and-shoot links to CICS information, such as the following.

- CICS Trace Table

  Selecting the CICS Trace Table link will provide you with a display of the CICS trace table. An example is shown in Figure 18-14.

```
  File  View  Services  Help
CICS Information                                                Line 1 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR
TRANID: TRAR      CICS ABEND: AZI4                      2004/11/09  13:07:13

CICS Control Blocks

CICS Transaction Storage

Last CICS 3270 Screen Buffer--not available

CICS Trace Table--not available

CICS Trace Formatting

CICS Recovery Manager

*** Bottom of data.


 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right
```

Figure 18-14 CICS Information display

The standard CICS trace table is enhanced by Fault Analyzer for greater ease of use. Information is added to indicate the call point origin addresses, including the module name, CSECT name, offset within CSECT, and source line or listing statement number if available.

As in all other sections of the interactive report, source line numbers or listing statement numbers can be selected by placing the cursor on them and pressing Enter. This will provide a full source listing

- Last CICS 3270 Screen Buffer

  Selecting the Last CICS 3270 Screen Buffer link will provide you with a display of the last 3270 screen buffer written by CICS.

  All non-printable characters are shown as periods (.). It is not always possible for Fault Analyzer to provide CICS 3270 screen buffer information. For example, if there are indications of VTAM session errors on the terminal, or if one of the following CICS abends has occurred (* indicates a wildcard character), then no 3270 screen buffer information will be provided:

  - ATC*
  - AZI*
  - AZTS
  - ATNI
  - AZCT
  - AKCT

- Last CICS 3270 Screen Buffer Hex

  Selecting the Last CICS 3270 Screen Buffer Hex point-and-shoot field will provide you with the same display as the Last CICS 3270 Screen Buffer point-and-shoot field, but in a format that permits viewing of the hexadecimal values of all characters.

  All non-printable characters are shown as periods (.). The conditions under which information is available are the same as for the Last CICS 3270 Screen Buffer point-and-shoot field.

- Messages

  Messages written to the system console that were not identified as belonging to any specific event are listed under the heading Messages. Also included are any LE messages identified for specific events.

  The explanation of an individual message can be selected by placing the cursor on the message number and pressing Enter.

- DB2 Information

  Selecting the DB2 Information link will provide you with a display of information related to DB2. The next eight panels, Figure 18-15 through Figure 18-22 show an example of the DB2 information.

```
  File  View  Services  Help
 ------------------------------------------------------------------------
DB2 Information                                              Line 1 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR_
JOBNAME: FBIIDRUN  ABEND: n/a                   STLABF6   2004/11/09  10:55:21


DB2 Subsystem DB1E (Group Attach Name DB0E)

DB2 Version . . . . . . . . : V7R1M0
Plan Name . . . . . . . . . : FBIDBIVP
Database Request Module Name: FBIDBIVP
Primary Authorization ID. . : KDFORD
Current SQL ID. . . . . . . : KDFORD
Precompiler Statement No. . : 333
Last Executed SQL Statement : SELECT

Output Host Variables:
  Name and Data Type. . . . : n/a CHARACTER(8)
    At Address. . . . . . . : 0BF29674
    Data Value. . . . . . . : FBIXPLAN

 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right
```

Figure 18-15 DB2 information display (Part 1 of 8)

```
  File  View  Services  Help
 ------------------------------------------------------------------------
DB2 Information                                              Line 18 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR
JOBNAME: FBIIDRUN  ABEND: n/a                   STLABF6   2004/11/09  10:55:21
  Name and Data Type. . . . : n/a CHARACTER(8)
    At Address. . . . . . . : 0BF2967C
    Data Value. . . . . . . : CONOVER

  Name and Data Type. . . . : n/a CHARACTER(6)
    At Address. . . . . . . : 0BF29684
    Data Value. . . . . . . : 041101

  Name and Data Type. . . . : n/a CHARACTER(1)
    At Address. . . . . . . : 0BF2968A
    Data Value. . . . . . . : B

  Name and Data Type. . . . : n/a CHARACTER(1)
    At Address. . . . . . . : 0BF2968B
    Data Value. . . . . . . : S

  Name and Data Type. . . . : n/a CHARACTER(1)
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right
```

Figure 18-16 DB2 information display (Part 2 of 8)

```
  File  View  Services  Help
 ------------------------------------------------------------------------
DB2 Information                                              Line 35 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR
JOBNAME: FBIIDRUN  ABEND: n/a                   STLABF6   2004/11/09  10:55:21
    At Address. . . . . . . : 0BF2968C
    Data Value. . . . . . . : Y

  Name and Data Type. . . . : n/a CHARACTER(1)
    At Address. . . . . . . : 0BF2968D
    Data Value. . . . . . . : Y

  Name and Data Type. . . . : n/a CHARACTER(8)
    At Address. . . . . . . : 0BF2968E
    Data Value. . . . . . . : 13055404

  Name and Data Type. . . . : n/a CHARACTER(5)
    At Address. . . . . . . : 0BF29696
    Data Value. . . . . . . : 04776

  Name and Data Type. . . . : n/a CHARACTER(5)
    At Address. . . . . . . : 0BF2969B
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right
```

Figure 18-17 DB2 information display (Part 3 of 8)

```
  File  View  Services  Help
 ------------------------------------------------------------------------
DB2 Information                                              Line 52 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR
JOBNAME: FBIIDRUN  ABEND: n/a                   STLABF6   2004/11/09  10:55:21
    Data Value. . . . . . . : 04419

  Name and Data Type. . . . : n/a CHARACTER(1)
    At Address. . . . . . . : 0BF296A0
    Data Value. . . . . . . : U

  Name and Data Type. . . . : n/a CHARACTER(1)
    At Address. . . . . . . : 0BF296A1
    Data Value. . . . . . . : C

  Name and Data Type. . . . : n/a CHARACTER(8)
    At Address. . . . . . . : 0BF296A2
    Data Value. . . . . . . : CONOVER

  Name and Data Type. . . . : n/a CHARACTER(8)
    At Address. . . . . . . : 0BF296AA
    Data Value. . . . . . . : FBISTART
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right
```

Figure 18-18 DB2 information display (Part 4 of 8)

```
  File  View  Services  Help
 ------------------------------------------------------------------------
DB2 Information                                              Line 69 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR
JOBNAME: FBIIDRUN  ABEND: n/a                   STLABF6   2004/11/09  10:55:21

  Name and Data Type. . . . : n/a CHARACTER(5)
    At Address. . . . . . . : 0BF296B2
    Data Value. . . . . . . : 01024

  Name and Data Type. . . . : n/a CHARACTER(1)
    At Address. . . . . . . : 0BF296B7
    Data Value. . . . . . . : N

  Name and Data Type. . . . : n/a CHARACTER(16)
    At Address. . . . . . . : 0BF296B8
    Data Value. . . . . . . : (All blank characters)

  Name and Data Type. . . . : n/a CHARACTER(3)
    At Address. . . . . . . : 0BF296C8
    Data Value. . . . . . . : 1

 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right
```

Figure 18-19 DB2 information display (Part 5 of 8)

```
  File  View  Services  Help
 ------------------------------------------------------------------------------
 DB2 Information                                                Line 86 Col 1 80
 Command ===> _____________________________________________ Scroll ===> CSR
 JOBNAME: FBIIDRUN  ABEND: n/a                    STLABF6   2004/11/09  10:55:21
   Name and Data Type. . . . : n/a CHARACTER(1)
     At Address. . . . . . . : 0BF296CB
     Data Value. . . . . . . : D

   Name and Data Type. . . . : n/a CHARACTER(1)
     At Address. . . . . . . : 0BF296CC
     Data Value. . . . . . . : E

 DB2 Control Blocks

 SQL Communications Area (SQLCA) for Event # 3 CSECT FBIDBIVP at Address 0BF299
 Offset     Field        Value
 Dec  Hex   Name         Hex                                 EBCDIC
    0   (0) SQLCAID      E2D8D3C3 C1404040                   *SQLCA          *
    8   (8) SQLCABC      00000088                            *...h           *
   12   (C) SQLCODE      00000000                            *....           *
                         SQLCODE 0 Explanation
  F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
  F8=Down     F10=Left     F11=Right
```

Figure 18-20 DB2 information display (Part 6 of 8)

```
  File  View  Services  Help
 ------------------------------------------------------------------------------
 DB2 Information                                                Line 103 Col 1 80
 Command ===> _____________________________________________ Scroll ===> CSR
 JOBNAME: FBIIDRUN  ABEND: n/a                    STLABF6   2004/11/09  10:55:21
   16  (10) SQLERRML     0000                                *..             *
   18  (12) SQLERRMC     40404040 40404040 40404040 40404040 *               *
   34  (22)              40404040 40404040 40404040 40404040 *               *
   50  (32)              40404040 40404040 40404040 40404040 *               *
   66  (42)              40404040 40404040 40404040 40404040 *               *
   82  (52)              40404040 4040                       *               *
   88  (58) SQLERRP       C4E2D540 40404040                   *DSN            *
   96  (60) SQLERRD       00000000 00000000 00000000 00000000 *...............*
  112  (70)              00000000 00000000                   *........       *
  120  (78) SQLWARN       40404040 40404040 404040            *               *
  131  (83) SQLSTATE      F0F0F0F0 F0                         *00000          *
                         SQLSTATE 00000 Explanation

 SQL Communications Area (SQLCA) for Subsystem DB1E
 Offset     Field        Value
 Dec  Hex   Name         Hex                                 EBCDIC
    0   (0) SQLCAID      E2D8D3C3 C1404040                   *SQLCA          *
  F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
  F8=Down     F10=Left     F11=Right
```

Figure 18-21 DB2 information display (Part 7 of 8)

```
  File  View  Services  Help
 ------------------------------------------------------------------------------
 DB2 Information                                              Line 120 Col 1 80
 Command ===> _____________________________________________ Scroll ===> CSR
 JOBNAME: FBIIDRUN  ABEND: n/a                    STLABF6   2004/11/09  10:55:21
    8   (8) SQLCABC      00000088                             *...h            *
   12   (C) SQLCODE      00000000                             *....            *
                         SQLCODE 0 Explanation
   16  (10) SQLERRML     0000                                 *..              *
   18  (12) SQLERRMC     40404040 40404040 40404040 40404040 *                *
   34  (22)              40404040 40404040 40404040 40404040 *                *
   50  (32)              40404040 40404040 40404040 40404040 *                *
   66  (42)              40404040 40404040 40404040 40404040 *                *
   82  (52)              40404040 4040                       *                *
   88  (58) SQLERRP      C4E2D540 40404040                   *DSN             *
   96  (60) SQLERRD      00000000 00000000 00000000 FFFFFFFF *................*
  112  (70)              00000000 00000000                   *........        *
  120  (78) SQLWARN      40404040 40404040 404040             *                *
  131  (83) SQLSTATE     F0F0F0F0 F0                         *00000           *
                         SQLSTATE 00000 Explanation

 *** Bottom of data.
  F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
  F8=Down     F10=Left     F11=Right
```

Figure 18-22 DB2 information display (Part 8 of 8)

Information for one or more DB2 subsystems is provided. If the SQLCA data area for a DB2 subsystem is available from a detailed event display, then only a reference to the event is provided here, as shown in the example. Otherwise, the contents of the SQLCA will be shown in this display.

- IMS Information

  Selecting the IMS Information link will provide you with a display of information related to IMS. Figure 18-23 through Figure 18-25 provide an example of IMS information.

```
 File  View  Services  Help
 ------------------------------------------------------------------------------
 IMS Information                                              Line 1 Col 1 80
 Command ===> ____________________________________________ Scroll ===> CSR
 JOBNAME: IBCB0030  USER ABEND: 4036                MVS2     2002/11/29 13:51:55

 IMS Version . . . . . . . . : V7R1M0
 IMS Region Type . . . . . . : Online Control Region (DB/DC)
 IMS Subsystem Name. . . . . : IB21
 Application Program Name. . : ICCB0010
 PSB Name. . . . . . . . . . : DFHSAM25
```

Figure 18-23 IMS information display (Part 1 of 3)

```
Last DL/I Call Parameter List

Note that storage addressed by individual parameters might no longer be valid.

Parameter 1 . . . . . : 1730A3D0
  DL/I Call Function. . . . : GU (Get Unique)

Parameter 2 . . . . . : 00016398
  (See "IMS Control Blocks" for details of this PCB)

Parameter 3 . . . . . : 1730A310

Parameter 4 . . . . . : 9730A3B0
  SSA # 1 . . . . . : PARTROOT(         =          )

IMS Control Blocks

Input/Output Program Communications Block (IOPCB):
  At Address. . . . . . . . : 00016320
  PCB Name. . . . . . . . . : IOPCB
  Relative PCB Number . . . : 1
  PCB Type. . . . . . . . . : I/O
  Logical Terminal ID . . . : n/a
  Status Code . . . . . . . : '  ' (Normal status)
  User ID . . . . . . . . . : DFHSAM25
  Group Name. . . . . . . . : n/a
  Formatting Module Name. . : n/a

Data Base Program Communications Block (DBPCB) (*** Current/Last Used ***):
  At Address. . . . . . . . : 00016398
  PCB Name. . . . . . . . . : n/a
  Relative PCB Number . . . : 2
  PCB Type. . . . . . . . . : Data Base or Online
  Data Base Name. . . . . . : DI21PART
  Segment Level . . . . . . : 01
  Status Code . . . . . . . : '  ' (Normal status)
  Processing Options. . . . : A
  Segment Name. . . . . . . : PARTROOT
  Number of Segments. . . . : 5
  Key Feedback Length . . . : 17

  Key Feedback Data:
    Address  Offset      Hex                                 EBCDIC
    000163F4             F0F2C1D5 F9F6F0C3 F1F04040 40404040 *02AN960C10     *
    00016404       +10   40                                  *               *

  JCB DL/I Call Trace (Most recent call first):
      Call                      Status
    # Code Description         Code    Description
    1 01   GHU or GU            '  '    Status good.
```

Figure 18-24 IMS information display (Part 2 of 3)

```
IMS Accounting Information

DL/I Data Base Calls:
  GU Calls. . . . . : 1
  GN Calls. . . . . : 0
  GNP Calls . . . . . : 0
  GHU Calls . . . . . : 0
  GHN Calls . . . . . : 0
  GHNP Calls. . . . . : 0
  ISRT Calls. . . . . : 0
  DLET Calls. . . . . : 0
  REPL Calls. . . . . : 0
  Total Calls . . . . . : 1
DL/I Message Calls:
  GU Calls. . . . . : 0
  GN Calls. . . . . : 0
  ISRT Calls. . . . . : 0
  PURG Calls. . . . . : 0
  CMD Calls . . . . . : 0
  GCMD Calls. . . . . : 0
  CHNG Calls. . . . . : 0
  AUTH Calls. . . . . : 0
  SETO Calls. . . . . : 0
DL/I System Service Calls:
  APSB Calls. . . . . : 0
  DPSB Calls. . . . . : 0
  GMSG Calls. . . . . : 0
  ICMD Calls. . . . . : 0
  RCMD Calls. . . . . : 0
  CHKP Calls. . . . . : 0
  XRST Calls. . . . . : 0
  ROLB Calls. . . . . : 0
  ROLS Calls. . . . . : 0
  SETS Calls. . . . . : 0
  SETU Calls. . . . . : 0
  INIT Calls. . . . . : 0
  INQY Calls. . . . . : 0
  LOG Calls . . . . . : 0

IMS Parameter Modules

Module DFSPRPX0 Address . . : 000079C0

*** Bottom of data.
 F1=Help      F3=Exit      F5=RptFind   F6=Actions   F7=Up        F8=Down
F10=Left     F11=Right
```

Figure 18-25 IMS information display (Part3 of 3)

The IMS Information display provides:

- General information about the IMS region.
- Last DL/I call parameter list with detailed information for all PCBs. All PCBs are shown in the order of their relative PCB number with identification of current (or most recently used) PCBs. If available, JCB call trace information follows each database PCB, showing the most recent call and up to five previous calls.
- IMS accounting information.
- The address of the DFSPRPX0 parameter module. Selecting the address point-and-shoot field permits you to view the module storage in hex-dump format.

► Open Files

Selecting the Open Files link will provide you with a display that lists all open files which could not be associated with any particular event, as well as files that might be listed in the detail section of the report for individual events. An example of the open file list is shown in Figure 18-26.

```
  File  View  Services  Help
 ------------------------------------------------------------------------
 System-Wide Open Files                                   Line 1 Col 1 80
 Command ===> _____________________________________________ Scroll ===> CSR
 JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7          STLABF6   2004/11/08  21:58:09


 Non-Event-Related Open Files

 File Name . . . . . . . . . : SYSOUT

 *** Bottom of data.



  F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
  F8=Down     F10=Left     F11=Right
```

Figure 18-26 System-Wide open files display

The listed file names are point-and-shoot fields. Place the cursor on the file of interest and press Enter to see the associated detailed file information.

For example, if selecting the file name SYSOUT from the sample display, you might see the File Information display shown in Figure 18-27. To return from the File Information display, use the Exit command (PF3).

```
  File  View  Services  Help
File Information                                                Line 1 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR
JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7          STLABF6   2004/11/08  21:58:09

File Name . . . . . . . . . : SYSOUT
  Data Set Name . . . . . . : GRACINE.$TSTFACB.JOB01308.D0000110.?
  File Attributes . . . . . : ORGANIZATION=SEQUENTIAL, ACCESS MODE=n/a, RECFM=
  Last I/O Function . . . . : WRITE
  Open Status . . . . . . . : OUTPUT

  Current Record. . . . . . : Record data length 121
  Address  Offset    Hex                                  EBCDIC
  0001C0B8           40404040 40404040 40404040 40404040 *                *
        Lines 0001C0C8-0001C118 same as above
  0001C128       +70  40404040 40404040 40               *                *

*** Bottom of data.


 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right
```

Figure 18-27 File information display

If the data set shown is QSAM, VSAM, or IAM, and exists, then it will be presented as a point-and-shoot field. If the field is selected, a menu is returned where you can select either Edit or Browse.

If IBM File Manager for z/OS is available, it will be used to perform the requested function against the data set. Otherwise, if the data set is QSAM non-spanned record format, then ISPF is invoked.

- Storage Areas

  Selecting the Storage Areas link will provide you with point-and-shoot links to any event-related formatted storage areas, any hex-dumped storage ranges, and in the case of COBOL, information about any static storage for programs that are no longer on the DSA chain, as shown in Figure 18-28.

```
  File  View  Services  Help
System-Wide Storage Areas                                       Line 1 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR
JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7          STLABF6   2004/11/08  21:58:09

Event 1 Program IDISCBL1 Storage Areas

Hex-Dumped Storage

*** Bottom of data.









 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right
```

Figure 18-28 System-Wide Storage Areas display

Selecting any event-related point-and-shoot links from this display (such as the Event 4 Program COBMAIN Storage Areas link) results in a display which is similar to the one presented if the Associated Storage Areas link is selected from the detailed section for the event.

The Hex-Dumped Storage link provides a display of relevant unformatted storage areas which might be for one or more events.

An example of the hex-dumped storage display is shown in Figure 18-29.

```
  File  View  Services  Help
 ------------------------------------------------------------------------------
 Hex-Dumped Storage                                        CHARS '0c1710d0' found
 Command ===> ____________________________________________ Scroll ===> CSR
 JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7            STLABF6   2004/11/08  21:58:09
 0C151D00      +310  E9D4C3C8 02000001 0C151140 0BF00F55  000167FC 0BF0111C 0BF
 0C151D20      +330  00000000 00000000 0C1710D0 0C14D448  0BF00E5C 0BF00F80 0BF
 0C151D40      +350  8BF010CC 8C11ED70 078D2000 8BF01158  00060007 00000000 000
 0C151D60      +370  00000000 00000000 00000000 00000000  00000000 00000000 000
 0C151D80      +390  00000000 00000000 00000000 00000000  00000000 00000000 000
 0C151DA0      +3B0  00000000 00000000 00000000 00000000  00000040 00000000 000
 0C151DC0      +3D0  00000000 00000000 00000000 00000000  00000000 00000000 000
          Lines 0C151DE0-0C151EE0 same as above
 0C151F00      +510  C3C9C2C8 0BF09888 00000000 00020A00  00000000 00000000 000
 0C151F20      +530  00000000 00000000 00000000 00000000  00000000 00000000 000
          Lines 0C151F40-0C152120 same as above

          Event 1 Program IDISCBL1 BLW=0000 (Address 0C1710D0)
          Event 1 Program IDISCBL1 GPR 8 (Address 0C1710D0)
          See Event 1 Program IDISCBL1 Storage Areas for address range 0C1710D0

 *** Bottom of data.
  F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
  F8=Down     F10=Left     F11=Right
```

Figure 18-29 HEX-dumped storage display

- Java Information

  See Chapter 7, “Performing WebSphere or Java dump analysis” in the User and Reference Guide.

- WebSphere Information
  - See Chapter 7, “Performing WebSphere or Java dump analysis” in the User and Reference Guide.

- Language Environment Heap Analysis
  - Selecting the Language Environment Heap Analysis link will provide you with a display of information related to the LE heap. The next five panels, Figure 18-30 through Figure 18-34 provide an example.

```
  File  View  Services  Help
 ------------------------------------------------------------------------
Language Environment Heap Analysis                                Top of data
Command ===> ___________________________________________ Scroll ===> CSR
JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7          STLABF6   2004/11/08  21:58:09


Enclave-Level Storage
  Management (ENSM) Address : 0BF0BF28
Heap allocation
  initialization value
  specified . . . . . . . . : No
Heap free initialization
  value specified . . . . . : No

User Heap Analysis

Heap Anchor Node (HANC) . . : 0C171018
Heapid. . . . . . . . . . . : 00000000
Root Address. . . . . . . . : 0C171108
Segment Length. . . . . . . : 00008000
Root Length . . . . . . . . : 00007F10
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right
```

Figure 18-30 Language environment heap analysis display (Part 1 of 5)

```
  File  View  Services  Help
 ------------------------------------------------------------------------
Language Environment Heap Analysis                          Line 18 Col 1 80
Command ===> ___________________________________________ Scroll ===> CSR
JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7          STLABF6   2004/11/08  21:58:09

Free Storage Tree In This Heap Segment

Node  Node     Node     Parent   Left     Right    Left     Right
Depth Address  Length   Address  Node     Node     Length   Length
    0 0C171108 00007F10 00000000 00000000 00000000 00000000 00000000

Statistics For This Heap Segment

Total Bytes Of Identified
  Free Storage. . . . . . . : 00007F10
Total No Of Identified Free
  Elements. . . . . . . . . :        1
Total Bytes Of Identified
  Allocated Storage . . . . : 000000D0
Total No Of Identified
  Allocated Storage Elements:        1
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right
```

Figure 18-31 Language environment heap analysis display (Part 2 of 5)

```
  File  View  Services  Help
 ------------------------------------------------------------------------
 Language Environment Heap Analysis                          Line 35 Col 1 80
 Command ===> _____________________________________________ Scroll ===> CSR
 JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7          STLABF6   2004/11/08  21:58:09
 Total Bytes Of Unidentified
   Storage . . . . . . . . . : 00000000

 No errors were found during this segment analysis

 Anywhere Heap Analysis

 Heap Anchor Node (HANC) . . : 0C14D000
 Heapid. . . . . . . . . . . : 0BF0BF70
 Root Address. . . . . . . . : 0C14EF98
 Segment Length. . . . . . . : 00004000
 Root Length . . . . . . . . : 00002068

 Free Storage Tree In This Heap Segment

 Node  Node     Node     Parent   Left     Right    Left     Right
 Depth Address  Length   Address  Node     Node     Length   Length
  F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
  F8=Down     F10=Left     F11=Right
```

Figure 18-32 Language environment heap analysis display (Part 3 of 5)

```
  File  View  Services  Help
 ------------------------------------------------------------------------
 Language Environment Heap Analysis                          Line 52 Col 1 80
 Command ===> _____________________________________________ Scroll ===> CSR
 JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7          STLABF6   2004/11/08  21:58:09
     0 0C14EF98 00002068 00000000 00000000 00000000 00000000 00000000

 Statistics For This Heap Segment

 Total Bytes Of Identified
   Free Storage. . . . . . . : 00002068
 Total No Of Identified Free
   Elements. . . . . . . . . :        1
 Total Bytes Of Identified
   Allocated Storage . . . . : 00001F78
 Total No Of Identified
   Allocated Storage Elements:       11
 Total Bytes Of Unidentified
   Storage . . . . . . . . . : 00000000

 No errors were found during this segment analysis

  F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
  F8=Down     F10=Left     F11=Right
```

Figure 18-33 Language environment heap analysis display (Part 4 of 5)

```
  File  View  Services  Help
 ------------------------------------------------------------------------
Language Environment Heap Analysis                          Line 61 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR
JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7          STLABF6   2004/11/08  21:58:09
  Allocated Storage . . . . : 00001F78
Total No Of Identified
  Allocated Storage Elements:       11
Total Bytes Of Unidentified
  Storage . . . . . . . . . : 00000000

No errors were found during this segment analysis

Below Heap Analysis

There are no segments in this heap

Additional Heap Control Blocks

There are no additional heaps

*** Bottom of data.
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right
```

Figure 18-34 Language environment heap analysis display (Part 5 of 5)

18.8 Abend job information

Select option **4** from the initial interactive report display to view the Abend Job Information section of the report as shown in Figure 18-35.

```
  File  View  Services  Help
 ------------------------------------------------------------------------
Abend Job Information                                        Line 1 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR
JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7          STLABF6   2004/11/08  21:58:09

IBM Fault Analyzer Abend Job Information:

  Abend Date. . . . . . . . : 2004/11/08
  Abend Time. . . . . . . . : 21:58:09
  System Name . . . . . . . : STLABF6
  Job Type. . . . . . . . . : Batch
  Job Name. . . . . . . . . : $TSTFACB
  Job Step Name . . . . . . : GO
  ASID. . . . . . . . . . . : 2A
  Abend TCB Address . . . . : 00ACAD90
  Job Class . . . . . . . . : A
  EXEC Program Name . . . . : IDISCBL1
  User ID . . . . . . . . . : GRACINE

  Data Sets:

 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right
```

Figure 18-35 Abend Job Information display

This display provides information about the environment that existed when the fault was analyzed in real time. The information shown depends on the type of fault analyzed.

18.9 Options in effect

Select option **5** from the initial interactive report display to view the Options in Effect section of the report as shown in Figure 18-36.

```
  File  View  Services  Help
 ------------------------------------------------------------------------
 Options in Effect                                          Line 1 Col 1 80
 Command ===> _____________________________________________ Scroll ===> CSR
 JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7          STLABF6   2004/11/08  21:58:09

 IBM Fault Analyzer Options in Effect:

   FaultID(F00002)
   Language(ENU)
   NoLocale

   Data Sets:

     {The following Fault Analyzer data set or path names were either prealloca
     defaults.}

     DDname   Data Set or Path Name
     IDIBOOKS ADTOOLS.FAULT.ANAL51.SIDIBOOK
     IDICACHE IDI.CACHE
     IDIDOC   ADTOOLS.FAULT.ANAL51.SIDIDOC1
     IDIHIST  IDI.HIST.TEST
  F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
  F8=Down     F10=Left     F11=Right
```

Figure 18-36 Options in effect display

18.10 Displaying associated storage areas

In the interactive report, the link for Associated Storage Areas is provided in the detailed event display depending on the programming language used. What is displayed when selecting this link depends on both the programming language and whether or not a compiler listing or side file has been made available to Fault Analyzer for the program:

- For COBOL programs without a compiler listing or side file, the TGT and base locators are displayed in hexadecimal dump format.
- For COBOL programs with a compiler listing or side file supplied, the source declaration of all fields along with their current content is provided. An example of the associated storage areas display for a COBOL program with source listing provided is shown in the next five figures, Figure 18-37 through Figure 18-41.

```
  File  View  Services  Help
 ------------------------------------------------------------------------
 Interactive Reanalysis Report                                Line 1 Col 1 80
 Command ===> ____________________________________________ Scroll ===> CSR
 JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7          STLABF6   2004/11/08  21:58:09

 Fault Summary:
 Module IDISCBL1, program IDISCBL1, source line # 31 : Abend S0C7 (Data Excepti

 Select one of the following options and press Enter to access further fault in
 2  1.  Synopsis
    2.  Event Summary
    3.  System-Wide Information
    4.  Abend Job Information
    5.  Options in Effect

 {Fault Analyzer maximum storage allocated: 1.43 megabytes.}

 *** Bottom of data.


  F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
  F8=Down     F10=Left     F11=Right
```

Figure 18-37 Associated Storage Areas display (Part 1 of 5)

```
  File  View  Services  Help
 ------------------------------------------------------------------------
 Event Summary                                                Line 1 Col 1 80
 Command ===> ____________________________________________ Scroll ===> CSR
 JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7          STLABF6   2004/11/08  21:58:09

 {The following events are presented in chronological order.}

 Event           Fail  Module   Program  EP
 #  Type         Point Name     Name     Name     Event Location (*)      Loaded
 s1 Abend S0C7   ***** IDISCBL1 IDISCBL1 IDISCBL1 L#31 P+422              SYS043
  2 Call               CEEPLPKA n/a      CEEHDSP  E+3986                  CEE.SC
  3 Abend U4039        CEEPLPKA n/a      CEEHSDMP E+D4                    CEE.SC


 (*) One or more of the following abbreviations might appear in the "Event Loca

     F#n  Source file number (refer to detailed event information for file iden
     L#n  Source file line number
     S#n  Listing file statement number (refer to detailed event information fo
     M+x  Offset from start of load module
     P+x  Offset from start of program
  F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
  F8=Down     F10=Left     F11=Right
```

Figure 18-38 Associated Storage Areas display (Part 2 of 5)

```
  File  View  Services  Help
 ------------------------------------------------------------------------
 Event Summary                                                Line 1 Col 1 80
 Command ===> ____________________________________________ Scroll ===> CSR
 JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7            STLABF6   2004/11/08  21:58:09

 {The following events are presented in chronological order.}

 Event           Fail  Module   Program  EP
 #  Type         Point Name     Name     Name     Event Location (*)      Loaded
 s1 Abend S0C7   ***** IDISCBL1 IDISCBL1 IDISCBL1 L#31 P+422              SYS043
  2 Call               CEEPLPKA n/a      CEEHDSP  E+3986                  CEE.SC
  3 Abend U4039        CEEPLPKA n/a      CEEHSDMP E+D4                    CEE.SC


 (*) One or more of the following abbreviations might appear in the "Event Loca

     F#n  Source file number (refer to detailed event information for file iden
     L#n  Source file line number
     S#n  Listing file statement number (refer to detailed event information fo
     M+x  Offset from start of load module
     P+x  Offset from start of program
  F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
  F8=Down     F10=Left     F11=Right
```

Figure 18-39 Associated Storage Areas display (Part 3 of 5)

```
  File  View  Services  Help
 ------------------------------------------------------------------------
 Event 1 of 3: Abend S0C7 *** Point of Failure ***            Line 62 Col 1 80
 Command ===> ____________________________________________ Scroll ===> CSR
 JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7            STLABF6   2004/11/08  21:58:09
   R10: 0BF00E5C (Module IDISCBL1 program IDISCBL1 + X'12C')
   R11: 0BF00F80 (Module IDISCBL1 program IDISCBL1 + X'250')
   R12: 0BF00E2C (Module IDISCBL1 program IDISCBL1 + X'FC')
   R13: 0C151030 (8466384 bytes of storage addressable)
   R14: 8BF010CC (Module IDISCBL1 program IDISCBL1 + X'39C', source line # 23 )
   R15: 8C11ED70 (Module IGZCPAC + X'43480')

 Associated Messages

 CEE3207S Job-specific text not available

 Associated Storage Areas


 Next Event Details

 *** Bottom of data.
  F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
  F8=Down     F10=Left     F11=Right
```

Figure 18-40 Associated Storage Areas display (Part 4 of 5)

```
  File  View  Services  Help
 ____________________________________________________________________________
Associated Storage Areas                                      Line 1 Col 1 80
Command ===> ____________________________________________ Scroll ===> CSR
JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7           STLABF6   2004/11/08  21:58:09

Task Global Table (TGT) at address 0C14D448 for length 328


WORKING-STORAGE SECTION
 Off Hex Value                            EBCDIC Value       Source (Starting a
BLW=0000 at address 0C1710D0
   0 E6D6D9D2 C9D5C760 E2E3D6D9 C1C7C540 *WORKING-STORAGE *  01  FILLER
  10 40404040                            *               *
  18 0986888F                            986888             01  NUMBERX PIC 9
                                                            01  ERROR-FLD.
  20 C1C2C3C4                            *ABCD           *      05  ERROR-COU
                                                                05  FLDY REDE
  20 C1C2C3C4                            *ABCD           *          07 FLDZ P
  28 0000                                *..             *  01  BAD-RESULT PI


 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right
```

Figure 18-41 Associated Storage Areas display (Part 5 of 5)

By scrolling down through the displayed information, you will also find Linkage Section information, File Section information, and so forth, as appropriate for the current program.

18.11 Expanding messages and abend codes

Messages or abend codes are initially never expanded when using the interactive dump reanalysis feature of Fault Analyzer. This is to prevent the need to scroll through potentially very long explanations to see report items that might follow. Instead, to view the explanation for messages or abend codes in the interactive report, place the cursor on the message identifier or abend code and press Enter. This will bring up a display similar to what you see in the batch report; the next three figures, Figure 18-42 to Figure 18-44 provide an example.

```
  File  View  Services  Help
System-Wide Information                                    Line 1 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR
JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7          STLABF6   2004/11/08  21:58:09

Open Files

Storage Areas

Messages

Language Environment Heap Analysis

*** Bottom of data.

 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right
```

Figure 18-42 Message Explanation display (Part 1 of 3)

```
  File  View  Services  Help
System-Wide Messages                                       Line 1 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR
JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7          STLABF6   2004/11/08  21:58:09

Event 1 Program IDISCBL1 Messages

CEE3207S Job-specific text not available

Non-Event-Related Messages

SMF000I  $TSTFACB    DELDS       IDCAMS      0000

SMF000I  $TSTFACB    COBOL       IGYCRCTL    0000

SMF000I  $TSTFACB    LKED        HEWL        0000

*** Bottom of data.

 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right
```

Figure 18-43 Message Explanation display (Part 2 of 3)

```
  File  View  Services  Help
 ------------------------------------------------------------------------------
Message CEE3207S Explanation                                   Line 1 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR
JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7            STLABF6   2004/11/08  21:58:09

  CEE3207S The system detected a data exception (System Completion
         Code=0C7).

  Explanation:  Your program attempted to use a decimal instruction
  incorrectly. See a Principles of Operation manual for a full list of data
  exceptions.

  Programmer Response:  Check the variables associated with the failing
  statement to make sure that they have been initialized correctly.

  System Action:  The thread is terminated.

  Symbolic Feedback Code:  CEE347

*** Bottom of data.

 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right
```

Figure 18-44 Message Explanation display (Part 3 of 3)

When a message is displayed in the report, the first occurrence of the message contains the actual text from when it was issued. If the instance-specific text is not available, then the comment `Job-specific text not available` will replace it.

An explanation is included in the expansion that immediately follows the issued message or abend code; it is generally obtained from the softcopy books provided with Fault Analyzer. If an explanation for the message or abend code could not be found, then the text `Explanation not available` will replace it.

18.12 Displaying source code

To display the source code for an entire program, place the cursor on any point-and-shoot source line number or listing statement number and press Enter. For example, if line number 11 was selected from an event in the interactive report, the result would be the panel shown in Figure 18-45.

```
  File  View  Services  Help
 ____________________________________________________________________________
Program IDISCBL1 Compiler Listing                              Line 81 Col 1 80
Command ===> ___________________________________________________ Scroll ===> CSR
JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7            STLABF6   2004/11/08  21:58:09
               000000E8 0000 0000 0000
               000000EE 0000 0000 0000
               000000F4 0000 0000 0000
               000000FA 0000
000006          DATA DIVISION.
000007          FILE SECTION.
000009          WORKING-STORAGE SECTION.
000010          01  FILLER                 PIC X(20)  VALUE 'WORKING-STORAGE'.
000011          01  NUMBERX PIC 999999 COMP-3.
000012          01  ERROR-FLD.
000013              05  ERROR-COUNT PIC 999999 COMP-3.
000014              05  FLDY REDEFINES ERROR-COUNT.
000015                  07 FLDZ PIC XXXX.
000016          01  BAD-RESULT PIC 99 COMP-3.
               000000FC 0000 0001
               00000100 4040 4040      STH   R4,64(,R4)
               00000104 4040 4040      STH   R4,64(,R4)
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right
```

Figure 18-45 Compiler Listing display

Note that the source line or statement number initially selected is highlighted.

The example shown here assumes that detailed information is suppressed.

The information (all lines starting in column 1) pertains to the program source code.

On the left side of the display is information about the source line or listing statement number of the source displayed. This is followed by the actual source code at this location in the program.

To add machine instruction information to the listing, select **View** → **Add Detail Information**. This will cause the Compiler Listing display to be reformatted with pseudo assembler instructions inserted into the program source code.

This information pertains to the disassembly of machine instructions, which are shown interspersed with the source code information following the line of code to which they belong.

> **Note:** For COBOL programs compiled with TEST(,,SEPARATE), all pseudo assembler instructions are placed following the last line of source code.

To remove the pseudo assembler instructions from the display, select **View** → **Remove Detail Information**. If you scroll to the top of the listing, for example by issuing the UP MAX command, you will see information about where the compiler listing or side file was obtained.

18.13 Displaying storage locations

To display storage locations, place the cursor on any point-and-shoot address (for example, a register value), and press Enter. Alternatively, use the SHOW command. An example of the storage display is shown in Figure 18-46.

```
  File  View  Services  Help
 ____________________________________________________________________________
Dump Storage                                                 0C16C0D0-0C18C03F
Command ===> ____________________________________________ Scroll ===> CSR
JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7          STLABF6   2004/11/03  13:14:50

Address  Offset    Hex                                 EBCDIC
         Event 1 Program IDISCBL1 BLW=0000
0C16C0D0            This is. ..D5C760 E2E3D6D9 C1C7C540 *WORKING-STORAGE *
0C16C0E0       +10  40404040 00000000 0986888F 00000000 *    .....fh.....*
0C16C0F0       +20  C1C2C3C4 00000000 00000000 00000000 *ABCD............*
0C16C100       +30  00000000 00000000 00000000 00000000 *................*
         Lines 0C16C110-0C16DFF0 same as above
         Address range 0C16E000-0C175CAF not in minidump
         Module CEEMENU3
         Address range 0C175CB0-0C1805D7 not in minidump
         Module IDIXDCAP
         Address range 0C1805D8-0C18BFFF not in minidump
0C18C000    +1FF30  90000000 00400012 00000000 5000010C *..... ......&...*
0C18C010    +1FF40  FFFFE1A0 38260000 40080104 00000000 *........ .......*
0C18C020    +1FF50  0008A497 978599D4 A2870000 00000000 *..upperMsg......*
0C18C030    +1FF60  1CCEA106 00001D08 00000000 00000000 *..~.............*
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F7=Up        F8=Down
F10=Prev     F11=Next
```

Figure 18-46 Dump storage display

Placing the cursor anywhere in the hexadecimal storage display area and pressing Enter will take you to the selected address.

A record is maintained of the last 10 addresses displayed. To re-display areas previously selected, use the PREV (PF10) or NEXT (PF11) commands. The number of bytes per line shown depends on the current preferred formatting width.

> **Note:** If it is less than 131, then 16 bytes will be shown per line. Otherwise, 32 bytes will be shown.

If available, the following information can be provided:

- A description of the initially selected address
- Descriptions of the beginning of other storage areas
  - Modules
  - Programs
  - Any user notes

> **Note:** The FIND command used from this display behaves different from all other displays since it is the minidump which is searched instead of the formatted display itself.

18.13.1 Creating and managing user notes

User notes are comments that the interactive user can add against any storage location. They are saved in the history file fault entry and are available to all users.

User notes are created from the Dump Storage display by placing the cursor on the area of storage to which the note will apply, typing one or more characters that are distinguishable from hexadecimal digits, and pressing Enter.

For example, given the Dump Storage display shown in Figure 18-47, placing the cursor at the address, and typing `This is`, the display in Figure 18-48 would be expected.

```
  File  View  Services  Help
 ------------------------------------------------------------------------------
 Dump Storage                                                 0C16C0D0-0C18C03F
 Command ===> ____________________________________________ Scroll ===> CSR
 JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7          STLABF6   2004/11/03  13:14:50

 Address  Offset    Hex                                 EBCDIC
          Event 1 Program IDISCBL1 BLW=0000
 0C16C0D0            This is. ..D5C760 E2E3D6D9 C1C7C540 *WORKING-STORAGE *
 0C16C0E0       +10  40404040 00000000 0986888F 00000000 *    .....fh.....*
 0C16C0F0       +20  C1C2C3C4 00000000 00000000 00000000 *ABCD............*
 0C16C100       +30  00000000 00000000 00000000 00000000 *................*
          Lines 0C16C110-0C16DFF0 same as above
          Address range 0C16E000-0C175CAF not in minidump
          Module CEEMENU3
          Address range 0C175CB0-0C1805D7 not in minidump
          Module IDIXDCAP
          Address range 0C1805D8-0C18BFFF not in minidump
 0C18C000    +1FF30  90000000 00400012 00000000 5000010C *..... ......&...*
 0C18C010    +1FF40  FFFFE1A0 38260000 40080104 00000000 *........ .......*
 0C18C020    +1FF50  0008A497 978599D4 A2870000 00000000 *..upperMsg......*
 0C18C030    +1FF60  1CCEA106 00001D08 00000000 00000000 *..~.............*
  F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F7=Up        F8=Down
 F10=Prev     F11=Next
```

Figure 18-47 DUMP STORAGE DISPLAY

Pressing Enter results in an ISPF edit panel as shown in Figure 18-48 initialized with the text typed.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------------
 EDIT       Note.0C16C0D0                                  Columns 00001 00072
 Command ===> ____________________________________________ Scroll ===> PAGE
 ****** ***************************** Top of Data ******************************
 ==MSG> -Warning- The UNDO command is not available until you change
 ==MSG>           your edit profile using the command RECOVERY ON.
 000001 This is...
 ****** **************************** Bottom of Data ****************************

  F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
  F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 18-48 DUMP STORAGE DISPLAY with the text type (part 1 of 3)

From here the note can be completed, adding as many lines as required. The first line should be treated as a heading since it will be the only line of the note shown if the display of the note is later collapsed. An example of the final note is as shown in Figure 18-49.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------
EDIT       Note.0C16C0D0                                   Columns 00001 00072
Command ===> ____________________________________________ Scroll ===> PAGE
****** **************************** Top of Data ****************************
==MSG> -Warning- The UNDO command is not available until you change
==MSG>           your edit profile using the command RECOVERY ON.
000001 This is a REMINDER.
000002 The contents of storage at this offset into this module could be
000003 significant for the understanding of the error that caused this fault.
****** **************************** Bottom of Data **************************
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 18-49 DUMP STORAGE DISPLAY with the text type (part 2 of 3)

Press PF3 to return to the Dump Storage display, which will show the newly created user note as shown in Figure 18-50, inserted immediately ahead of the storage to which it belongs.

```
  File  View  Services  Help
 ------------------------------------------------------------------------
Dump Storage                                              0C16C0D0-0C18C00F
Command ===> ____________________________________________ Scroll ===> CSR
JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7          STLABF6   2004/11/03  13:14:50

Address  Offset    Hex                                  EBCDIC
         Event 1 Program IDISCBL1 BLW=0000
- This is a REMINDER.
  The contents of storage at this offset into this module could be
  significant for the understanding of the error that caused this fault.
0C16C0D0            E6D6D9D2 C9D5C760 E2E3D6D9 C1C7C540 *WORKING-STORAGE *
0C16C0E0       +10  40404040 00000000 0986888F 00000000 *    .....fh.....*
0C16C0F0       +20  C1C2C3C4 00000000 00000000 00000000 *ABCD............*
0C16C100       +30  00000000 00000000 00000000 00000000 *................*
         Lines 0C16C110-0C16DFF0 same as above
         Address range 0C16E000-0C175CAF not in minidump
         Module CEEMENU3
         Address range 0C175CB0-0C1805D7 not in minidump
         Module IDIXDCAP
         Address range 0C1805D8-0C18BFFF not in minidump
0C18C000    +1FF30  90000000 00400012 00000000 5000010C *..... ......&...*
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F7=Up        F8=Down
F10=Prev     F11=Next
```

Figure 18-50 DUMP STORAGE DISPLAY with the text type (part 3 of 3)

By default, all user notes are shown expanded, as indicated by the minus sign point-and-shoot field preceding the note.

Place the cursor on this field (the minus sign) and press Enter; the note will be collapsed as shown in Figure 18-51.

```
  File  View  Services  Help
 ------------------------------------------------------------------------
 Dump Storage                                                 0C16C0D0-0C18C02F
 Command ===> _____________________________________________ Scroll ===> CSR
 JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7          STLABF6   2004/11/03  13:14:50

 Address  Offset    Hex                                  EBCDIC
          Event 1 Program IDISCBL1 BLW=0000
 + This is a REMINDER.
 0C16C0D0            E6D6D9D2 C9D5C760 E2E3D6D9 C1C7C540 *WORKING-STORAGE *
 0C16C0E0       +10  40404040 00000000 0986888F 00000000 *    .....fh.....*
 0C16C0F0       +20  C1C2C3C4 00000000 00000000 00000000 *ABCD............*
 0C16C100       +30  00000000 00000000 00000000 00000000 *................*
          Lines 0C16C110-0C16DFF0 same as above
          Address range 0C16E000-0C175CAF not in minidump
          Module CEEMENU3
          Address range 0C175CB0-0C1805D7 not in minidump
          Module IDIXDCAP
          Address range 0C1805D8-0C18BFFF not in minidump
 0C18C000    +1FF30  90000000 00400012 00000000 5000010C *..... ......&...*
 0C18C010    +1FF40  FFFFE1A0 38260000 40080104 00000000 *........ .......*
 0C18C020    +1FF50  0008A497 978599D4 A2870000 00000000 *..upperMsg......*
  F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F7=Up        F8=Down
 F10=Prev     F11=Next
```

Figure 18-51 User note display list (collapsed +/-)

The plus sign that now appears in front of the notes line indicates that the note is collapsed. By simply placing the cursor on this point-and-shoot field, the user can toggle between the collapsed and expanded view.

It is also possible to overtype the point-and-shoot field with two additional action characters (case insensitive):

D Delete the user note
E Edit the user note

To see all user notes that exist for the current fault entry, enter the NOTELIST command from the command line of any display within the interactive report, or select **View** → **List User Notes**.

As indicated in the optional help text on this display, the point-and-shoot field preceding each user note can be overtyped to request a specific action in the same way it is done in the Dump Storage display.

Additionally, the User Note List display permits all user notes to be expanded or collapsed simultaneously by selecting the *expand all* or *collapse all* point-and-shoot fields at the top of the display. The expand/collapse state of any note is common between the User Note List display and the Dump Storage display, so that any changes made in one display are reflected in the other.

To display the storage associated with a user note, place the cursor on the address point-and-shoot field, and press Enter. User notes are saved in the history file fault entry when the user exits from the interactive report. At this time, if user notes have been added or modified, the user is prompted to acknowledge the update of the fault entry with a display.

Pressing Enter from the display permits the history file fault entry to proceed, while pressing PF3 or PF12 will cause exit from the interactive report without any updates to user notes.

18.14 Mapping storage areas using DSECT information

By using the DSECT command from within the interactive report, storage areas can be mapped based on PDS(E) data set members containing assembler macro or DSECT copybooks. The DSECT command can be entered from the command line of any display, or via PF key assignment. By default, the DSECT command is assigned to PF4.

> **Note:** If the DSECT command is issued while the cursor is on an address point-and-shoot field, the pop-up display address is automatically initialized to that address.

You can now supply the start address (if not already filled in), and the name of the DSECT to use when mapping the storage area. The name of the DSECT can be provided in one of two ways:

1. Just the DSECT name can be entered, in which case the IDIDSECT concatenation is searched for a match. If multiple occurrences of the requested DSECT exist in the IDIDSECT concatenation, then press PF4 and select the appropriate one from the resulting list of all available DSECTs. A DSECT is selected from the list using an **S**, or it can be edited by entering an **E**.
2. The PDS(E) data set and member name, where the specified DSECT is stored, can be supplied. If a DSECT name is not provided, then it defaults to be the same as the PDS(E) member name.

Once a valid storage address and DSECT name have been specified, then the storage will be mapped and displayed as shown in Figure 18-52.

```
  File  View  Services  Help
 ┌──────────────────────────── Storage DSECT Mapping ────────────────────────────
 │
 │ Enter the name of the Dsect in the Dsect Name field to be used to map the
 │ storage address provided in the Address field. Press PF4 to display a list
 │ of all available Dsects.  Optionally a specific Dsect can be used by
 │ supplying a Dataset and Member name in the DSN field.  In this case if a
 │ Dsect name is not provided it will be made equal to the member name.
 │
 │ Address    0C16C0D0
 │ Dsect Name _____________________________________________________________
 │ DSN  . . . ______________________________________________________
 │
 │
 │  F1=Help     F3=Exit    F12=Cancel
 └──────────────────────────────────────────────────────────────────────────────
        Address range 0C175CB0-0C1805D7 not in minidump
        Module IDIXDCAP
        Address range 0C1805D8-0C18BFFF not in minidump
0C18C000    +1FF30  90000000 00400012 00000000 5000010C *..... ......&...*
0C18C010    +1FF40  FFFFE1A0 38260000 40080104 00000000 *........ .......*
0C18C020    +1FF50  0008A497 978599D4 A2870000 00000000 *..upperMsg......*
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F7=Up        F8=Down
F10=Prev     F11=Next
```

Figure 18-52 Storage DSECT Mapping display

Scroll up, down, right, and left as needed to display additional DSECT information.

Press PF3 to return from the Storage DSECT Mapping display.

18.14.1 IDIDSECT concatenation

The IDIDSECT concatenation can optionally be specified in the data sets sections of your options data set. If specified, then it should specify the name of one or more PDS(E) data sets, which contain DSECT files to be used when processing the DSECT command.

When in an interactive report, the first time the DSECT command is issued, it processes each data set in the IDIDSECT concatenation. If the data set contains a $DINDEX member, then the DSECT details in this member are used. Otherwise, each member in the data set is assumed to contain a DSECT of the same name.

Once all the DSECT details have been determined for a data set, then the process is repeated for the next data set, until all the data sets in the IDIDSECT concatenation have been processed. This process only happens once per interactive report session, and hence, if new DSECTs are added to a data set in the IDIDSECT concatenation, or if the $DINDEX member is updated, then it will be necessary to either restart the interactive report, or explicitly identify the new DSECT by specifying the data set and member name in which it is contained.

18.14.2 Indexing

To allow for DSECT names of up to the maximum of 63 characters, and individual PDS(E) members containing multiple DSECTS, a $DINDEX member can be created.

The $DINDEX member (which can be created using the IDIPDSCU utility) should contain a line for every DSECT in each member of the PDS(E). The format of each line should be the DSECT name, followed by a space, followed by the member in which that DSECT is found, as shown in Example 18-1.

Example 18-1 Indexing example

```
DSECT1 MEMBER1
DSECT2 MEMBER1
LONGDSECTNAME1 MEMBER2
LONGDSECTNAME2 MEMBER2

In this example, PDS(E) member MEMBER1 contains DSECTs DSECT1 and DSECT2, and
MEMBER2 contains DSECTs LONGDSECTNAME1 and LONGDSECTNAME2.
```

18.14.3 DSECT indexing utility

The IDIPDSCU utility is used to create a $DINDEX member for a given data set. It does this by calling the assembler for each member in the data set, and extracting the imbedded DSECTs from the assembler output.

In situations where DSECT or macro expansions require special keyword specifications, separate members might have to be coded. These members will need to call the macro in question, providing the required keywords, and will need to be stored in the same data set as the macro they invoke.

CICS provides in its SDFHMAC data set a member called DFHTCTZE, which provides multiple terminal-related DSECTs. If this member is processed directly by the IDIPDSCU utility, it will not detect the TCTENIB DSECT because this requires special macro keywords to

be specified. In this case, if a member is created in the SDFHMAC data set (or a copy of it), which contains the following source line, then all DSECTs will be detected, including:

```
TCTENIB:
DFHTCTZE CICSYST=YES
```

The IDIPDSCU utility can be used either by entering IDIPDSCU next to a data set name in ISPF, or as a batch utility, in which case the data set to process is passed as a parameter as shown in Example 18-2.

Example 18-2 IDIPDSCU utility

```
//UTILJOB1 JOB ...
//RUNUTIL EXEC PGM=IDIPDSCU,PARM=('fully_qualified_PDS(E)_dataset_name')
//STEPLIB DD DISP=SHR,DSN=IDI.SIDIMOD1
//SYSPRINT DD SYSOUT=*
```

The IDIPDSCU utility will create a $DINDEX member in the target data set, so you must have write access to this data set.

18.15 Displaying chained data areas

Storage can be scanned for a chain of linked control blocks. The RUNCHAIN command can be invoked either by entering RUNCHAIN on any interactive report command line, or by assigning RUNCHAIN to a PF key. When invoked, you will be shown a pop-up panel similar to Figure 18-53.

```
  File  View  Services  Help
 ┌──────────────────────── Storage RUNCHAIN Command ────────────────────────
 │
 │ Enter the required fields and press Enter .
 │
 │ Start Address  . . . . . . . 0C16C0D0
 │ Max Number Control Blocks    9999  (Decimal)
 │ Forward Pointer Offset . . . ____  (Hex)
 │ End of Chain Identifier  . . ________  (Hex, Default Values 00000000,
 │                                        FFFFFFFF)
 │ Eyecatcher Text  . . . . . . ____________________
 │ Eyecatcher Offset  . . . . . ____  (Hex)
 │
 │
 │  F1=Help     F3=Exit    F12=Cancel
 └──────────────────────────────────────────────────────────────────────────
         Address range 0C175CB0-0C1805D7 not in minidump
         Module IDIXDCAP
         Address range 0C1805D8-0C18BFFF not in minidump
0C18C000    +1FF30  90000000 00400012 00000000 5000010C *..... ......&...*
0C18C010    +1FF40  FFFFE1A0 38260000 40080104 00000000 *........ .......*
0C18C020    +1FF50  0008A497 978599D4 A2870000 00000000 *..upperMsg......*
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F7=Up        F8=Down
F10=Prev     F11=Next
```

Figure 18-53 Storage RUNCHAIN Command entry display

For a given Start Address and Forward Pointer Offset, the RUNCHAIN command will follow the chain of control blocks until one of the following end conditions is met:

1. The number of control blocks scanned has exceeded the maximum number set by the user (the default value is 9999).

2. The forward pointer of the current control block contains one of the End-of-Chain values. These values are:
   - X’00000000’
   - X’FFFFFFFF’
   - The initial start address’ implying the chain has looped
   - A user-supplied End-of-Chain identifier
3. The forward pointer of the current control block points to invalid or unavailable storage.

For each control block, its address and the first 32 bytes of data are shown. Optionally, you can provide an eye catcher and its offset in the control block, in which case, for each control block, the text at the specified offset is compared against the supplied text, and if they do not match, then a warning message is issued.

To exit from the RUNCHAIN command, enter Exit (PF3).

18.16 Disassembling object code

The DISASM command disassembles object code at a given address.

The DISASM command is invoked either by entering DISASM on any interactive report command line, or by assigning DISASM to a PF key. When invoked, you will be shown a pop-up panel similar to Figure 18-54.

```
  File  View  Services  Help
 ┌──────────────────────────── Storage Disassemble ────────────────────────────┐
 │                                                                             │
 │ WARNING Before using this function you must be aware of and respect the     │
 │ intellectual property rights of others. You are not authorized to use this  │
 │ function to disassemble, copy or create assembly listings or disassembled   │
 │ Assembler Language source code in violation of any contractual or other     │
 │ legal obligation. You are authorized to use this function only for object   │
 │ code for which you have verified you have the right to perform              │
 │ disassembly.                                                                │
 │                                                                             │
 │ Start Address  . . . . . . . . . 0C148448                                  │
 │ Origin Address (optional)  . . . ________                                   │
 │                                                                             │
 │                                                                             │
 │  F1=Help     F3=Exit    F12=Cancel                                          │
 └─────────────────────────────────────────────────────────────────────────────┘
0C1484F0       +A8  00000000 00000000 00000000 00000000 *................*
        Lines 0C148500-0C148520 same as above
0C148530       +E8  0BF00E2C 00000000 0C14857C 0C1483F0 *.0........e@..c0*
0C148540       +F8  0BF00EDF 00000000 0BF00D30 0BF00E68 *.0.......0...0..*
0C148550      +108  0C14857C 0BF00E58 00000000 0C16C0D0 *..e@.0........{}*
 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F7=Up        F8=Down
F10=Prev     F11=Next
```

Figure 18-54 Storage Disassemble display

Optionally, an origin address can be provided, in which case the offset of each disassembled instruction is calculated relative to the origin address, rather than the start address. If an origin address is not provided, then it defaults to the same as the start address.

PF7 and PF8 can be used to scroll backwards and forwards. To exit from the DISASM command, enter Exit (PF3).

18.17 Converting STORE CLOCK values

The STCK command converts binary STORE CLOCK values to human-readable date and time format. The STCK command is invoked either by entering STCK on any interactive report command line, or by assigning STCK to a PF key. When invoked, you will be shown a panel similar to Figure 18-55.

```
File  View  Services  Help
┌──────────────────────────────── STCK Conversion ────────────────────────────────
│
│ Enter the 16 hex character STORE CLOCK (STCK) value in the field and press
│ ENTER to display its Date Time value.
│
│ STCK Value   0986888F
│ Date Time  :
│ The STCK value should be 16 hex characters.
│
│  F1=Help     F3=Exit    F12=Cancel
└──────────────────────────────────────────────────────────────────────────────
  0 E6D6D9D2 C9D5C760 E2E3D6D9 C1C7C540 *WORKING-STORAGE *  01  FILLER
 10 40404040                            *               *
 18 0986888F                            986888             01  NUMBERX PIC 9
                                                           01  ERROR-FLD.
 20 C1C2C3C4                            *ABCD           *      05  ERROR-COU
                                                               05  FLDY REDE
 20 C1C2C3C4                            *ABCD           *          07 FLDZ P
 28 0000                                *..             *  01  BAD-RESULT PI


F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
F8=Down     F10=Left     F11=Right
```

Figure 18-55 STCK Conversion display

The STCK value must be entered as 16 hexadecimal characters. Any imbedded blanks are ignored.

To exit from the STCK command, enter Exit (PF3).

18.18 User-specific report formatting

The EXEC command, a REXX formatting user exit can be executed. This type of exit is able to generate a display of user-specific information, such as formatting of data areas that are unique to the analyzed application environment.

Non-REXX formatting user exits cannot be executed through the EXEC command. If an exit name is specified with the EXEC command, then the exit is executed and a display containing any information is presented.

However, if no exit name is specified, then a list of all REXX formatting user exits, available in the IDIEXEC concatenation, is presented in a Formatting User Exit Selection List display.

To be recognized as a Formatting user exit for the Formatting User Exit Selection List display, the exit names must be specified in a control member ($$UFMTX) within each IDIEXEC data set.

Each record in the control member can contain the member name of the exit (case insensitive) and optionally a comment which is included in the Formatting User Exit Selection List display. A sample control member is shown in Figure 18-56.

```
  File  View  Services  Help
 ------------------------------------------------------------------------
 Formatting User Exit Selection List                          Line 1 Col 1 80
 Command ===> _____________________________________________ Scroll ===> CSR
 JOBNAME: $TSTFACB  SYSTEM ABEND: 0C7          STLABF6   2004/11/03  13:14:50

 {The following line commands are available: S (Select), B (Browse), E (Edit).}

    Name     Comment/Arguments
 __ IDISUFM1 Sample Formatting user exit to display TCB information
 __ IDISUFM2 Sample Formatting user exit for CICS CWA
 __ IDISUFM3 Sample Formatting user exit to illustrate the use of formatting ta

 *** Bottom of data.





  F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
  F8=Down     F10=Left     F11=Right
```

Figure 18-56 $$UFMTX member

The sample control member is provided in softcopy format as member IDISUFMX in data set IDI.SIDISAM1.

To use this sample control member, do the following:

1. Copy it to another data set and rename it to $$UFMTX.
2. Copy the three sample Formatting user exits named within it to the same data set.
3. Specify the data set name now containing the control member and the exits in the DATASETS(IDIEXEC(*data-set-name*)) option.
4. Invoke the Fault Analyzer ISPF interface and perform interactive reanalysis against a fault entry.
5. Issue the EXEC command without specifying an exit name.

From the Formatting User Exit Selection List display, a line command can be issued against individual exits as listed in Table 18-1.

Table 18-1 Exits

Selection	Selection description
S	Executes the exit
B	Enter ISPF browse against the exit
E	Enter ISPF edit against the exit

Often, exits require one or more parameters to be passed.

This can be done by clearing (if necessary) and overtyping the Comments/Arguments field of the Formatting User Exit Selection List display to the right of the exit name. This will cause a color change of the field when it has been overtyped, indicating that the data will be used as parameters for the exit.

By clearing the field, and pressing Enter, the original comment can be redisplayed.

18.19 Prompting for compiler listing or side file

If no satisfactory compiler listing or side file was found for a COBOL or PL/I program, then a prompt will be displayed as shown in Figure 18-57.

```
 Options  View  Help
                         Compiler Listing Not Found
 Command ===> ____________________________________________________________

 Compiler listing or side file not found for:
  Program Name . . . : IDISCBL1
  Language . . . . . : COBOL
  Compile Date . . . : 2001/10/18  (YYYY/MM/DD)
  Compile Time . . . : n/a       (HH:MM:SS)

 Select one of the following options and press Enter:
 __  1.  Continue without compiler listing or side file for this program
     2.  Specify compiler listing or side file to use for this program
     3.  Retry search for compiler listing or side file for this program
     4.  Do not prompt again for any missing listing or side file
     5.  Only prompt for the point of failure program listing or side file
 This is the point of failure program.  1

  F1=Help     F3=Exit    F12=Cancel

    F00065 IDIVPCOB IBMUSER  MVS2     S0C7  2001/09/24 13:41:06
    F00064 IDIVPCOB IBMUSER  MVS2     S0C7  2001/09/24 11:20:20
F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 18-57 Compiler Listing Not Found display

The prompt provides you with these choices:

1. Continue without compiler listing or side file for this program

   If a compiler listing or side file cannot be supplied, select this option to continue without program source code information. Alternatively, enter the Exit (PF3) or Cancel (PF12) command.

2. Specify compiler listing or side file to use for this program

   This option will display a pop-up panel in which you can provide the data set and member name (if a PDS(E) data set) of a compiler listing or side file that should be used for the current program, as shown in Figure 18-58.

   The data set name is specified in accordance with the ISPF convention of prefixing with the current TSO prefix, unless enclosed in single quotes. The member name defaults to the program name for which the listing or side file is required. If the actual member name for your listing or side file differs from the program name, you will need to change this field. If a sequential data set is specified, then the member name will be ignored. Having specified the desired data set and member name, press Enter.

```
 Options  View  Help
 ------------------------- Compiler Listing Not Found -------------------------
 Command ===> ____________________________________________________________________
   ------------------- Specify Compiler Listing or Side File -------------------
 C  Command ===> ____________________________________________________________
    |                                                                          |
    | Specify the data set and member name containing the compiler listing     |
    | or side file and press Enter.                                            |
    |                                                                          |
    | Data set name  . . . ______________________________________________      |
 S  | Member . . . . . . . IDISCBL1                                            |
 2  |                                                                          |
    ----------------------------------------------------------------------------
    F00068 FRED     CICSUSER CICS04   AEIL  2001/09/25 12:11:46
    F00067 FRED     CICSUSER CICS04   AEIL  2001/09/25 11:59:57
    F00066 FRED     CICSUSER CICS04   AEIL  2001/09/25 11:06:56
    F00065 IDIVPCOB IBMUSER  MVS2     S0C7  2001/09/24 13:41:06
    F00064 IDIVPCOB IBMUSER  MVS2     S0C7  2001/09/24 11:20:20
 F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 18-58 Specify Compiler Listing or Side File display

3. Retry search for compiler listing or side file for this program

   Selecting this option will cause Fault Analyzer to repeat the search for the compiler listing or side file via the standard search path. This option can be selected after, for example, having recompiled the current program via a split screen ISPF session, and providing the compiler listing or side file to Fault Analyzer in, for example, the IDILCOB data set concatenation. This repeated search is only performed once. The user will not be prompted a second time for the same program, even if the listing or side file is still not found.

4. Do not prompt again for any missing listing or side file

   If you select this option, then Fault Analyzer will not prompt you again for a missing compiler listing or side file for any program for the duration of the current interactive reanalysis session.

5. Only prompt for the point of failure program listing or side file

   If you select this option, then Fault Analyzer will only prompt you again for a missing compiler listing or side file for a program, if that program has been determined as belonging to the point-of-failure event. If the initial prompt is already for the point-of-failure program, then a message is added to the display to indicate this [1].

18.19.1 Controlling prompting

In order for the compiler listing or side file prompting to occur during interactive reanalysis, at least one IDILANGX or compiler listing data set (for example, IDILCOB) must have been provided.

It follows that, if source analysis is never to be performed, and therefore no prompting during interactive reanalysis is desired, then no IDILANGX or compiler listing data set definitions should be specified through DataSets options in the IDICNF00 config member.

On the other hand, to ensure that prompting always occurs during interactive reanalysis if no satisfactory compiler listing or side file is found, regardless of whether any compiler listing or side file data sets were specified for the real-time analysis, add at least one IDILANGX or compiler listing data set definition through the DataSets option in the IDICNF00 config member.

18.20 Data sets used for interactive reanalysis

When performing interactive reanalysis through the ISPF interface, pre-allocation will be performed as required for any Fault Analyzer compiler listing or side file data sets that were used in real-time. Allocations will be performed for Fault Analyzer data sets if they were explicitly included in the real-time JCL, or supplied through the DataSets option or an Analysis Control user exit. These data sets are used in the reanalysis in an attempt to recreate the same execution environment as was used in real-time.

DataSets options specified via the IDIOPTS user options file or the PARM field will cause those data sets to be logically concatenated to the data sets from the real-time execution.

If the Display panel to alter allocated data sets option on the Interactive options display is set to Y, then it is possible to make changes to the real-time data set specifications before initiating the reanalysis. Also, any data sets that were used in real-time but that do not exist in the reanalysis environment are identified by a comment as shown in the following example for IDILCOB:

```
//* The following IDILCOB data set is unavailable:
//* DD DISP=SHR,DSN=D01.COBOL.LISTINGS
```

19

Fault Analyzer real-time analysis report

The real-time analysis report is produced whenever Fault Analyzer analyzes an abend. It is included as part of the job output. This chapter describes how to control what is included in this report.

19.1 Successful analysis processing

If Fault Analyzer deems the analysis to be successful, it will suppress the writing of a SYSABEND, SYSUDUMP, or SYSMDUMP (if one is allocated to the abending job step and if Fault Analyzer was invoked using the IEAVTABX MVS change options/suppress dump exit, IDIXDCAP), unless the RetainDump(ALL) option is in effect.

The criteria for success are:

- There is one unambiguous point of failure.
- The source line of code has been identified.

This means that if you do not provide a compiler listing or side file for the abending program, then Fault Analyzer will not be able to deem the analysis a success, and so will not suppress the writing of a dump. However, subsequent fault reanalysis is still possible using the saved minidump.

19.2 Fault history file selection

When a fault is being analyzed in real-time by Fault Analyzer, a fault history file must be available in which details of the analysis can be recorded. There are a number of ways in which the name of the fault history file can be provided to Fault Analyzer.

The following is a list of these in the order of their override significance (each entry in the list overrides all previous entries):

- The product default name, IDI.HIST.
- The IDIHIST suboption of a DataSets option specified in the parmlib config member, IDICNF00. This includes either the logical parmlib concatenation or the installation-wide alternate parmlib data set name provided via the IDISCNF USERMOD supplied with Fault Analyzer.
- The IDIHIST suboption of a DataSets option specified in a config member identified via the IDICNFUM user options module.

> **Note:** If a user options module is used, it replaces the default IDICNF00 parmlib config member. Thus, even if the user options module designated config member did not include an IDIHIST suboption of a DataSets option, any specification of IDIHIST in the default IDICNF00 parmlib config member would not be recognized.

- The IDIHIST suboption of a DataSets option provided via the IDIOPTS DDname in the abending job step.
- An explicitly coded IDIHIST DD statement in the abending job step.
- The data set name provided by an Analysis Control user exit in the CTL data area IDIHIST field.

19.3 Controlling the real-time analysis with options

You can set options globally, so they control the output for all jobs. However, you can also set an option just for one job. In this case, you should set the option in the user options file.

Options that you are more likely to use for real-time analysis are:

RetainDump(ALL)

Specify this option to retain the SYSABEND, SYSUDUMP, or SYSMDUMP unconditionally. Fault Analyzer permits the writing of a dump, and records the name of the dump data set in the fault history file if a SYSMDUMP DD statement was specified.

Without this option, many dumps will be suppressed when Fault Analyzer deems that the performed analysis is adequate. This option does not affect the writing of the minidump to the history file. The dump disposition part of this option is applicable to the use of the MVS IEAVTABX change options/suppress dump exit only.

Detail

This option adjusts the level of detail given in the real-time analysis report. If a dump is produced, you may change this option for reanalysis.

Exclude

This option will exclude the job from analysis.

You can also use the DataSets option to point to listings and side files.

19.4 Pointing to listings with JCL DD statements

No DD statements are required to run Fault Analyzer in either batch or real time, although a SYSMDUMP DD statement is needed for normal SYSMDUMP processing in real time when using the MVS IEAVTABX change options/suppress dump exit, unless the IDITABD USERMOD is applied.

You can specify the following DD statements in the JCL if appropriate. If they are not specified, the definitions from the PARMLIB configuration member IDICNF00, the IDIOPTS user options file, or an Analysis Control user exit are used to identify these data sets:

- IDILC: PDS(E) data set containing C compiler listings
- IDILCOB: PDS(E) data set containing COBOL compiler listings (other than OS/VS COBOL)
- IDILCOBO: PDS(E) data set containing OS/VS COBOL compiler listings
- IDILPLI: PDS(E) data set containing PL/I compiler listings (other than Enterprise PL/I)
- IDILPLIE: PDS(E) data set containing Enterprise PL/I compiler listings
- IDIADATA: PDS(E) data set containing SYSADATA from Assembler compilations
- IDILANGX: PDS(E) data set containing IDILANGX side files for all languages

Do not specify a member name on any of these DD statements.

19.5 Real-time analysis report

The real-time analysis report is produced whenever Fault Analyzer analyzes an abend. It is included as part of the job output.

You can specify the IDIREPRT DD statement in your JCL. This indicates where you want the Fault Analyzer report written. If it is not present in the job step JCL, then it is dynamically allocated with SYSOUT=* destination.

To disable the IDIREPRT automatic allocation, you can use an IDIREPRT DD DUMMY JCL statement in the abending job step. Alternatively, an Analysis Control user exit can be used to allocate IDIREPRT to DUMMY (for example, as a REXX EXEC using the IDIALLOC command).

If IDIREPRT is allocated as dummy, then reports are written to the fault history file only.

For long-running jobs or started tasks, such as CICS, IMS MPRs, or WLM-managed DB2, it can be advantageous to have the Fault Analyzer reports spooled to JES as individual files at the time when the report writing has completed, as opposed to having potentially numerous reports concatenated in one file, that is, spooled only when the abending job step ends. This can be done using a Fault Analyzer Analysis Control user exit.

19.6 SYSLOG summary

During real-time analysis, a message is written to the operator console providing a one-line summary of the fault reason.

As an example, the following message can be produced on the operator console:

```
IDI0002I There was an unsuccessful REWRITE of file MYFILE01 in program COBFERRD at line
# 21
```

If the Quiet option is in effect, then this message might not be written to the SYSLOG.

19.7 Using the program SNAP interface (IDISNAP)

A program SNAP interface is provided to assist users in debugging problems with applications that do not abend, or that for any other reason cannot be analyzed by Fault Analyzer using one of the normal abend invocation exits.

This permits a call to Fault Analyzer from anywhere within an application program to request an analysis of the current environment. The program SNAP interface module name is IDISNAP.

An example of where a call to IDISNAP would be in a DB2 application after execution of a SQL statement that results in a negative SQLCODE.

Apart from the way in which Fault Analyzer is invoked, there is no difference between this type of analysis and any other real-time analysis caused by an abend. It is recommended that you invoke IDISNAP dynamically to ensure that you are always using the most current version.

19.8 Dump registration processing

Unlike the SYSABEND, SYSMDUMP, and SYSUDUMP processes, which run in the user address space, the SVC dump process in MVS runs from the DUMPSRV address space.

This difference means that the MVS change options/suppress dump exit, which is one of the normal means of invoking Fault Analyzer, does not work for SVC dumps.

For SVC dumps, Fault Analyzer provides the IDIXTSEL exit module. SVC dumps occur for system abends, and are also used by CICS for its system dumps. If the IEAVTSEL post-dump

exit, IDIXTSEL, has been installed, then a skeleton fault entry is created whenever an SVC dump is written.

This differs from normal real-time processing in that no analysis is performed, and therefore no report or minidump is produced.

This Fault Analyzer process is known as dump registration.

The dump registration processing permits the use of two user exits which effectively are the equivalent of the normal Analysis Control and Notification user exits. These are specified using the DumpRegistrationExits option.

The dump registration fault entry contains only limited information, such as the time of its creation, the system name, and the name of the job which caused the SVC dump to be written. If available, the abend code and abending program name is also provided.

However, the first reanalysis of the dump registration fault entry will refresh the fault entry and save a report and minidump with it. If the first reanalysis to refresh the fault entry is performed interactively, then a panel is displayed upon exiting the interactive report, which permits the cancellation of the refresh, or the suppression of the minidump. If instead performing the reanalysis in batch, then a user exit can be used to perform the equivalent function. This exit is effectively an End Processing user exit, and is specified using the RefreshExits option.

20

Fault Analyzer batch reanalysis report

The batch reanalysis report might be used when a user wants to either re-run analysis on a number of faults or would prefer the reanalysis to run in batch rather than hold up the TSO session. The batch report is directed to the SYSPRINT DD statement of the batch job step. The batch reanalysis report looks the same as the real-time analysis report.

20.1 Batch reanalysis options

To specify general batch reanalysis options that apply to your batch jobs only, first select Batch Reanalysis Options from the Fault Entry List Display Options menu. The panels shown in Figure 20-1 and Figure 20-2 are displayed.

```
  File  View  Services  Help
 ------------------------------------------------------------------------
 Batch Reanalysis Options                                   Line 1 Col 1 80
 Command ===> _____________________________________________ Scroll ===> CSR

 Press PF3 to save options or PF12 to cancel.

 General Options:
   Options line for batch
   reanalysis. . . . . . . . : ____________________________________________
   Redisplay this panel
   before each reanalysis. . : N (Y/N)
   Display panel to edit
   generated JCL . . . . . . : Y (Y/N)
   Job card style. . . . . . : P (P=Parameters, S=Statements)

 Job Card Parameters:
   Job name suffix . . . . . : A (A-Z, 0-9, @, #, or $)
   Job class . . . . . . . . : A (A-Z or 0-9)
   Job notify. . . . . . . . : Y (Y/N)
   Job time minutes. . . . . : 10 (0-99)
   Message class . . . . . . : X (A-Z or 0-9)
  F1=Help      F3=Exit      F5=RptFind   F6=Actions   F7=Up        F8=Down
 F10=Left     F11=Right    F12=Cancel
```

Figure 20-1 Batch Reanalysis Options display (Part 1 of 2)

```
  File  View  Services  Help
 ------------------------------------------------------------------------
 Batch Reanalysis Options                                   Line 16 Col 1 80
 Command ===> _____________________________________________ Scroll ===> CSR
   Job notify. . . . . . . . : Y (Y/N)
   Job time minutes. . . . . : 10 (0-99)
   Message class . . . . . . : X (A-Z or 0-9)
   Region megabytes. . . . . : 0   (0-2047)
   Accounting info . . . . . : ____________________________________________

 Reanalysis Report:
   Destination . . . . . . . : ______________________________________

 Reanalysis Options Data Set Control:
   Options data set name . . : ______________________________________________
   Options member name . . . : ________ (If PDS or PDSE)
   Use this data set during
   reanalysis. . . . . . . . : N (Y/N)
   Edit the options data set
   before reanalysis . . . . : N (Y/N)

 *** Bottom of data.
  F1=Help      F3=Exit      F5=RptFind   F6=Actions   F7=Up        F8=Down
 F10=Left     F11=Right    F12=Cancel
```

Figure 20-2 Batch Reanalysis Options display (Part 2 of 2)

The following fields are of particular interest:

- Options line for batch reanalysis
  - These options, which will be used in the PARM field of the generated batch reanalysis job, take precedence over any options specified through an options file. Options specification will fit on this line.
  - Use one of the options:
    - Display panel to edit generated JCL
    - Edit the options data set before reanalysis
  - Options specified on the options line are saved in the user profile.
- Redisplay this panel before each reanalysis

  If this option is set to Y, the Batch Reanalysis Options display will be shown each time a batch reanalysis is requested, prior to generating the JCL stream.
- Display panel to edit generated JCL

  If this option is set to Y, you will be presented with an ISPF EDIT display screen of the JCL stream generated by Fault Analyzer, as shown in Figure 20-3.

  Type the `SUBMIT` command to execute the job; then issue an Exit (PF3) or Cancel (PF12) command to return to the Fault Entry List display.

  If this option is set to N, then the generated JCL stream is submitted automatically without first displaying the JCL EDIT screen.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------------
EDIT       SYS04317.T072621.RA000.GRACINE.R0100016         Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
****** ***************************** Top of Data ******************************
000001 //GRACINEA JOB (),'FA - $TSTFACB',
000002 // CLASS=A,MSGCLASS=X,TIME=10,NOTIFY=GRACINE,REGION=0M
000003 //RUNDA    EXEC PGM=IDIDA,
000004 // PARM=('/FAULTID(F00001)',
000005 //      )
000006 //STEPLIB  DD DISP=SHR,DSN=CEE.SCEERUN
000007 //IDIHIST  DD DISP=SHR,DSN=IDI.HIST.TEST
000008 //IDILCOB  DD DISP=SHR,DSN=GRACINE.IDI.IVPCB.LISTINGS
000009 //SYSPRINT DD SYSOUT=*
****** **************************** Bottom of Data ****************************
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 20-3 batch reanalysis JCL stream EDIT

- Job card style

  This is a single character (P or S) that controls the style of job card specification, shown in Figure 20-4, as follows:
  - If P is specified (the default), a Job Card Parameters section will follow the General Options section after the user presses Enter.
  - If S is specified, a Job Card Statements section will follow the General Options section after the user presses Enter.

```
  File  View  Services  Help
 ------------------------------------------------------------------------
Batch Reanalysis Options                                   Line 13 Col 1 80
Command ===> ____________________________________________  Scroll ===> CSR
Job Card Statements:
  ===> ____________________________________________________________________
  ===> ____________________________________________________________________
  ===> ____________________________________________________________________
  ===> ____________________________________________________________________

Reanalysis Report:
  Destination . . . . . . . : ____________________________________________

Reanalysis Options Data Set Control:
  Options data set name . . : ____________________________________________
  Options member name . . . : ________ (If PDS or PDSE)
  Use this data set during
  reanalysis. . . . . . . . : N (Y/N)
  Edit the options data set
  before reanalysis . . . . : N (Y/N)

*** Bottom of data.
 F1=Help      F3=Exit      F5=RptFind   F6=Actions   F7=Up        F8=Down
F10=Left     F11=Right    F12=Cancel
```

Figure 20-4 Batch Reanalysis Options display

After you have made your changes to the options data set (if any), issue an Exit command (usually mapped to PF3).

20.2 Initiating batch reanalysis

To initiate batch reanalysis, enter **B** against the fault history entry. Depending on your batch options, one or more displays might be shown prior to the submission of the Fault Analyzer generated JCL stream.

20.3 Data sets used for batch reanalysis

When performing batch reanalysis through the ISPF interface, the generated JCL will include DD statements as required for any JOBLIB, STEPLIB, or Fault Analyzer compiler listing or side file data sets. DD statements will be added for Fault Analyzer data sets even if they were not explicitly included in the real-time JCL, but were supplied through the DataSets option or an Analysis Control user exit. These data sets are added to the reanalysis job in an attempt to recreate the same execution environment as was used in real-time.

Data set options specified via the IDIOPTS user options file or the PARM field will cause those data sets to be logically concatenated to the data sets from the real-time execution. If the Display panel to edit generated JCL option on the Batch Reanalysis Options display is set to Y, then it is possible to make changes to the real-time data set specifications before submitting the reanalysis job. Also, any data sets that were used in real-time but do not exist in the reanalysis environment are identified by a comment as shown in Example 20-1.

Example 20-1 IDILCOB

```
//IDILCOB DD DISP=SHR,DSN=GRACINE.DUMPA.LISTING.CICS.COBOL
// DD DISP=SHR,DSN=GRACINE.DA.LISTING.COBOL
//* The following IDILCOBO data set is unavailable:
//* DD DISP=SHR,DSN=GRACINE.DUMPA.LISTING.CICS.COBOSVS
//* The following IDILPLI data set is unavailable:
//* DD DISP=SHR,DSN=GRACINE.DUMPA.LISTING.CICS.PLI
```

20.4 Creating your own batch reanalysis job

Batch mode may also be invoked via JCL that you have created or saved from a job previously generated by Fault Analyzer and later modified if necessary. As identification of the fault to be reanalyzed, you need to specify either a fault ID (using the FaultID option) or a SYSMDUMP or SVC dump data set name (using the DumpDSN option), as shown in Example 20-2.

Example 20-2 Batch job example

```
//RTURNERA JOB (),íFAULT ANALYZERí,CLASS=A,MSGCLASS=X,NOTIFY=&SYSUID
//*
//* Allocate a PDSE for compiler listings
//*
//ALLOC EXEC PGM=IEFBR14
//DD1 DD DISP=(,CATLG),DSN=&SYSUID..COBLIST,SPACE=(CYL,(1,1,5)),
// DCB=(RECFM=FBA,LRECL=133),DSNTYPE=LIBRARY
//*
//* Recompile MYPGMA
//*
//CBLCOMP EXEC IGYWC,PARM.COBOL=íLIST,MAP,Source,XREFí
//COBOL.SYSIN DD DISP=SHR,DSN=MY.COBOL.Source(MYPGMA)
//COBOL.SYSPRINT DD DISP=SHR,DSN=&SYSUID..COBLIST(MYPGMA)
//*
//* Recompile MYPGMB
//*
//CBLCOMP EXEC IGYWC,PARM.COBOL=íLIST,MAP,Source,XREFí
//COBOL.SYSIN DD DISP=SHR,DSN=MY.COBOL.Source(MYPGMB)
//COBOL.SYSPRINT DD DISP=SHR,DSN=&SYSUID..COBLIST(MYPGMB)
//*
//* Reanalyze SYSMDUMP data set
//*
//RUNDA EXEC PGM=IDIDA,PARM=(í\/DumpDSN(MY.DUMPDS)í)
//SYSPRINT DD SYSOUT=*
//IDIHIST DD DISP=SHR,DSN=MY.HISTORY.FILE
//IDILCOB DD DISP=SHR,DSN=&SYSUID..COBLIST
//*
//* Delete temporary compiler listings PDSE
//*
//DELETE EXEC PGM=IEFBR14
//DD1 DD DISP=(OLD,DELETE),DSN=&SYSUID..COBLIST
```

Example 20-2 includes the recompilation of two COBOL programs, MYPGMA and MYPGMB.

These are assumed to be involved in the fault being reanalyzed, but their compiler listings might not have been available to Fault Analyzer during real-time analysis.

Providing them to the fault reanalysis will enable identification of the line of source code where the error occurred. Note that only one program is compiled in each job step. This is to facilitate the naming of the compiler listing data set member in accordance with the rules outlined.

You can optionally add a IDIOPTS DD statement to your JCL. This statement supplies the name of a sequential file containing Fault Analyzer options, providing job step overrides of product and installation defaults. Other DD statements that you can add into your JCL are described in 19.4, “Pointing to listings with JCL DD statements” on page 471. Any options specified in the JCL EXEC statement PARM field override options set via the IDIOPTS file.

21

Fault Analyzer and subsystems

This chapter describes how Fault Analyzer interacts with various subsystems. It explains how to set up and customize FA with CICS, DB2, and IMS, as well as how to use FA to perform WebSphere and Java dump analysis.

21.1 Fault Analyzer and CICS

We recommend that you install the resources identified in Table 21-1 using the steps described in IBM Fault Analyzer for z/OS and OS/390 User's Guide and Reference, SC18-9374-00, under “Customizing the CICS environment.”

Table 21-1 CICS customization

NAME	TYPE	GROUP
IDIPLT	PROGRAM	IDI
IDIXCCEE	PROGRAM	IDI
IDIXCX52	PROGRAM	IDI
IDIXCX53	PROGRAM	IDI
IDIXFA	PROGRAM	IDI
CFA	TRANSACTION	IDI

Fault Analyzer can be invoked under CICS in the following three ways:

1. **XPCABND** Global user exit using program IDIXCX52 or IDIXCX53. This is the main exit provided to invoke Fault Analyzer for CICS transaction fault analysis.
2. **XDUREQ** Global user exit using program IDIXCX52 or IDIXCX53. This exit can be used to invoke Fault Analyzer for CICS dumps generated from an EXEC CICS DUMP command. The analysis performed by Fault Analyzer at this exit point is the same as for the XPCABND exit point.
3. **LE Exit** LE abnormal termination exit using program IDIXCCEE. This exit is only effective with Language Environment-based application programs when the CEECXTAN exit has been set.

   CICS AKCS abends can be analyzed using this exit if the failing program is LE enabled *and* an entry exists in the CICS dump table for AKCS, specifying that a transaction dump is to be taken.

The first two exits are CICS global user exit points, and Fault Analyzer is enabled and disabled at these points using CICS calls. This means that, by default, these exit points are not enabled in a CICS region. They are enabled either by adding an entry to the CICS PLT, or by using the supplied CFA transaction once CICS has initialized.

The LE abnormal termination exit, however, requires a modification to LE to invoke Fault Analyzer, and hence its effect is system wide. Fault Analyzer provides a mechanism for controlling the use of this exit at a CICS region level, but in order for this mechanism to work, the LE exit must first be enabled system wide. Once enabled at a system-wide level, then the initial setting in a CICS region will be enabled.

IDI.SIDIMOD1 needs to be added to the DFHRPL concatenation of the CICS JCL for any of the exits to be successfully enabled.

To use Fault Analyzer with CICS, you need to perform the following steps:

1. Configure Language Environment for CICS to invoke Fault Analyzer.
2. Define the required programs to your CICS system.
3. Add the required programs to your startup PLT.
4. Define a transaction for Fault Analyzer.

21.1.1 Configuring Language Environment for CICS to invoke Fault Analyzer

Fault Analyzer provides a Language Environment abnormal termination exit for CICS, IDIXCCEE, as an additional method of invoking Fault Analyzer to the CICS XPCABND global exit. This exit is specific to LE U1xxx or U4xxx-type abends and for more information, see the CICS LE abnormal termination CEECXTAN CSECT exit, IDIXCCEE. To enable this exit, you must add it to the CEECXTAN CSECT for Language Environment for CICS. To do this, make a copy of the CEEWCEXT softcopy sample member in the CEE.SCEESAMP data set. Make the changes suggested in the sample member and replace the lines:

```
<<< REPLACE THESE 2 LINES WITH A COPY OF CEECXTAN AND OVERRIDE AS DESIRED >>>
```

with:

```
CEEXAHD ,User exit header
CEEXART TERMXIT=IDIXCCEE
CEEXAST ,Terminate the list
```

21.1.2 Defining required program to CICS

The following programs and BMS map must be defined to your CICS system, unless CICS program auto install is active:

- IDIPLT
- IDIPLTD
- IDIPLTS
- IDIXCX52
- IDIXCX53
- IDIXFA
- IDIXMAP (BMS map)

These programs are all assembler programs, and should be defined in a group that is included in a group list used during CICS startup.

The sample job shown in Figure 21-1 includes definitions for all of these programs.

```
//IDICICS JOB  ...
//IDICICS EXEC PGM=DFHCSDUP,REGION=1024K,
//             PARM='CSD(READWRITE),PAGESIZE(60),NOCOMPAT'
//STEPLIB  DD DISP=SHR,DSN=xxx.SDFHLOAD
//DFHCSD   DD DISP=SHR,DSN=xxx.DFHCSD
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
  DEF PROGRAM(IDIPLT) GROUP(FA) EXECKEY(CICS)
      Language(ASSEMBLER) CEDF(NO) DATALOCATION(ANY)
  DEF PROGRAM(IDIPLTD) GROUP(FA) EXECKEY(CICS)
      Language(ASSEMBLER) CEDF(NO) DATALOCATION(ANY)
  DEF PROGRAM(IDIPLTS) GROUP(FA) EXECKEY(CICS)
    LANGUAGE(ASSEMBLER) CEDF(NO) DATALOCATION(ANY)
  DEF PROGRAM(IDIXCX52) GROUP(FA)
      Language(ASSEMBLER) CEDF(NO) DATALOCATION(ANY)
  DEF PROGRAM(IDIXCX53) GROUP(FA)
      Language(ASSEMBLER) CEDF(NO) DATALOCATION(ANY)
  DEF PROGRAM(IDIXCCEE) GROUP(FA)
      Language(ASSEMBLER) CEDF(NO) DATALOCATION(ANY)
  DEF PROGRAM(IDIXFA) GROUP(FA) EXECKEY(CICS)
      Language(ASSEMBLER) CEDF(NO) DATALOCATION(ANY)
  DEF TRANSACTION(CFA) GROUP(FA)
      PROGRAM(IDIXFA) TASKDATALOC(ANY)
  DEF MAPSET(IDIXMAP) GROUP(FA)
  ADD G(FA) L(list-name)
/*
```

Figure 21-1 Fault Analyzer CICS program and transaction definition job

This sample job is provided as member IDISCICS in data set IDI.SIDISAM1.

Keep the following considerations in mind:

- In order for Fault Analyzer to be invoked under CICS, IDI.SIDIMOD1 must be added to the DFHRPL concatenation.
- CICS tracing must be active for Fault Analyzer to display CICS trace information.
- If CICS is used without LE in the LINKLIST, IDILEDS USERMOD must be installed.
- Fault Analyzer is not invoked for CICS transaction abends if the NODUMP keyword is used on the EXEC CICS ABEND statement that is causing the abend, as shown in Example 21-1.

Example 21-1 EXEC CICS ABEND statement

```
EXEC CICS ABEND ABCODE(<abcd>) NODUMP END_EXEC
```

Note: ABCODE must be used for Fault Analyzer to be invoked.

21.1.3 Controlling CICS transaction abend analysis

Once the CFA transaction has been installed (you might have chosen to install it under a different name, as described previously), it can be used to install or uninstall the following Fault Analyzer invocation exits:

- XPCABND CICS global user exit
- XDUREQ CICS global user exit
- LE abnormal termination exit

In addition, the CFA transaction can be used to install or uninstall the Fault Analyzer SDUMP screening feature.

Prior to installing either the XPCABND or XDUREQ exits, the CFA transaction issues a CICS NEWCOPY command for program IDIXCX52 and IDIXCX53 if both exits are in the <Uninstalled> state. Hence, to load a new copy of IDIXCX52 or IDIXCX53, for example after applying maintenance, use the CFA transaction to uninstall the XPCABND and XDUREQ exits, and then re-install either or both exits as required.

There are two ways to interact with the CFA transaction: from a CICS terminal, or from the MVS console. These methods are described in the following sections.

21.1.4 Using a CFA from a CICS terminal

To use the CFA transaction from a CICS terminal, simply enter CFA. You are subsequently presented with a display similar to Figure 21-2.

```
                    Fault Analyzer Control Transaction

Option: F=Force purge active task

_ Number of ACTIVE analysis tasks   0001 Task(0000049) Elapsed Seconds(002)
  Number of WAITING analysis tasks  0000

Options: I=Install U=Uninstall F=Force purge active task

                  Current Status/Error Message
_ XPCABND         Installed
_ XDUREQ          Installed
_ LE Exit         Installed
_ SDUMP Screening Uninstalled









PF1=Help  PF3=Exit   PF9=IVP   ENTER=Update
```

Figure 21-2 CFA transaction display

Initially, the display shows the current status of the CICS Fault Analyzer exit points, plus details of any active and waiting Fault Analyzer tasks. By entering an I or U (for Install or Uninstall) next to a specific exit point, its status can be changed accordingly. If there is an active analysis task (as show in the example), then a CICS TASK FORCEPURGE can be issued for that task by entering an F in the input field next to the active task details.

This function is only possible if CICS transaction isolation is INACTIVE, or if ACTIVE, that the IDIXFA program is defined to have an EXECKEY of CICS. For help information about a specific CICS exit, place your cursor on the exit selection field on the main panel and press PF1.

21.1.5 Using CFA from an MVS console

To use the CFA transaction from an MVS console, issue the MODIFY (F) command with the CFA command parameter. The CFA command syntax is shown Figure 21-3.

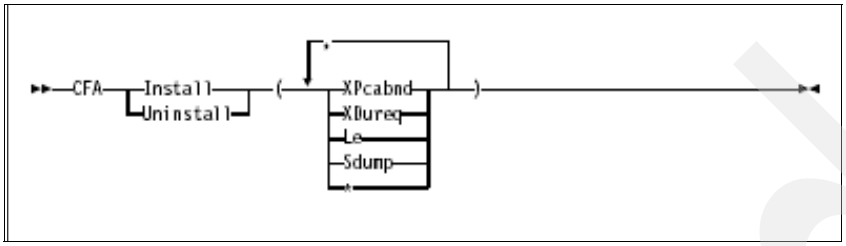


Figure 21-3 CFA command syntax

To uninstall all exits from CICS region CICS01, enter the command:

```
F CICS01,CFA U(*)
```

To install the LE exit for use by CICS region TESTCICS, enter the command:

```
F TESTCICS,CFA I(L)
```

To install the XPCABND and XDUREQ exits for use by CICS region PRODCICS, enter the command:

```
F PRODCICS,CFA I(XP,XD)
```

21.1.6 Ensuring transaction abend analysis is not suppressed by DUMP(NO)

If the active transaction definition for an abending transaction has the DUMP(NO) option specified, then CICS will not call the XPCABND global user exit, and Fault Analyzer will not be invoked. To check the DUMP setting for a transaction, do one of the following:

- Use the CEDA transaction to view the transaction definition in question and check the DUMP(YES|NO) setting. Care should be taken when using this method as there might be multiple definitions of the same transaction, and the order in which the definitions are installed by CICS is important.
- Check the active transaction definition in a dump.
- Use the CICS-supplied transaction, CECI, to check the DUMP setting for the active transaction. Where *nnnn* is the transaction ID in question, this can be done by issuing the following command:

  ```
  CECI INQUIRE TRANSACTION(nnnn)
  ```

  Having issued this command, the displayed DUMPING value has the following meaning:

  – A value of 00186 means DUMP(YES).

  – A value of 00187 means DUMP(NO).

21.1.7 Preventing LE from causing the CICS trace to wrap

When a CICS transaction abends and Language Environment is active in the abending enclave, Language Environment by default writes diagnostic information to a transient data queue named CESE. This occurs if the IBM-supplied Language Environment default run-time option TERMTHDACT(TRACE) is in effect. Because these diagnostics are recorded before Fault Analyzer receives control to process the abend, the CICS trace table is liable to wrap

around, and application trace data might be lost. Depending on your level of MVS, it is recommended that the TERMTHDACT option is set to one of the following:

- TERMTDHACT(TRACE,CICSDDS,...)

  This will cause Language Environment to write its diagnostics to the CICS transaction dump data set.

  This option might only be available if the required maintenance has been applied to your CICS system.

- TERMTDHACT(QUIET)

  This will suppress most of the Language Environment diagnostics.

21.1.8 Specifying data sets through the IDIOPTS DDname

To avoid the need to recycle CICS if compiler listing or side file data sets change, specify these via the data sets option in a user options file pointed to by the IDIOPTS DDname.

The data set used must be a PDS(E) in order to permit update using ISPF EDIT while CICS is running.

User options file IDIOPTS

Fault Analyzer supports the specification of a user options file through the IDIOPTS DDname as shown in Figure 21-4.

```
//MYJOB1   JOB  ...
//STEP1    EXEC PGM=MYAPPL
//SYSMDUMP DD   DISP=SHR,DSN=MY.DUMP.DATA.SET
//IDIOPTS  DD   *
  RetainDump(ALL)  /* do not suppress the dump if MYAPPL
                      abends */
/*
```

Figure 21-4 File to override any dump suppression if program MYAPPL abends

The user options file must be fixed 80-byte record length format.

JCL EXEC statement PARM field

You can specify options in the JCL EXEC statement PARM field when executing Fault Analyzer in batch reanalysis mode as shown in Figure 21-5.

```
//MYJOB2   JOB
//STEP1    EXEC PGM=IDIDA,
// PARM='/Detail(LONG),PreferredFormattingWidth(132)'
:
```

Figure 21-5 JCL EXEC statement PARM field

Option descriptions

Descriptions of all options are in the IBM Fault Analyzer for z/OS User's and Reference Guide SC18-9374-00, chapter 25.

21.1.9 Language Environment abend considerations

If an abend occurs in Language Environment while trying to recover from a transaction abend under CICS, then Fault Analyzer will not be invoked. This is because CICS normal behavior for these types of abends is to not drive the XPCABND exit.

However, by enabling the CICS LE abnormal termination CEECXTAN CSECT exit IDIXCCEE, which is provided with Fault Analyzer, these types of abends will still be analyzed.

21.1.10 Installing the MVS post-dump exit IDIXTSEL

This optional Fault Analyzer post-dump exit, IDIXTSEL, is installed in the IEAVTSEL installation exit list. The exit, which is only invoked for SVC dumps, is installed by the USERMOD, IDIWTSEL. It is normally only required to register CICS system abend dumps.

To install this USERMOD, edit and submit the sample job IDIWTSEL. This will include IDIXTSEL in the IEAVTSEL installation exit list. If you have other exits defined in this list, add the IDIXTSEL exit last.

To activate this change, re-IPL or cancel the DUMPSRV address space so that it restarts with the new exit.

For dump registration via this exit to occur, it is necessary to also start the Fault Analyzer subsystem

Starting the subsystem

To start the subsystem, the following simple job can be submitted:

```
//IDISS JOB ... //IDISSTST EXEC PGM=IDISAMAN,TIME=NOLIMIT
//* (Optional DD statements might follow, as described below)
```

Alternatively, the subsystem can be established using a started task. The IDIS subsystem dynamically allocates datasets to SYSOUT=*, so it must be run under the job entry subsystem (JES).

Ensure that the TIME=NOLIMIT parameter is specified as shown in the example to prevent subsystem abend S522.

There is no requirement for any REGION size specification; default 32 MB region is adequate. If you have more than one version of DB2 installed, or if the DB2 load module library is not in LINKLIST, then you must add DD statements for all DB2 subsystems that are not accessible via LINKLIST, and for which you want Fault Analyzer to perform analysis, as follows:

```
//DB2subsystem-id DD DISP=SHR,DSN=data-set-name
```

where *subsystem-id* is the DB2 subsystem ID (usually 4 characters), and *data-set-name* is the associated load module library.

For a data sharing group, the group attach name is used as the subsystem ID.

If, for example, the DB2 subsystem with an ID of DSN1 requires the load library DSN1.LOADLIB, which is not in LINKLIST, then add the following JCL DD statement to the Fault Analyzer subsystem job:

```
//DB2DSN1 DD DISP=SHR,DSN=DSN1.LOADLIB
```

The subsystem needs access to the DB2 PLAN:DSNACLI, and READ access to the following SYSIBM catalog tables:

- SYSIBM.SYSDBRM
- SYSIBM.SYSPACKAGE
- SYSIBM.SYSPACKSTMT
- SYSIBM.SYSPLAN
- SYSIBM.SYSSTMT

The subsystem name used by Fault Analyzer is IDIS.

This name does not need to be defined in the IEFSSNxx parmlib member because it is dynamically defined by the IDISAMAN program.

Stopping the subsystem

The subsystem may be cancelled and restarted at any time.

A MODIFY command of the following format may also be used to stop the subsystem:

```
F <name>,STOP
```

Note: <name> is the appropriate identifier for the MODIFY command, depending on the way in which the subsystem was started. If either the FA subsystem is not active, or if an incorrect identifier was used on the MODIFY command, MVS issues the message: `IEE341I XYZ NOT ACTIVE.`

If the FA subsystem is already active when another attempt to start it is performed, then the following message will be issued to the operator console: `IDISAMAN The Fault Analyzer Subsystem is already active in <jobname job-id>`

<jobname> is the job or started task name of the currently executing FA subsystem. <job-id> is the JES job or started task ID.

21.1.11 Storage requirements

The real-time execution following an abend requires extra storage in the abending region while the analysis is carried out on the in-storage data.

The following are the requirements for the *minimum* available region size, assuming that neither Language Environment, nor Fault Analyzer, are available from LPA:

- A minimum of 243 kilobytes below-the-line storage regardless of execution environment.
- A minimum of 16 megabytes above-the-line storage for CICS transactions.
- A minimum of 14 megabytes above-the-line storage for programs other than CICS transactions.

Depending on the type of fault being analyzed, and the environment in which this occurs, additional storage might be required.

If the initial amount of required storage is not available, then message IDI0055I will be issued and no analysis will be performed.

The storage requirements under CICS are for MVS GETMAIN-managed storage, not CICS DSA-managed storage. So, to increase below-the-line MVS GETMAIN-managed storage, you would need to decrease CICS below-the-line DSA-managed storage (and similarly for above-the-line storage).

Information about the actual storage used by Fault Analyzer is available at the end of the real-time analysis report. However, the amount of storage provided in the report accounts for the explicit allocations performed by Fault Analyzer only and does not include, for example, Language Environment heap and stack storage or storage used for load modules.

In post-abend situations, where the minidump or SYSMDUMP is being processed, only a marginal increase in storage requirements occurs over that of the real-time execution, as the result of allocating space for referenced dump pages. The increase is typically less than 500 kilobytes.

For interactive reanalysis, the storage is required in the TSO region.

The minimum available region size above-the-line can be reduced with the size of required modules that are either available from LPA, and therefore do not need to be loaded, or already loaded, if, for example, the abending program uses LE.

Having LE in LPA saves almost 5 megabytes, and Fault Analyzer in LPA almost 7 megabytes, reducing the storage requirement for a typical non-CICS program to less than 3 megabytes.

If the necessary below-the-line size is not available, then message IDI0086E will be issued and processing terminates.

If the necessary above-the-line size is not available, then message IDI0055E will be issued and processing terminates. Additionally, message IDI0087I might be issued to provide information about storage that could be made available if the command included in the message text is issued to add modules to LPA. The module names likely to be included in the message are the Fault Analyzer modules IDIDA, IDIBOOKR, and IDILANGX. The are two ways to place these modules in LPA, and save approximately 7 megabytes above-the-line storage. Use either of the following methods:

1. Issue the following MVS operator command:

   ```
   SETPROG LPA,ADD,MOD=(IDIDA,IDIBOOKR,IDILANGX),DSN=LNKLST
   ```

2. Add the following line to the LPALSTxx parmlib member:

   ```
   IDI.SIDILPA1 to the
   ```

> **Note:** If Fault Analyzer modules are loaded into LPA, then it is important that the FA modules be placed again in the LPA after applying Fault Analyzer maintenance. Failure to perform this step following the installation of maintenance will prevent the update of Fault Analyzer LPA modules. Because all Fault Analyzer modules are not in LPA, this can cause a mismatch between the old and the new code, which might lead to undefined behavior.

The MVS IEFUSI exit can be used as a general way to provide additional region size if JCL change is not practical for all jobs.

A sample IEFUSI exit is provided as member IDISUSI in the IDI.SIDISAM1 data set. The exit increases the region size of all jobs by 16 megabytes.

Figure 21-6 through Figure 21-28 show the IDISUSI job jcl sample.

```
  Menu  Utilities  Compilers  Help
 ISRBROBA  ADTOOLS.FAULT.ANAL51.SIDISAM1(IDISUSI)     Line 00000000 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
******************************** Top of Data ********************************
//IDISUSI JOB  <JOB CARD PARAMETERS>
//*
//*******************************************************************
//* Licensed Materials - Property of IBM                            *
//*                                                                 *
//* 5655-M20                                                        *
//*                                                                 *
//* (c) Copyright IBM Corp. 2003. All rights reserved.              *
//*                                                                 *
//* US Government Users Restricted Rights - Use,                    *
//* duplication or disclosure restricted by GSA ADP                 *
//* Schedule Contract with IBM Corp.                                *
//*                                                                 *
//*******************************************************************
//*******************************************************************
//*                                                                 *
//* IDISUSI JOB                                                     *
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 21-6 IDISUSI job jcl sample (Part 1 of 23)

```
  Menu  Utilities  Compilers  Help
 ISRBROBA  ADTOOLS.FAULT.ANAL51.SIDISAM1(IDISUSI)     Line 00000018 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
//*                                                                 *
//* This JCL will install a sample SMF Step Initialization user exit *
//* written in Assembler.                                           *
//*                                                                 *
//* Caution: This is neither a JCL procedure, nor a complete job.   *
//* Before using this job, you will have to make the following      *
//* modifications:                                                  *
//*                                                                 *
//* 1. Review the logic of the assembler and make any modifications *
//*    necessary to meet your system requirements.                  *
//* 2. Change the job card to meet your installation standards      *
//* 3. Change the data set name of SYSLMOD to meet your             *
//*    installation requirements.                                   *
//*                                                                 *
//*******************************************************************
//ASSEMBLE EXEC PGM=ASMA90,PARM='LINECOUNT(0),RENT'
//SYSPRINT DD SYSOUT=*
//SYSUT1   DD DISP=(,PASS),UNIT=SYSALLDA,SPACE=(TRK,(1,5))
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 21-7 IDISUSI job jcl sample (Part 2 of 23)

```
  Menu  Utilities  Compilers  Help
 ISRBROBA  ADTOOLS.FAULT.ANAL51.SIDISAM1(IDISUSI)     Line 00000036 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
//SYSLIN   DD DISP=(,PASS),UNIT=SYSALLDA,SPACE=(TRK,(1,5))
//SYSLIB   DD DISP=SHR,DSN=SYS1.MACLIB
//         DD DISP=SHR,DSN=SYS1.MODGEN
//SYSIN    DD DATA,DLM='##'
IDISUSI  TITLE 'SMF STEP INITIATION EXIT ROUTINE                '       0014000
*/* LIB:                                                            */ 0015000
*/* GDE:                                                            */ 0016000
*/* DOC: THIS IS SAMPLE SMF EXIT ROUTINE TO SET REGION SIZE         */ 0017000
*/*      AND REGION LIMIT (BELOW AND ABOVE 16MB)                    */ 0018000
*/****START OF SPECIFICATIONS*****************************************/ 0019000
*/*                                                                 */ 0020000
*/* $MAC(IDISUSI) COMP(SC100) PROD(JBB6607): SMF STEP INITIATION EXIT*/ 0021000
*/*                                          ROUTINE                */ 0022000
*/*                                                                 */ 0023000
*/*01* MODULE NAME = IDISUSI                                        */ 0024000
*/*                                                                 */ 0025000
*/*01* DESCRIPTIVE NAME =  SMF EXIT ROUTINE TO UPDATE THE GETMAIN   */ 0026000
*/*                        LIMITS                                   */ 0027000
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 21-8 IDISUSI job jcl sample (Part 3 of 23)

```
  Menu  Utilities  Compilers  Help
 ISRBROBA  ADTOOLS.FAULT.ANAL51.SIDISAM1(IDISUSI)     Line 00000054 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
*/*                                                                 */ 0028000
*/*01* COPYRIGHT = LICENSED MATERIALS - PROPERTY OF IBM             */ 0029000
*/*                THIS MODULE IS "RESTRICTED MATERIALS OF IBM"     */ 0030000
*/*                5647-A01 (C) COPYRIGHT IBM CORP. 1999            */ 0031000
*/*                                                                 */ 0032000
*/*01* STATUS = JBB6607                                             */ 0033000
*/*                                                                 */ 0034000
*/*01* FUNCTION =  BASED ON THE REQUESTED REGION SIZE BELOW 16MB,   */ 0035000
*/*                CALCULATES THE REGION LIMIT BELOW 16MB AS BEING  */ 0036000
*/*                THE REGION SIZE PLUS N TIMES 64K.                */ 0037000
*/*                IT ALSO SETS A REGION SIZE AND LIMIT ABOVE 16MB. */ 0038000
*/*                                                                 */ 0039000
*/*02*   OPERATION = INVOKED AT STEP INITIATION AND ACTIVATED FOR   */ 0040000
*/*                 ALL JOBS.                                       */ 0041000
*/*                                                                 */ 0042000
*/*01* NOTES =                                                      */ 0043000
*/*                                                                 */ 0044000
*/*02*   DEPENDENCIES =                                            */ 0045000
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 21-9 IDISUSI job jcl sample (Part 4 of 23)

```
  Menu  Utilities  Compilers  Help
 ISRBROBA  ADTOOLS.FAULT.ANAL51.SIDISAM1(IDISUSI)     Line 00000072 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
*/*                                                                  */ 0046000
*/*02*   CHARACTER DEPENDENCIES = CHARACTER SET IS EBCDIC.           */ 0047000
*/*                               REASSEMBLE IF A DIFFERENT          */ 0048000
*/*                               CHARACTER SET IS NEEDED.           */ 0049000
*/*                                                                  */ 0050000
*/*02*   RESTRICTIONS = NONE                                        */ 0051000
*/*                                                                  */ 0052000
*/*02*   REGISTER CONVENTIONS = STANDARD CONVENTIONS.                */ 0053000
*/*                                                                  */ 0054000
*/*03*     REGISTERS SAVED = 2-14                                   */ 0055000
*/*                                                                  */ 0056000
*/*03*     REGISTER USAGE =                                         */ 0057000
*/*                                                                  */ 0058000
*/*          REGISTER  0     = UNUSED                                */ 0059000
*/*          REGISTER  1     = ADDRESS OF INPUT PARAMETERS           */ 0060000
*/*          REGISTER  1     = ADDRESS OF COMMON EXIT                */ 0060100
*/*                            PARAMETER AREA                        */ 0060200
*/*          REGISTER  2     = ADDRESS OF JOB STEP NAME              */ 0061000
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 21-10 IDISUSI job jcl sample (Part 5 of 23)

```
  Menu  Utilities  Compilers  Help
 ISRBROBA  ADTOOLS.FAULT.ANAL51.SIDISAM1(IDISUSI)     Line 00000090 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
*/*          REGISTER  3     = ADDRESS OF PROGRAM NAME               */ 0062000
*/*          REGISTER  4     = ADDRESS OF ACCOUNTING INFO            */ 0064000
*/*          REGISTER  5     = ADDRESS OF VSM PARM LIST              */ 0065000
*/*          REGISTER  6     = ADDRESS OF V=R FLAG                   */ 0066000
*/*          REGISTER  7     = ADDRESS OF DATASPACE VALUES           */ 0067000
*/*          REGISTER  8     = ADDRESS OF SRM INFORMATION            */ 0068000
*/*          REGISTERS 9     = ADDRESS OF SUBSYS NAME                */ 0069000
*/*          REGISTERS 10    = WORK REGISTER                         */ 0070000
*/*          REGISTERS 11    = WORK REGISTER                         */ 0071000
*/*          REGISTER  12    = ADDRESSABILITY TO IDISUSI             */ 0072000
*/*                            CSECT                                 */ 0073000
*/*          REGISTER  13    = CALLERS SAVE AREA ADDRESS             */ 0074000
*/*          REGISTER  14    = WORK REGISTER                         */ 0075000
*/*          REGISTER  15    = WORK REGISTER                         */ 0076000
*/*                                                                  */ 0077000
*/*03*     REGISTERS RESTORED = 2-14                                */ 0078000
*/*                                                                  */ 0079000
*/*02*   PATCH LABEL = PATCH (UNUSED AND INTIALIZED TO               */ 0080000
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 21-11 IDISUSI job jcl sample (Part 6 of 23)

```
  Menu  Utilities  Compilers  Help
 ISRBROBA  ADTOOLS.FAULT.ANAL51.SIDISAM1(IDISUSI)     Line 00000108 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
*/*                    BINARY ZEROES)                                  */ 0081000
*/*                                                                    */ 0082000
*/*01* MODULE TYPE = PROCEDURE                                         */ 0083000
*/*                                                                    */ 0084000
*/*02*   PROCESSOR = ASSEMBLER H VERSION 2                            */ 0085000
*/*                                                                    */ 0086000
*/*02*   MODULE SIZE =  SEE ASSEMBLER LISTING                         */ 0087000
*/*                                                                    */ 0088000
*/*02*   ATTRIBUTES =                                                 */ 0089000
*/*                                                                    */ 0090000
*/*        LOCATION = LPA                                              */ 0091000
*/*        STATE = SUPERVISOR                                          */ 0092000
*/*        AMODE = 31                                                  */ 0093000
*/*        RMODE = ANY                                                 */ 0094000
*/*        KEY = KEY 0                                                 */ 0095000
*/*        MODE = ENABLED                                              */ 0096000
*/*        SERIALIZATION = NONE                                        */ 0097000
*/*        TYPE = REENTRANT                                            */ 0098000
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 21-12 IDISUSI job jcl sample (Part 7 of 23)

```
  Menu  Utilities  Compilers  Help
 ISRBROBA  ADTOOLS.FAULT.ANAL51.SIDISAM1(IDISUSI)     Line 00000126 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
*/*                                                                    */ 0099000
*/* ================================================================ */ 0100000
*/*                                                                    */ 0101000
*/*01* ENTRY POINT = IDISUSI                                           */ 0102000
*/*                                                                    */ 0103000
*/*02*   PURPOSE = INCREASE ABOVE LINE REGION TO ACCOMODATE FAULT     */ 0104000
*/*               ANALYZER                                             */ 0105000
*/*                                                                    */ 0105100
*/*02*   LINKAGE =                                                    */ 0106000
*/*                                                                    */ 0107000
*/*03*     CALLERS =                                                  */ 0108000
*/*                                                                    */ 0109000
*/*02*   INPUT = REG1 POINTS TO A LIST OF FOUR-BYTE ADRESSES          */ 0110000
*/*             AS FOLLOWS:                                            */ 0111000
*/*             1. ADDRESS OF COMMON EXIT PARAMETER AREA               */ 0112000
*/*             2. ADDRESS OF AN 8-BYTE AREA CONTAINING                */ 0113000
*/*                THE JOB STEP NAME FROM THE EXEC STATEMENT.          */ 0114000
*/*             3. ADDRESS OF AN 8-BYTE AREA CONTAINING                */ 0115000
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 21-13 IDISUSI job jcl sample (Part 8 of 23)

```
  Menu  Utilities  Compilers  Help
 ISRBROBA  ADTOOLS.FAULT.ANAL51.SIDISAM1(IDISUSI)     Line 00000144 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
*/*                 THE PROGRAM NAME FROM THE EXEC STATEMENT.          */ 0116000
*/*              4. ADDRESS OF AN AREA CONTAINING THE ACCOUNTING       */ 0117000
*/*                 INFORMATION FROM THE EXEC STATEMENT.               */ 0118000
*/*              5. ADDRESS OF A SIX-WORD AREA CONTAINING REGION       */ 0119000
*/*                 LIMIT VALUES.                                      */ 0120000
*/*              6. ADDRESS OF A WORD CONTAINING V=R FLAG              */ 0121000
*/*              7. ADDRESS OF A FOUR-WORD AREA CONTAINING DEFAULT     */ 0122000
*/*                 VALUES FOR DATA SPACES, HIPERSPACES AND DATA       */ 0122100
*/*                 SHARING (THROUGH THE IARVSERV MACRO)               */ 0122200
*/*              8. ADDRESS OF A FOUR-CHAR AREA THAT CONTAINS THE      */ 0122210
*/*                 NAME OF THE SYBSYSTEM FOR THE JOB.                 */ 0122300
*/*              9. ADDRESS OF THREE 64-BIT FIELDS CONTAINING          */ 0122400
*/*                 MEMLIMIT FLAGS AND VALUES.                         */ 0122500
*/*                                                                    */ 0122600
*/*02*   ENTRY REGISTERS =                                             */ 0123000
*/*                                                                    */ 0124000
*/*02*   OUTPUT = NONE                                                 */ 0125000
*/*                                                                    */ 0126000
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 21-14 IDISUSI job jcl sample (Part 9 of 23)

```
  Menu  Utilities  Compilers  Help
 ISRBROBA  ADTOOLS.FAULT.ANAL51.SIDISAM1(IDISUSI)     Line 00000162 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
*/*02*   EXIT NORMAL = AT PROGRAM END VIA BSM 0,14 TO                  */ 0127000
*/*                    RETURN IN CALLER'S MODE                         */ 0128000
*/*                                                                    */ 0129000
*/*03*   CONDITIONS =                                                  */ 0130000
*/*                                                                    */ 0131000
*/*02*   EXIT REGISTERS =                                              */ 0132000
*/*                                                                    */ 0133000
*/*03*   RETURN CODE = ZERO                                            */ 0134000
*/*                                                                    */ 0135000
*/*01*   EXIT ERROR = NONE                                             */ 0136000
*/*                                                                    */ 0137000
*/*03*   CONDITIONS =                                                  */ 0138000
*/*                                                                    */ 0139000
*/*02*   EXIT REGISTERS =                                              */ 0140000
*/*                                                                    */ 0141000
*/*03*   RETURN CODE = ZERO                                            */ 0142000
*/*                                                                    */ 0143000
*/* ================================================================ */ 0144000
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 21-15 IDISUSI job jcl sample (Part 10 of 23)

```
  Menu  Utilities  Compilers  Help
 ISRBROBA  ADTOOLS.FAULT.ANAL51.SIDISAM1(IDISUSI)     Line 00000180 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
*/*                                                                  */ 0145000
*/*01* EXTERNAL REFERENCES =                                         */ 0146000
*/*                                                                  */ 0147000
*/*02*   ROUTINES = NONE                                             */ 0148000
*/*                                                                  */ 0149000
*/*02*   DATA AREAS = NONE                                           */ 0150000
*/*                                                                  */ 0151000
*/*02*   CONTROL BLOCKS = NONE                                       */ 0152000
*/*                                                                  */ 0153000
*/*01* TABLES = NONE                                                 */ 0154000
*/*                                                                  */ 0155000
*/*01* MAPPING MACROS = CVT, IHAGDA                                  */ 0156000
*/*                                                                  */ 0157000
*/*01* EXECUTABLE MACROS = SAVE, RETURN                              */ 0158000
*/*                                                                  */ 0159000
*/*01* SERIALIZATION = NONE                                          */ 0160000
*/*                                                                  */ 0161000
*/*01* MESSAGES = NONE                                               */ 0162000
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 21-16 IDISUSI job jcl sample (Part 11 of 23)

```
  Menu  Utilities  Compilers  Help
 ISRBROBA  ADTOOLS.FAULT.ANAL51.SIDISAM1(IDISUSI)     Line 00000198 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
*/*                                                                  */ 0163000
*/*01* ABEND CODES = NONE                                            */ 0164000
*/*                                                                  */ 0165000
*/*01* WAIT STATE CODES = NONE                                       */ 0166000
*/*                                                                  */ 0167000
*/*01* SYSGEN =                                                      */ 0168000
*/*      LOAD MODULE = IDISUSI                                       */ 0169000
*/*      DISTRIBUTION LIBRARY = ASAMPLIB                             */ 0170000
*/*      SYSGEN MACRO = N/A                                          */ 0171000
*/*      ALIAS NAME = NONE                                           */ 0172000
*/*      ENTRY POINT = IDISUSI                                       */ 0173000
*/*      TARGET LIBRARY = SAMPLIB                                    */ 0174000
*/*                                                                  */ 0175000
*/*01* CHANGE ACTIVITY = THIS IS A NEW SMF EXIT WHICH IS PROVIDED    */ 0176000
*/*                     FOR MVS/XA.  IT WILL RUN IN 31 BIT MODE      */ 0177000
*/*                     ABOVE THE LINE.                              */ 0178000
*/*                                                                  */ 0179000
*/*$CC= REASONCD RELEASE DTCHGD PGMR: DESCRIPTION                    */ 0180000
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 21-17 IDISUSI job jcl sample (Part 12 of 23)

```
  Menu  Utilities  Compilers  Help
 ISRBROBA  ADTOOLS.FAULT.ANAL51.SIDISAM1(IDISUSI)     Line 00000216 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
*/*$L1= SMFEXITS JBB6607 980501 PDSE: CONFORM TO BCP/MVS         @L1A*/ 0181000
*/*                                  STANDARDS                   @L1A*/ 0182000
*/*                                                                  */ 0183000
*/****END OF SPECIFICATIONS*********************************************/ 0184000
         EJECT                                                          0185000
IDISUSI  CSECT ,                       CSECT NAME DECLARED              0186000
IDISUSI  AMODE 31                                                       0187000
IDISUSI  RMODE ANY                                                      0188000
*                                                                       0189000
*                                                                       0190000
*                                                                       0191000
* REGISTER EQUATES                                                      0192000
*                                                                       0193000
R00      EQU   00                      REGISTER 0                       0194000
@FAINC   EQU   00                      FAULT ANALYZER INCREMENT         0194100
R01      EQU   01                      REGISTER 1                       0195000
R02      EQU   02                      REGISTER 2                       0196000
R03      EQU   03                      REGISTER 3                       0197000
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 21-18 IDISUSI job jcl sample (Part 13 of 23)

```
  Menu  Utilities  Compilers  Help
 ISRBROBA  ADTOOLS.FAULT.ANAL51.SIDISAM1(IDISUSI)     Line 00000234 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
R04      EQU   04                      REGISTER 4                       0198000
R05      EQU   05                      REGISTER 5                       0199000
R06      EQU   06                      REGISTER 6                       0200000
R07      EQU   07                      REGISTER 7                       0201000
@REGSZ    EQU   07                     REGION SIZE BELOW                0202000
R08      EQU   08                      REGISTER 8                       0203000
@REGLIM   EQU   08                     REGION LIMIT BELOW               0204000
R09      EQU   09                      REGISTER 9                       0205000
@EREGSZ   EQU   09                     REGION SIZE ABOVE                0206000
R10      EQU   10                      REGISTER 10                      0207000
@EREGLIM  EQU   10                     REGION LIMIT ABOVE               0208000
R11      EQU   11                      REGISTER 11                      0209000
R12      EQU   12                      REGISTER 12                      0210000
R13      EQU   13                      REGISTER 13                      0211000
R14      EQU   14                      REGISTER 14                      0212000
R15      EQU   15                      REGISTER 15                      0213000
         EJECT                                                          0214000
         SAVE  (14,12),,IDISUSI_&SYSDATE._&SYSTIME                      0215000
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 21-19 IDISUSI job jcl sample (Part 14 of 23)

```
  Menu  Utilities  Compilers  Help
 ISRBROBA  ADTOOLS.FAULT.ANAL51.SIDISAM1(IDISUSI)     Line 00000252 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
         USING IDISUSI,R12             SET UP BASE ADDRESSABILITY
*
* NOTE THAT NO SAVE_AREA WAS GETMAINED SINCE THIS ROUTINE
*      CALLS NO OTHER ROUTINE OR ISSUES ANY SVC'S
*
         LR    R12,R15                 LOAD BASE REG WITH ENTRY POINT
         LM    R01,R08,0(R01)          LOAD INPUT PARM LIST ADDRESSES
*
*    R01 = ADDRESS OF COMMON PARAMETER EXIT ARE
*    R02 = ADDRESS OF 8-BYTE AREA CONTAINING JOB STEP NAME
*    R03 = ADDRESS OF 8-BYTE AREA CONTAINING PROGRAM NAME
*    R04 = ADDRESS OF AREA CONTAINING ACCOUNTING INFORMATION
*    R05 = ADDRESS OF SIX-WORD AREA CONTAINING REGION LIMIT VALUES
*    R06 = ADDRESS OF WORD CONTAINING V=R FLAG
*    R07 = ADDRESS OF FOUR-WORD AREA FOR DATASPACE & HIPERSPACE VALUES
*    R08 = ADDRESS OF FOUR-CHAR AREA CONTAINING SUBSYS NAME
*
         CLC   0(4,R08),=C'OMVS'       FORKED OR SPAWNED?
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 21-20 IDISUSI job jcl sample (Part 15 of 23)

```
  Menu  Utilities  Compilers  Help
 ISRBROBA  ADTOOLS.FAULT.ANAL51.SIDISAM1(IDISUSI)     Line 00000270 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
         BE    EXIT                    YES USE DEFAULT VALUES
         TM    0(R06),X'80'            V=R JOB?
         BO    EXIT                    YES USE DEFAULT VALUES
         USING REGION,R05              ADDRESSABILITY FOR REGION DSECT
         L     R11,REGSZREQ            GET REQUESTED REGION SIZE
         LTR   R11,R11                 IS IT ZERO
         BZ    EXIT                    YES USE DEFAULT VALUES
         OI    REGFLAGS,X'80'          SET, SO WE CONTROL REGIONSIZE
         SPACE
*-------------------------------------------------------------------*
* DECIDE ON INCREMENT SIZE TO USE BASED ON JOBNAME AND PGM NAME      *
*-------------------------------------------------------------------*
*        USING JMR,R01                  ADDRESSABILITY FOR JOB MGMT REC
*        CLC   JMRJOB(3),=C'CIC'        CICS?
*        BNE   CHKPGMNM                 NO, CHECK PROGRAM NAME
*        L     @FAINC,F24M              INCREASE REGION BY 24M
*        B     GETGDA
*        DROP  R01
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 21-21 IDISUSI job jcl sample (Part 16 of 23)

```
  Menu  Utilities  Compilers  Help
 ISRBROBA  ADTOOLS.FAULT.ANAL51.SIDISAM1(IDISUSI)     Line 00000288 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
CHKPGMNM DS    0H                                                        0246160
*        CLC   0(8,R03),=CL8'DFHSIP'   CICS?                             0246170
*        BNE   USE16M                  NO, CHECK PROGRAM NAME            0246180
*        L     @FAINC,F24M             INCREASE REGION BY 24M            0246190
*        B     GETGDA                                                    0246200
USE16M   DS    0H                                                        0246490
         L     @FAINC,F16M             INCREASE REGION BY 16M            0246520
         SPACE                                                           0246530
*-----------------------------------------------------------------*      0246540
* GET THE MAXIMUM PRIVATE AREA SIZES FROM THE GDA AND CALCULATE     *     0246550
* MAX REGION VALUES                                                 *     0246560
*-----------------------------------------------------------------*      0246570
GETGDA   DS    0H                                                        0246600
         L     R10,16                  -> CVT                            0247000
         USING CVT,R10                                                   0248000
         L     R10,CVTGDA              -> GDA                            0249000
         USING GDA,R10                                                   0250000
         L     @REGLIM,GDAPVTSZ        GET PVT SIZE BELOW                0251000
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 21-22 IDISUSI job jcl sample (Part 17 of 23)

```
  Menu  Utilities  Compilers  Help
 ISRBROBA  ADTOOLS.FAULT.ANAL51.SIDISAM1(IDISUSI)     Line 00000306 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
         S     @REGLIM,F512K           MINUS PSA, SYS.REG, LSQA, SWA... 0252000
         LR    @REGSZ,@REGLIM                                           0253000
         S     @REGSZ,F64K             64K BETWEEN SIZE AND LIMIT       0254000
         L     @EREGLIM,GDAEPVTS       GET PVT SIZE ABOVE               0255000
         DROP  R10                                                      0256000
*-----------------------------------------------------------------*      0257000
* UPDATE PARAMETERS BASED ON REQUESTED REGION SIZE                  *     0258000
*-----------------------------------------------------------------*      0259000
         C     R11,F16M                                                 0260000
         BH    REQGT16M                                                 0261000
         A     @FAINC,F32M             SET REGION LIMIT ABOVE TO        0262000
         ST    @FAINC,REGLIMA           INCREMENT PLUS 32MEG            0262100
         CR    R11,@REGSZ                                               0263000
         BH    DEFAULT                                                  0264000
         ST    R11,REGSIZB             SET REGION SIZE BELOW            0265000
         A     R11,F64K                64K BETWEEN SIZE AND LIMIT       0266000
         ST    R11,REGLIMB             SET REGION LIMIT BELOW           0267000
         B     SETSIZA                 GO SET REGION SIZE ABOVE         0268000
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 21-23 IDISUSI job jcl sample (Part 18 of 23)

```
  Menu  Utilities  Compilers  Help
 ISRBROBA  ADTOOLS.FAULT.ANAL51.SIDISAM1(IDISUSI)     Line 00000324 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
DEFAULT  DS    0H                                                        0269000
         ST    @REGSZ,REGSIZB          SET REGION SIZE BELOW             0270000
         ST    @REGLIM,REGLIMB         SET REGION LIMIT BELOW            0271000
         B     SETSIZA                 GO SET REGION SIZE ABOVE          0272000
REQGT16M DS    0H                                                        0273000
         ST    @REGSZ,REGSIZB          SET REGION SIZE BELOW             0274000
         ST    @REGLIM,REGLIMB         SET REGION LIMIT BELOW            0275000
         LR    R01,@FAINC                                                0276000
         A     R01,F32M                SET REGION LIMIT ABOVE TO         0276010
         CR    R11,R01                                                   0276100
         BH    REQGTDEF                                                  0277000
         ST    R01,REGLIMA              INCREMENT PLUS 32MEG             0277200
         B     SETSIZA                 GO SET REGION SIZE ABOVE          0279000
REQGTDEF DS    0H                                                        0280000
         AR    R11,@FAINC              INCREASE FOR FAULT ANALYZER       0280100
         CR    R11,@EREGLIM                                              0281000
         BH    REQGTMAX                                                  0282000
         ST    R11,REGLIMA             SET REGION LIMIT ABOVE            0283100
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 21-24 IDISUSI job jcl sample (Part 19 of 23)

```
  Menu  Utilities  Compilers  Help
 ISRBROBA  ADTOOLS.FAULT.ANAL51.SIDISAM1(IDISUSI)     Line 00000342 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
         B     SETSIZA                 GO SET REGION SIZE ABOVE          0284000
REQGTMAX DS    0H                                                        0285000
         ST    @EREGLIM,REGLIMA        SET REGION LIMIT ABOVE            0286000
SETSIZA  DS    0H                                                        0287000
         L     @EREGSZ,REGLIMA         GET REGION LIMIT ABOVE            0288000
         S     @EREGSZ,F512K           512K BETWEEN SIZE AND LIMIT       0289000
         ST    @EREGSZ,REGSIZA         GET REGION SIZE ABOVE             0290000
*                                                                        0291000
* RETURN TO CALLER                                                      0292000
*                                                                        0293000
EXIT     DS    0H                                                        0294000
         LM    14,12,12(13)            RESTORE REGISTERS                 0295000
         LA    15,0                    SET RETURN CODE TO ZERO           0296000
         BSM   0,14                    RETURN TO CALLER IN ITS MODE      0297000
         EJECT                                                           0298000
*                                                                        0299000
*  DATA AREA                                                            0300000
*                                                                        0301000
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 21-25 IDISUSI job jcl sample (Part 20 of 23)

```
  Menu  Utilities  Compilers  Help
 ISRBROBA  ADTOOLS.FAULT.ANAL51.SIDISAM1(IDISUSI)     Line 00000360 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
F32M     DC    A(32*1024*1024)         32M                                0304000
F24M     DC    A(24*1024*1024)         24M                                0304010
F16M     DC    A(16*1024*1024)         16M                                0304100
F512K    DC    A(512*1024)             512K                               0305000
F64K     DC    A(64*1024)              64K                                0306000
ZERO     DC    F'0'                                                       0307000
*                                                                         0308000
*                                                                         0309000
* PATCH AREA                                                              0310000
*                                                                         0311000
         DS    0F                                                         0312000
MODLEN   EQU   *-IDISUSI          EQUATE FOR MODULE LENGTH(- PATCH)       0313000
PTCHLEN  EQU   ((MODLEN+7)/8+7)/8*8 PATCH AREA LENGTH EQUATE              0314000
PTCHLNTH DC    Y(PTCHLEN)          LENGTH OF PATCH AREA                   0315000
PTCHBASE DC    S(PATCH)            BASE DISPLACED ADDRESS OF PATCH        0316000
PATCH    DC    XL(PTCHLEN)'00'     PATCH AREA                             0317000
*                                                                         0318000
*  DSECTS                                                                 0319000
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 21-26 IDISUSI job jcl sample (Part 21 of 23)

```
  Menu  Utilities  Compilers  Help
 ISRBROBA  ADTOOLS.FAULT.ANAL51.SIDISAM1(IDISUSI)     Line 00000378 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
*                                                                         0320000
         CVT DSECT=YES                                                    0321000
         IEFJMR                                                           0322000
         IHAGDA                                                           0322100
REGION   DSECT                                                            0323000
REGFLAGS DS    F                   FLAG WORD                              0324000
REGSZREQ DS    F                   REGION SIZE REQUESTED                  0325000
REGLIMB  DS    F                   REGION LIMIT BELOW 16MB                0326000
REGSIZB  DS    F                   REGION SIZE BELOW 16MB                 0327000
REGLIMA  DS    F                   REGION LIMIT ABOVE 16MB                0328000
REGSIZA  DS    F                   REGION SIZE ABOVE 16MB                 0329000
         END   IDISUSI                                                    0330000
##
//BIND     EXEC  PGM=IEWL,PARM='LIST,RENT',                               0333000
//            COND=(4,LT,ASSEMBLE)                                        0334000
//SYSPRINT DD SYSOUT=*
//SYSLMOD  DD DSN=load.lib(IDISUSI),DISP=SHR    <===== CHANGE
//SYSLIN   DD DSN=*.ASSEMBLE.SYSLIN,DISP=SHR
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 21-27 IDISUSI job jcl sample (Part 22 of 23)

```
  Menu  Utilities  Compilers  Help
 ISRBROBA  ADTOOLS.FAULT.ANAL51.SIDISAM1(IDISUSI)     Line 00000396 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
******************************** Bottom of Data ********************************

 F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
F10=Left   F11=Right  F12=Cancel
```

Figure 21-28 IDISUSI job jcl sample (Part 23 of 23)

21.2 Fault Analyzer and DB2

This section contains information specific to customizing for Fault Analyzer in a DB2 environment.

21.2.1 Binding DB2

If Fault Analyzer is to be run against abends occurring in applications that use DB2, ensure that the DB2 Call Level Interface (CLI) has been installed and the required setup has been performed to bind plan DSNACLI. The DSNACLI plan can be created using the sample job in member DSNTIJCL of the DB2 SDSNSAMP data set.

21.2.2 DB2 and Language Environment

If the CEEEXTAN LE abnormal termination exit (IDIXCEE) has not been installed, then either TERMTHDACT(UATRACE), TERMTHDACT(UADUMP), TERMTHDACT(UAONLY), or TERMTHDACT(UAIMM) must be passed through to LE in order to have Fault Analyzer invoked for the DB2 abend. Example 21-2 is a COBOL/DB2 example that illustrates how LE options can be passed.

Example 21-2 COBOL/DB2 example illustrating how LE options can be passed.

```
//MYJOB JOB
//STEP1 EXEC PGM=IKJEFT01
//DBRMLIB DD DSN=TEST.DB2.DBRMLIB.DATA,DISP=SHR
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//IDIREPRT DD SYSOUT=*
//IDIHIST DD DISP=SHR,DSN=TEST.DB2.HIST
//IDILCOB DD DISP=SHR,DSN=TEST.LISTING.DB2.COBOL
//SYSIN DD *
//SYSTSIN DD *
   DSN SYSTEM(DSN1)
   BIND PLAN(DSNTEST1) QUALIFIER(DSN8510) MEMBER(DACBD001) -
      ACT(REP) ISOLATION(CS)
   RUN PROGRAM(DSNTIAD) PLAN(DSNTIA51) -
      LIB('DSN510.RUNLIB.LOAD')
   RUN PROGRAM(DACBD001) PLAN(DSNTEST1) -
      LIB('CTEST.DB2.LOAD') - PARMS('/TERMTHDACT(UADUMP) ')
   END
//*
```

21.2.3 DB2 stored procedures

Note: Information in this section is based on DB2 version 5.

If the CEEEXTAN LE abnormal termination exit (IDIXCEE) has not been installed, DB2 stored procedures must be stored with RUNOPTS TERMTHDACT(UATRACE), TERMTHDACT(UADUMP), TERMTHDACT(UAONLY), or TERMTHDACT(UAIMM) to invoke Fault Analyzer via the MVS exit IEAVTABX (IDIXDCAP) as shown in Example 21-3.

Example 21-3 DB2 stored procedures with RUNOPTS to invoke Fault Analyzer via exit IEAVTABX(IDIXDCAP)

```
Insert into SYSIBM.SYSPROCEDURES (RUNOPTS ) VALUES ('TERMTHDACT(UADUMP)')
```

When a DB2 stored procedure abnormally terminates, the job executing the calling DB2 program will contain a SQL abend for the DB2 stored procedure in the SYSPRINT output as shown in Figure 21-29.

```
*** CALL TO storprc NOT SUCCESSFUL:
 DSNT408I SQLCODE = -965, ERROR:  STORED PROCEDURE storprc TERMINATED ABNORMALLY
 DSNT418I SQLSTATE   = 51021 SQLSTATE RETURN CODE
 DSNT415I SQLERRP    = DSNX9CAC SQL PROCEDURE DETECTING ERROR
 DSNT416I SQLERRD    = 0  0  0  -1  0  0 SQL DIAGNOSTIC INFORMATION
 DSNT416I SQLERRD    = X'00000000'  X'00000000'  X'00000000'  X'FFFFFFFF'  X'00000000'  X'00000000' SQL DIAGNOSTIC
          INFORMATION
*** ISSUE ROLLBACK WORK BECAUSE STORED PROCEDURE CALL NOT SUCCESSFUL
 DSNT400I SQLCODE = 000,  SUCCESSFUL EXECUTION
*** SEVERE ERROR OCCURRED. PROGRAM IS TERMINATING.
```

Figure 21-29 SQL abend for DB2 stored procedure in the SYSPRINT output

The generated IDIREPRT after a DB2 stored procedure failure is located within the spooled output of the DB2 SPAS address space, as shown in Figure 21-30.

```
 Display  Filter  View  Print  Options  Help
--------------------------------------------------------------------------------
SDSF JOB DATA SET DISPLAY - JOB DB2SPAS  (STC03026)    LINE 1-6 (6)
COMMAND INPUT ===>                                          SCROLL ===> CSR
NP   DDNAME   STEPNAME PROCSTEP DSID OWNER    C DEST              REC-CNT
     JESMSGLG JES2                2 DSN1SPAS S                        2
     JESJCL   JES2                3 DSN1SPAS S                       19
     JESYSMSG JES2                4 DSN1SPAS S                        2
     SYSOUT   DSN1SPAS          101 DSN1SPAS S                        3
     CEEDUMP  DSN1SPAS          102 DSN1SPAS S                      403
     IDIREPRT DSN1SPAS          103 DSN1SPAS S                      280
```

Figure 21-30 Shows output of the DB2 SPAS address space

Fault Analyzer options can be customized via the IDIOPTS DDname within the DB2 SPAS started task, as shown in Example 21-4.

Example 21-4 FA options customized within DB2 SPAS STC

```
//DSN1SPAS PROC RGN=OK,TME=NOLIMIT,DB2SSN=DSN1,NUMTCB=8
//IEFPROC EXEC PGM=DSNX9STP,REGION=&RGN,TIME=&TME,
// PARM=í&DB2SSN,&NUMTCBî
//STEPLIB DD DISP=SHR,DSN=DSN510.RUNLIB.LOAD
// DD DISP=SHR,DSN=DSN510.SDSNLOAD
//IDIOPTS DD DISP=SHR,DSN=DA.DB2SAMP(IDIOPT1)
   where sample IDIOPT1 contains:
   Detail(M)
   MaxMinidumpPages(128)
   DataSets( IDILCOB (
   CTEST.DUMPA.LISTING.DB2.COBOL
   CTEST.DUMPA.LISTING.DB2.COB2LE
   CEST.DUMPA.LISTING.DB2.COB2 )
   IDILCOBO (CTEST.DUMPA.LISTING.DB2.COBOSVS)
   IDILPLI (CTEST.DUMPA.LISTING.DB2.PLI )
   IDIADATA (CTEST.DUMPA.SYSADATA.DB2 )
```

After a stored procedure terminates abnormally, the stored procedure must be restarted via the DB2 subsystem.

21.3 Fault Analyzer and IMS

This section contains information specific to Fault Analyzer in IMS environments.

21.3.1 IMS and Language Environment

If the CEEEXTAN LE abnormal termination exit (IDIXCEE) has not been installed, then either TERMTHDACT(UATRACE), TERMTHDACT(UADUMP), TERMTHDACT(UAONLY), or TERMTHDACT(UAIMM) must be passed through to LE in order to have Fault Analyzer invoked for the IMS abend.

Figure 21-31 is a COBOL/IMS example that illustrates how LE options can be passed by linking a CEEUOPT CSECT into the load module being executed.

```
//IMSLE1 JOB ...
//*
//*        STEP  1: ASSEMBLE CEEUOPT CSECT
//*
//HLASM    EXEC PGM=ASMA90,PARM='LINECOUNT(0)'
//SYSPRINT DD SYSOUT=*
//SYSUT1   DD DISP=(,PASS),UNIT=SYSALLDA,SPACE=(TRK,(1,5))
//SYSLIN   DD DISP=(,PASS),UNIT=SYSALLDA,SPACE=(TRK,(1,5,1)),DSN=&TEMP(CEEUOPT)
//SYSLIB   DD DSN=CEE.SCEEMAC,DISP=SHR
//         DD DSN=SYS1.MACLIB,DISP=SHR
//SYSIN    DD *
         TITLE 'CEEUOPT'
CEEUOPT  CSECT
CEEUOPT  AMODE ANY
CEEUOPT  RMODE ANY
         CEEXOPT TERMTHDACT=(UADUMP)
         END
//*
//*        STEP  2: COMPILE COBOL PROGRAM
//*
//COBCOMP  EXEC IMSCOBOL
//COB.SYSIN    DD DSN=DA.IMSSAMP.COBOL(BATCHJ2),DISP=SHR
//COB.SYSPRINT DD DSN=DA.LISTING.COBOL(BATCHJ2),DISP=SHR
//LKED.FRED    DD DSN=*.HLASM.SYSLIN,DISP=OLD
//LKED.SYSIN   DD *
  Include FRED(CEEUOPT)
  NAME    BATCHJ2(R)
/*
//*
//*        STEP  3: RUN THE PROGRAM
//*
//PROGRUN  EXEC PROC=DLIBATCH,MBR=BATCHJ2,PSB=PSB1,COND=(4,LT),
//         DBRC=Y,MON=Y,FMTO=D,TIME=5
//            UNIT=3390,
//            DCB=BLKSIZE=6144
//SYSPRINT DD SYSOUT=*
//DFSIVD1  DD DISP=SHR,DSN=IMS.DFSIVD1
//DFSIVD1I DD DISP=SHR,DSN=IMS.DFSIVD1I
//DFSCTL   DD DISP=SHR,
//            DSN=IMS.PROCLIB(DFSSBPRM)
//IDIREPRT DD SYSOUT=*
//SYSTSIN  DD *
//*
```

Figure 21-31 COBOL/IMS illustrating LE options being linked through CSECT CEEUOPT

21.4 Performing WebSphere or Java dump analysis

A feature unique to the interactive component of Fault Analyzer is the ability to analyze information related to WebSphere or Java.

This section describes how to:

- Set options for WebSphere or Java analysis
- Select a WebSphere or Java dump data set for analysis

These steps can be performed under Fault Analyzer interactively.

21.4.1 Setting options for WebSphere or Java analysis

The general interactive reanalysis options are also used for WebSphere or Java analysis.

User exits

Since WebSphere or Java analysis is performed as reanalysis against an MVS SVC dump or SYSMDUMP data set, only the Compiler Listing Read and Batch Report Tailoring user exits can be used:

- The Compiler Listing Read user exit (if available) is invoked whenever a line from a compiler listing is required.
- The Batch Report Tailoring user exit (if available) is invoked only if a fault history file entry for the analyzed WebSphere or Java is created, since a batch report is generated and saved with the entry for later viewing.

21.4.2 Type of dumps supported

Only SVC or SYSMDUMP WebSphere or Java dumps are supported by Fault Analyzer.

Limitation: Non-machine-readable dumps, like the LE CEEDUMP, cannot be used for Fault Analyzer analysis.

Using the SLIP command to obtain a dump

In some cases, it might be necessary to use the MVS SLIP (Serviceability Level Indication Processing) operator command to set traps for abends in applications and cause an SVC dump to be taken, which can later be used for analysis by Fault Analyzer.

21.4.3 Selecting a WebSphere or Java dump data set

To select a WebSphere or Java SVC dump or SYSMDUMP data set, first select the Analyze MVS Dump Data Set option from the Fault Entry List display File menu. This brings up the Analyze MVS Dump Data Set display as shown in Figure 21-32.

```
  File  Options  View  Services  Help
 ┌──────────────────────── Analyze MVS Dump Data Set ────────────────────────┐
 │                                                                            │
 │ Enter the name of a MVS SVC or SYSMDUMP data set and press Enter to        │
 │ initiate analysis.                                                         │
 │                                                                            │
 │ Dump Data Set Name    'my.mvsdump.dataset'_                                │
 │                                                                            │
 │  F1=Help    F3=Exit   F12=Cancel                                           │
 └────────────────────────────────────────────────────────────────────────────┘

   Fault_ID System   Job/Tran Abend Date       Time     User_ID  History_File_
     F00010 STLABF6  IMWEBF6  S422  2004/11/02 16:33:22 WEBSRV   IDI.HIST
     F00009 STLABF6  IMWEBF6  U4094 2004/11/02 16:32:51 WEBSRV   IDI.HIST
     F00008 STLABF6  IMWEBF6  U4038 2004/11/02 07:15:14 WEBSRV   IDI.HIST
     F00007 STLABF6  JMONITOR U4093 2004/10/29 07:34:48 n/a      IDI.HIST
     F00006 STLABF6  CONOVERY SB37  2004/10/25 09:47:46 CONOVER  IDI.HIST
     F00005 STLABF6  CONOVERF S0CB  2004/10/21 11:05:18 CONOVER  IDI.HIST
     F00004 STLABF6  TIMOTHY  n/a   2004/10/11 16:21:13 TIMOTHY  IDI.HIST
     F00003 STLABF6  TIMOTHY  n/a   2004/10/11 14:31:39 TIMOTHY  IDI.HIST
     F00002 STLABF6  ZHONG    n/a   2004/10/11 10:53:17 ZHONG    IDI.HIST

*** Bottom of data.
 F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 21-32 Analyze MVS Dump Data Set display

In Figure 21-32, type the name of the WebSphere or Java SVC dump or SYSMDUMP data set to be analyzed. The data set name specification follows the ISPF data set name rules, that is, a data set name that is not enclosed in single quotes is prefixed by the current TSO profile prefix. The data set specified is checked for existence before being accepted. In this example, data set ‘my.mvsdump.dataset’ is being analyzed. When the specified MVS dump data set name has been validated, the Fault Analyzer WebSphere or Java analysis commences as indicated by the following message being displayed:

```
Analyzing MVS dump data set. Please wait...
```

22

Fault Analyzer additional information

This chapter provides a few useful and quick references related to the Fault Analyzer. In particular, it covers:

- Fault Analyzer requirements
- Target system requirements
- Healthy installation and customization checklist

22.1 Requirements at a glance

The following sections define the software, hardware, and additional requirements for the IBM Fault Analyzer for z/OS.

22.1.1 Hardware requirements

- IBM eServer™ zSeries system
- Any hardware that runs IBM z/OS and OS/390, Version 2.10 software or later

22.1.2 Software requirements

- OS/390, Version 2.10 software or later
- z/OS, Version 1.1

22.1.3 Additional requirements

To use Fault Analyzer with CICS applications, one of the following applications is required:

- IBM CICS/ESA®, Version 5.1 or later
- CICS/TS for OS/390 Release 1 or later

22.2 Target system requirements

Figure 22-1 and Figure 22-2 list the target libraries required to be able to use Fault Analyzer.

```
  Menu  Options  View  Utilities  Compilers  Help
 ------------------------------------------------------------------------------
 DSLIST - Data Sets Matching ADTOOLS.FAULT.ANAL51.SIDI*            Row 1 of 13
 Command ===> _____________________________________________ Scroll ===> CSR

 Command - Enter "/" to select action                 Dsorg  Recfm  Lrecl  Blks
 ------------------------------------------------------------------------------
          ADTOOLS.FAULT.ANAL51.SIDIALPA                 PO     U        0  3276
          ADTOOLS.FAULT.ANAL51.SIDIAUTH                 PO     U        0  3276
          ADTOOLS.FAULT.ANAL51.SIDIBOOK                 PO     FB    4096  2457
          ADTOOLS.FAULT.ANAL51.SIDIDOC1                 PO     VB     255  2799
          ADTOOLS.FAULT.ANAL51.SIDIEXEC                 PO     FB      80  2792
          ADTOOLS.FAULT.ANAL51.SIDILPA1                 PO     U        0  3276
          ADTOOLS.FAULT.ANAL51.SIDIMAPS                 PO     VB    1024  2799
          ADTOOLS.FAULT.ANAL51.SIDIMLIB                 PO     FB      80  2792
          ADTOOLS.FAULT.ANAL51.SIDIMOD1                 PO     U        0  3276
          ADTOOLS.FAULT.ANAL51.SIDIPLIB                 PO     FB      80  2792
          ADTOOLS.FAULT.ANAL51.SIDISAM1                 PO     FB      80  2792
          ADTOOLS.FAULT.ANAL51.SIDISLIB                 PO     FB      80  2792
          ADTOOLS.FAULT.ANAL51.SIDITLIB                 PO     FB      80  2792
 ***************************** End of Data Set list *****************************

  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 22-1 Fault Analyzer V5.1 target system libraries (Part 1 of 2)

```
  Menu  Options  View  Utilities  Compilers  Help
 ------------------------------------------------------------------------------
 DSLIST - Data Sets Matching ADTOOLS.FAULT.ANAL51.SIDI*            Row 1 of 13
 Command ===> _____________________________________________ Scroll ===> CSR

 Command - Enter "/" to select action                Tracks %Used XT  Device
 ------------------------------------------------------------------------------
          ADTOOLS.FAULT.ANAL51.SIDIALPA                   5   20   1  3390
          ADTOOLS.FAULT.ANAL51.SIDIAUTH                 503   81   1  3390
          ADTOOLS.FAULT.ANAL51.SIDIBOOK                 831   94   2  3390
          ADTOOLS.FAULT.ANAL51.SIDIDOC1                 315   97   6  3390
          ADTOOLS.FAULT.ANAL51.SIDIEXEC                   5   40   1  3390
          ADTOOLS.FAULT.ANAL51.SIDILPA1                 356   94   5  3390
          ADTOOLS.FAULT.ANAL51.SIDIMAPS                  13   92   1  3390
          ADTOOLS.FAULT.ANAL51.SIDIMLIB                   2   50   1  3390
          ADTOOLS.FAULT.ANAL51.SIDIMOD1                 370   96   2  3390
          ADTOOLS.FAULT.ANAL51.SIDIPLIB                  26   92   1  3390
          ADTOOLS.FAULT.ANAL51.SIDISAM1                  19  100   2  3390
          ADTOOLS.FAULT.ANAL51.SIDISLIB                   2   50   1  3390
          ADTOOLS.FAULT.ANAL51.SIDITLIB                   2   50   1  3390
 ***************************** End of Data Set list *****************************

  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 22-2 Fault Analyzer V5.1 target system libraries (Part 2 of 2)

In these two figures, the high-level qualifiers shown are <ADTOOLS.FAULT.ANAL51> for Fault Analyzer target system libraries. The high-level qualifier will vary from one client to another as part of each client's unique environment.

22.2.1 Library names after you finish installing

Table 22-1 lists the data sets that should exist upon completion of the SMP/E APPLY of Fault Analyzer.

Table 22-1 Data sets

Data Set Name	Contents
IDI.SIDIALPA	Load modules that must be made available from LPA
IDI.SIDIAUTH	Authorized load modules that must be made available from LINKLIST
IDI.SIDIBOOK	Softcopy books
IDI.SIDIDOC1	Softcopy book indices and override files
IDI.SIDIEXEC	REXX execs
IDI.SIDIALPA	Load modules that can be placed in LPA to minimize the space required by Fault Analyzer in the abending region when performing real-time analysis
IDI.SIDIMAPS	Control block maps
IDI.SIDIMLIB	ISPF message members
IDI.SIDIMOD1	Non-authorized load modules that must be made available from LINKLIST
IDI.SIDIPLIB	ISPF panels
IDI.SIDISAM1	Softcopy samples and installation jobs
IDI.SIDISLIB	ISPF skeletons
IDI.SIDITLIB	ISPF tables

22.3 Healthy installation and customization checklist

A series of tasks must be performed to install and customize Fault Analyzer. The following checklist identifies, at a high level, the order in which decisions must be made and tasks completed to satisfy the special requirements for utilizing FA.

1. Fault Analyzer is installed using the SMP/E RECEIVE, APPLY, and ACCEPT commands. The SMP/E dialogs may be used to accomplish the SMP/E installation steps.

> **Tip:** SMP/E CALLLIBS Processing
>
> Fault Analyzer uses the CALLLIBS function provided in SMP/E to resolve external references during installation. When Fault Analyzer is installed, ensure that DDDEFs exist for the following libraries:
>
> – CEE.SCEELKED
> – CEE.SCEESPC
> – ISP.SISPLOAD

2. Make Fault Analyzer modules available via LINKLIST and LPA.
3. Allocate a Fault History file or all Fault History Files requirement.

4. Create a configuration member called IDICNF00 and customize it for the client's requirements for Fault Analyzer's system's default.
5. Determine whether IDICNF00 will be kept in SYS1.PARMLIB or re-directed to another PARMLIB suffix.

> **Tip:** This process is usermod driven under SMP/E and the job jcl IDICNFIG member in the SIDISAM1 suffix data set environment of Fault Analyzer. The process needs to be customized and submitted to make the software change.

6. Install BookRead API parmlib member.
7. Allocate the softcopy book cache data set.
8. Install the MVS change options/suppress dump exit IDIXCAP.
9. Enable LE abnormal termination exit IDIXCEE (optional and if required).
10. Install USERMOD to eliminate the need for jobs to include an MVS dump DD statement (optional).
11. Install USERMOD to identify the name of the LE run-time library (optional.)
12. Install USERMOD to enable implicit Fault Analyzer invocation from PL/I application (optional).
13. Customize the CICS environment (optional).
14. Customize the DB2 environment (optional).
15. Customize the IMS environment (optional).
16. Customize for ISPF.

> **Restriction:** Additional region size required
>
> Fault Analyzer runs in the same region as your abending program at the time of the abend. Therefore, there must be spare GETMAIN storage that is not used by the application in order for Fault Analyzer to run and analyze the program storage in its abend state. Initially, up to 16 megabytes of storage might be required, depending on the execution environment. This additional region size increases as the size and complexity of the abending program increases.

17. Additional customization can be performed on an as-needed basis using user exits.

Part 6

File Manager for z/OS

Part 6 covers File Manager for z/OS (FM), describing the following:

- Using File Manager
- Using File Manager with DB2
- Using File Manager with IMS
- Test data and File Manager

23

Using File Manager

In this chapter we take a detailed look at how IBM File Manager for z/OS (FM) can be used, including presenting examples of utility functions that you can use or modify for your needs. We briefly review the creation and use of templates, and present some helpful information that was discovered during our research.

23.1 What File Manager can be used for

You can use File Manager to display, edit, update, create, copy, compare, print, and erase data. For many of these tasks, you can use templates to enhance File Manager's processing options. In addition to using templates, you can enhance File Manager processing by writing your own custom procedures using DFSORT control statements, or the REXX programming language, or both.

The File Manager keyword `PROC=*` is used to indicate that an in-stream routine is being supplied. If using a DFSORT procedure, the first line must be *FASTPROC. If using both DFSORT and REXX procedures, the DFSORT must come first and the first line of the REXX procedure must be *REXXPROC. No identifying tag is required when only REXX is being used.

This chapter provides examples of what we believe are the most common requirements of a tool such as File Manager.

23.1.1 Performance guidelines

You can easily enhance File Manager with your own procedures, built using either DFSORT or REXX — if there is a choice, always use DFSORT. Performance of many print and copy actions can be enhanced by supplying a procedure containing statements from a subset of the DFSORT statements supported by File Manager, as an alternative to using an equivalent REXX procedure. REXX procedures may be the only option for the more complex select and change types of operations.

DFSORT is used internally wherever possible by File Manager Version 4, and later, for copying and printing data sets and for editing large QSAM data files via an auxiliary data set. It is used when all of the following conditions have been met:

- IBM DFSORT R14 is installed (it does not have to be licensed) at the correct maintenance level (PTF UQ90054, APAR PQ68263 for z/OS).
- The input data set is not partitioned.
- The record format of the input data set is not undefined.

File Manager supports only the following DFSORT statements:

- INCLUDE / OMIT
- INREC / OUTREC
- OUTFIL

If DFSORT is unavailable or unsuitable for a copy or print task, you should consider whether the required function can be achieved using template processing before you resort to a REXX procedure.

When using a REXX procedure, CPU usage and run time may be reduced by observing the following:

- Compiled REXX runs considerably faster than interpreted REXX. By supplying a DD card for the REXX compiler library (if it is installed), you instruct File Manager to compile your REXX before running it.
- Always use RETURN in REXX — never use EXIT. EXIT causes the entire REXX environment to shut down and it then has to be rebuilt to process the next record.

Be prepared to make two passes through the data, rather than one, if this will make processing more efficient.

23.1.2 Conventions used

The examples of batch jobs and reports in this chapter adhere to the following conventions:

- In all of the File Manager batch examples presented in this and other chapters, we include STEPLIB references to the File Manager load library and to the COBOL compiler load library. This is done for accuracy and completeness.
- An explicit reference to the File Manager load library is required only if File Manager is not installed in LINKLIST. An explicit reference to the COBOL compiler load library is required only if the COBOL compiler is not installed in LINKLIST, and when copybooks are processed into templates.
- If File Manager and the COBOL compiler are installed in LINKLIST at your site, your system programmer should modify the ISPF skeleton, FMNFTEXC. You can either comment out or remove the STEPLIB statement.
- In all of the File Manager report output examples presented in this and other chapters, we remove the title page, and all pages not pertinent to the example. This is done for brevity.

23.1.3 Simple copy

The most common usage of File Manager is a data set copy. The Copy utility screen is accessed via option **3.3** from the primary option menu as shown in Figure 23-1.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 File Manager                    Copy Utility
 Command ===> ____________________________________________________________________

 From Partitioned, Sequential or VSAM Data Set:
    Data set name . . . . . 'REDBK2.TEST.SEQFILE'____________________________
    Member  . . . . . . . . ________        (Blank or pattern for member list)
    Volume serial . . . . . ______
    Start key . . . . . . . ________________________________ key or slot
    Skip count  . . . . . . _______      number of records to be skipped
    Copy count  . . . . . . ALL____      number of records to be copied
 From Copybook or Template:
    Data set name . . . . . ____________________________________________
    Member  . . . . . . . . _________       (Blank or pattern for member)
 Processing Options:
    Copybook/template usage      Enter "/" to select option
    2  1. Above                  _  Edit template
       2. None                   _  Use proc *_______
       3. Create dynamic         _  Ignore length mismatch
                                 _  Batch Execution

  F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
  F9=Swap     F10=Actions  F12=Cancel
```

Figure 23-1 Initial Copy utility screen

Enter the name of the file you wish to copy to in Figure 23-2. If this file does not exist, you are presented with screens that allow you to allocate it.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 Copy from REDBK2.TEST.SEQFILE
 Command ===> _____________________________________________________________

 To Partitioned, Sequential or VSAM Data Set:
    Data set name . . . . . 'REDBK2.TEST.SEQFILE2'_____________________
    Member  . . . . . . . . ________
    Volume serial . . . . . ______

 Processing Options:
    Disposition             Execution "/" options
    2  1. Old or Reuse      /  Replace members
       2. Mod

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 23-2 Specifying the output data set

Successful copying of the file is reported as shown in Figure 23-3. DFSORT has been automatically used internally by File Manager to perform the copy.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 File Manager                    Copy Utility
 Command ===> _____________________________________________________________
 67 record(s) successfully copied
 From Partitioned, Sequential or VSAM Data Set:
    Data set name . . . . . 'REDBK2.TEST.SEQFILE'____________________
    Member  . . . . . . . . ________      (Blank or pattern for member list)
    Volume serial . . . . . ______
    Start key . . . . . . . ______________________________ key or slot
    Skip count  . . . . . . 0_______     number of records to be skipped
    Copy count  . . . . . . ALL_____     number of records to be copied
 From Copybook or Template:
    Data set name . . . . . ______________________________________________
    Member  . . . . . . . . __________     (Blank or pattern for member)
 Processing Options:
    Copybook/template usage     Enter "/" to select option
    2  1. Above                 _  Edit template
       2. None                  _  Use proc *_______
       3. Create dynamic        _  Ignore length mismatch
                                _  Batch Execution

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 23-3 Resolution of the simple copy process

23.1.4 Selective copying

Many data set copy requirements are selective in that only some of the records from the input data set are required in the output. You can still use DFSORT for this, simply by providing a DFSORT procedure. To indicate the requirement for a procedure, select Use proc as shown in Figure 23-4. The asterisk (*) indicates that the procedure will be dynamically created.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
File Manager                        Copy Utility
Command ===> ____________________________________________________________________

From Partitioned, Sequential or VSAM Data Set:
   Data set name . . . . . 'REDBK2.TEST.SEQFILE'____________________________
   Member  . . . . . . . . ________       (Blank or pattern for member list)
   Volume serial . . . . . ______
   Start key . . . . . . . ______________________________ key or slot
   Skip count  . . . . . . ________     number of records to be skipped
   Copy count  . . . . . . ALL_____     number of records to be copied
From Copybook or Template:
   Data set name . . . . . ________________________________________________
   Member  . . . . . . . . __________     (Blank or pattern for member)
Processing Options:
   Copybook/template usage     Enter "/" to select option
   2  1. Above                 _  Edit template
      2. None                  /  Use proc *______
      3. Create dynamic        _  Ignore length mismatch
                               _  Batch Execution

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 23-4 Selective copying

Pressing the Enter key provides you with the panel shown in Figure 23-5, in which the required selection statements can be entered.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------------
EDIT       SYS03301.T110909.RA000.REDBK2.R0102419          Columns 00001 00072
Command ===> ____________________________________________ Scroll ===> CSR
Enter your REXX procedure statements
==MSG> -Warning- The UNDO command is not available until you change
==MSG>           your edit profile using the command RECOVERY ON.
'''''' *FASTPROC
''''''  INCLUDE COND=(1,2,CH,EQ,C'04')
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 23-5 Selection statement specified

Proceeding with the copy, Figure 23-6 shows the result of the selection of only those records with 04 in columns 1 and 2 — only 12 records have been copied.

```
 Process   Options   Help
File Manager                        Copy Utility
Command ===> _____________________________________________________________
12 record(s) successfully copied
From Partitioned, Sequential or VSAM Data Set:
   Data set name . . . . . 'REDBK2.TEST.SEQFILE'
   Member  . . . . . . . . ________       (Blank or pattern for member list)
   Volume serial . . . . . ______
   Start key . . . . . . . ______________________________ key or slot
   Skip count  . . . . . . 0______      number of records to be skipped
   Copy count  . . . . . . ALL_____     number of records to be copied
From Copybook or Template:
   Data set name . . . . . ______________________________________________
   Member  . . . . . . . . __________     (Blank or pattern for member)
Processing Options:
   Copybook/template usage     Enter "/" to select option
   2  1. Above                 _  Edit template
      2. None                  /  Use proc *______
      3. Create dynamic        _  Ignore length mismatch
                               _  Batch Execution

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 23-6 Result of selection

23.1.5 How to perform a global find and replace in a PDS

File Manager's Find/Change Utility allows you to search for or change a string in a partitioned data set (PDS), a VSAM data set, or a sequential file.

Scenario: Global update of JCL

This section describes both online and batch approaches to the global update of a JCL deck.

Using File Manager online

This scenario presents the succession of panels to change all references of one user ID to another in all the members of a library. You reach the File/Change Utility panel by selecting option **3.6** on the Main panel of File Manager and entering data in the appropriate fields as shown below.

This scenario describes how to change all the members (* in the Member field) of the library REDBK2.FM.CNTL:

1. Since all these members are source JCL, enter a forward slash (**/**) in the JCL Source format field as shown in Figure 23-7.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 File Manager                 Find/Change Utility
 Command ===> ___________________________________________________________ Scroll PAGE

 Input Partitioned, Sequential or VSAM Data Set:
    Data set name  . 'REDBK2.FM.CNTL'
    Member . . . . . *_______     (Blank - selection, pattern - process list)
    Volume serial  . ______       (If not cataloged)

 Listing data set  . SRCHFOR.LIST

 Enter "/" to select option                              Listing Option
 / JCL Source format         _ Immediate change          1  1. Long
 _ Use REXX proc ________    _ Batch execution              2. Summary
 _ Stats off                 _ Exclusive (DISP=OLD)
 Process List:
 Sel Name     Prompt   Size   Created          Changed        ID
    ****  End of data  ****


  F1=Help      F2=Split     F3=Exit      F4=CRetriev  F5=Refresh   F7=Up
  F8=Down      F9=Swap     F12=Cancel
```

Figure 23-7 File/Change utility panel

2. Enter the change command `C REDBK2 PETERH ALL` on the command line and press Enter. As we did not ask for Immediate change, we get a confirmation panel displaying all the proposed modifications as shown in Figure 23-8.

```
   Menu  Utilities  Compilers  Help
  ------------------------------------------------------------------------------
  BROWSE    REDBK2.SRCHFOR.LIST                      Line 00000000 Col 001 080
  Command ===> ___________________________________________________ Scroll ===> PAGE
 ********************************* Top of Data **********************************
 IBM File Manager for z/OS
 Find/Change Listing DSN: REDBK2.FM.CNTL

   -- Find/Change summary section --
   Records found: 12 Records processed: 98
   Members w/recs: 3 Members wo/recs: 3
   Search cols: 1:80 Longest line: 80
   Edit options in effect: CAPS:OFF

                  Count  Value
   Found    :        12  >REDBK2<
   Changed  :        12  >PETERH<
 ------------------------------------------------------------------------
 ******************************** Bottom of Data ********************************


   F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
  F10=Left   F11=Right  F12=Cancel
```

Figure 23-8 Proposed modifications

3. To confirm the changes, press F3, followed by Enter to save, or F3 to cancel. After confirmation, File Manager completes the job as shown in Figure 23-9. We see that 12 changes have been made in 3 members.

```
 Process   Options   Help
File Manager                 Find/Change Utility
Command ===> _____________________________________________________ Scroll CSR
12 changes made in 3 members
Input Partitioned, Sequential or VSAM Data Set:
   Data set name  . 'REDBK2.FM.CNTL'
   Member . . . . . *_______     (Blank - selection, pattern - process list)
   Volume serial  . ______       (If not cataloged)

Listing data set  . SRCHFOR.LIST

Enter "/" to select option                              Listing Option
/ JCL Source format          _ Immediate change         2  1. Long
_ Use REXX proc *_______     _ Batch execution             2. Summary
_ Stats off                  _ Exclusive (DISP=OLD)
Process List:
Sel Name     Prompt   Size   Created          Changed        ID
_   COMPDT   Selected
_   JCL1     Changed
_   MTVSAM   Changed     18 2003/10/10  2003/10/21 11:26:45  REDBK2
_   MTVSAM1  Changed
 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F5=Refresh   F7=Up
 F8=Down      F9=Swap     F12=Cancel
```

Figure 23-9 Confirmation of changes

4. You can then refer to the listing data set, SRCHFOR.LIST, for the details of the changes.

Tip: The first time you ask File Manager to perform a find/change, request a “batch execution” (insert a / in this field on the panel referenced in the following figures), and save the displayed JCL in your JCL library. You then can reuse it as needed.

Using File Manager in batch

Example 23-1 presents the batch job generated by File Manager to change all references of one user ID to another, in all the members of a library.

Example 23-1 Batch job to change one user ID to another

```
//*
//FILEMGR  EXEC PGM=FMNMAIN
//STEPLIB  DD DSN=FMN.V4R1M0.SFMNMOD1,DISP=SHR
//*        DD DSN=IGY.SIGYCOMP,DISP=SHR
//SYSPRINT DD SYSOUT=*
//FMNTSPRT DD SYSOUT=*
//SYSTERM  DD SYSOUT=*
//SYSIN    DD *
$$FILEM FCH ,
$$FILEM MEMBER=*,
$$FILEM LIST=SUMMARY,
$$FILEM JCL=YES,
$$FILEM DSNIN=REDBK2.FM.CNTL
BNDS 1 80
CAPS OFF
C REDBK2 PETERH ALL
/+
```

More complex scenario

We decided to use the File Manager function Data Set Copy (DSC), along with some simple REXX code, to perform a very selective global find and replace.

Assume the following scenario: As a production support specialist, you need to help an application developer set up a portion of a job stream for a User Acceptance Test (UAT).

You need to take the production job card members, not the procedures, that were created for production, and convert them to UAT standards. The changes identified in Table 23-1 need to be made.

Table 23-1 JCL modifications to make in selected members of a PDS

Field	From	To
OPC user ID	ZOPCPRD	ZUATUSR
MSGCLASS	S	J
Symbolic	MODEP='P'	MODEP='U'
Member name	Do not copy if it ends in 'T'	

Any job that invokes the program FTP must be copied, but must not be changed. These jobs contain the string XMIT2 in the accounting information parameter of the JOB card. To ensure that no transmission occurs, the program name in the procedure will be changed from FTP to IEFBR14. How to make this change is not covered as part of this scenario.

How to set up the batch job

We decided to pre-allocate a separate output file for the changed members. The JCL is shown in Example 23-2.

Example 23-2 Batch job to search for string

```
//*
//* FILE MANAGER BATCH: SEARCH FOR STRING
//*
//STEP01   EXEC PGM=FMNMAIN
//STEPLIB  DD DSN=FMN.V4R1M0.SFMNMOD1,DISP=SHR
//*        DD DSN=IGY.SIGYCOMP,DISP=SHR
//SYSPRINT DD  SYSOUT=*
//FMNTSPRT DD  SYSOUT=*
//SYSTERM  DD  SYSOUT=*
//DDIN     DD  DISP=SHR,DSN=DAVIN6.SVLSAMP.JCL
//DDOUT    DD  DISP=SHR,DSN=DAVIN6.SVL@UAT.JCL
//SYSIN    DD  *
$$FILEM DSC  INPUT=DDIN,MEMBER=*,
$$FILEM      OUTPUT=DDOUT,REPLACE=YES,
$$FILEM      PROC=*
IF LEFT(INREC,3) <> '//*' THEN DO
   SELECT
      WHEN CO(INREC,' JOB ') & ,
           CO(INREC,'XMIT2') THEN DO
            PRINT('MEMBER NOT CHANGED BECAUSE IT IS FTP','CHAR')
           RETURN
      END
      WHEN CO(INREC,' PROC ') THEN DO
           PRINT('MEMBER NOT COPIED BECAUSE IT IS A PROC','CHAR')
```

```
            RETURN 'STOP IMMEDIATE'
       END
       WHEN RIGHT(STRIP(SUBSTR(INREC,3,8)),1) = 'T' THEN DO
            PRINT('MEMBER NOT COPIED BECAUSE IT IS FOR TEST','CHAR')
            RETURN 'STOP IMMEDIATE'
       END
OTHERWISE DO
            OUTREC = CHANGE(INREC,'ZOPCPRD','ZUATUSR')
            OUTREC = CHANGE(OUTREC,'MSGCLASS=S','MSGCLASS=J')
            OUTREC = CHANGE(OUTREC,"MODE='P'","MODE='U'")
            WRITE()
       END
   END
END
/+
/*
```

What is happening in this step

The file DDIN is the input file that contains all of the members, which consist of production and test jobs and procedures. While this is not something an application programmer would do in the real world, it does suffice for this example. The default output file has the DD name DDOUT.

The File Manager program keyword DSC is used to invoke the Data Set Copy function. The input and output files are identified, and the keyword PROC is used to indicate that an in-stream REXX routine is being supplied.

The File Manager control cards indicate that all of the members should be selected, and that if any already exist in the output file, they should be replaced. This allows us to run this sample repeatedly.

The first line of the REXX routine selects only non-comment lines for processing.

Then, three conditions are applied to the input record:

1. It is searched to see if it contains the strings JOB and XMIT2. If it does, the member is copied but is not changed.
2. It is searched to see if it contains the string PROC. If it does, the member is not copied.
3. It is parsed to determine if the last character of the job name is the letter T. If it is, the member is not copied.

Otherwise, the appropriate changes are made to the JCL and are written to the output file.

Reviewing the report output

The key portion of the batch job's output report is shown in Example 23-3.

Note: Each page in the report starts with the title, "IBM File Manager for z/OS and OS/390." Also, this report has been edited (represented by facing sets of slashes) to fit within the confines of this section.

Example 23-3 Report of global find and replace

```
IBM File Manager for z/OS and OS/390
$$FILEM DSC  INPUT=DDIN,MEMBER=*,
$$FILEM      OUTPUT=DDOUT,REPLACE=YES,
$$FILEM      PROC=*
Member SVLD011P - Copied
12 record(s) copied: 0 truncated: 0 fields truncated
                            MEMBER NOT COPIED BECAUSE IT IS FOR TEST
Member SVLD011T - Copied
0 record(s) copied: 0 truncated: 0 fields truncated
Member SVLD012P - Copied
12 record(s) copied: 0 truncated: 0 fields truncated
                            MEMBER NOT COPIED BECAUSE IT IS FOR TEST
Member SVLD012T - Copied
0 record(s) copied: 0 truncated: 0 fields truncated
Member SVLD021P - Copied
11 record(s) copied: 0 truncated: 0 fields truncated
                            MEMBER NOT COPIED BECAUSE IT IS FOR TEST
Member SVLD021T - Copied
0 record(s) copied: 0 truncated: 0 fields truncated
//\\
\\//
                            MEMBER NOT COPIED BECAUSE IT IS A PROC
Member SVLD104 - Copied
0 record(s) copied: 0 truncated: 0 fields truncated
Member SVLD104C - Copied
16 record(s) copied: 0 truncated: 0 fields truncated
Member SVLD104D - Copied
16 record(s) copied: 0 truncated: 0 fields truncated
                            MEMBER NOT CHANGED BECAUSE IT IS FTP
Member SVLD104E - Copied
16 record(s) copied: 0 truncated: 0 fields truncated
37 member(s) copied: 0 member(s) replaced: 0 member(s) error
```

The first page contains a copy of the input commands. This is followed by a series of status messages that indicate the processing performed during the copy.

The DSC function writes out any of the PRINT statements from the REXX routine before it writes its own statistics. These contain the name of the member and the action taken (copied or replaced), followed by the number of records copied.

We found that when the number of records is zero, the member is not copied, despite what the action indicates.

File Manager external REXX functions used in this routine

A brief explanation of each of the File Manager external REXX functions that were used in this routine follows.

DSC

This copies data from one file to another. The file can be any of the File Manager supported structures (VSAM, QSAM, or PDS).

CONTAIN
If the string being searched for is contained in the input record, then CONTAIN returns 1. Otherwise, CONTAIN returns 0.

PRINT
This prints the string in a specified format to the output report.

WRITE
Writes a record to the specified data sets. If the WRITE function is successful, it returns a value of 0. If the WRITE function is unsuccessful, it raises the REXX syntax error condition.

RETURN
In REXX, you can use the RETURN or EXIT instruction to leave a procedure; RETURN is preferred for performance reasons. You can optionally specify a character string as a parameter on the instruction. This character string is returned to the caller of the procedure.

STOP IMMEDIATE
The character string STOP IMMEDIATE tells File Manager to terminate the current function without writing the current record to the primary output data set. When used with DSC, the member is not copied.

23.1.6 How to create one VSAM file using another as a model

When modifications to your application require you to create a new file, or when a testing effort requires a clean copy, you can model it based on an existing file that has common attributes.

In this example, we use File Manager to create one VSAM file by using another as a model:

1. Access File Manager MVS in your ISPF session.
2. Go to Catalog Services (option 3.4) and list the VSAM files for your application.
3. Select a file that has attributes which resemble those of the file you want to create.

> **Note:** If you are going to use the pull-down menus, your cursor must be on the same line as the data set name. You can either scroll the list until the file you want to work with is the first one displayed, or position your cursor and press PF6 (PROCESS) to display the process pull-down.

4. Type `LIST` in the line commands area, or select the **Process** pull-down and select **List**.

   The VSAM Entry Detail panel is displayed with information for the current file as shown in Example 23-4.

Example 23-4 VSAM entry panel information

```
File Manager                  VSAM Entry Detail
Command ===>

VSAM Catalog Entry:
   Data set name  . . 'REDBK2.TEST.VSAM'
   Catalog ID . . . . 'SYS1.ICFCAT.VSTF2U0'

VSAM Associations:
   VSAM data type . . KSDS         KSDS, ESDS, RRDjusS, VRRDS or LDS
   Creation date  . . 2003.286     Expiration date (NONE)
```

```
   Data . . . . . . . 'REDBK2.TEST.VSAM.DATA'
   Index  . . . . . . 'REDBK2.TEST.VSAM.INDEX'

   AIX  . . . . . . .
     Data . . . . . .
     Index  . . . . .
   Path . . . . . . .

SMS Definitions:
   SMS managed  . . . Y
   Data class . . . . *UNKNOWN
   Storage class  . . STF2CLS1
   Management class . *UNKNOWN
   Last backup date . 0000.000.0000

VSAM Cluster Attributes:
   CI size  . . . . . 20480    size of the data control intervals
   Buffer space . . . 41472    buffer space to be allocated at open time

   Share options .    Cross region . 1          Cross systems  3
   Process options    Reuse  . . . . N          Recovery . . . Y
                      Spanned  . . . N          Erase  . . . . N
                      Extended . . . N          Extended addr  N
                      Compressed . . N          Writecheck . . N

VSAM Data Allocation:
   Allocation unit  . TRK      REC, KB, MB, TRK, or CYL

   Space . . . . .    Primary . . 1            Secondary . 1
   Record size . .    Average . . 4089         Maximum . . 4089
   Free space . .     % of CI . . 0            % of CA . . 0

   Volume serial(s) . STF203
   Device type(s) . . 3390

VSAM Key Definition:
   Key length . . . . 10               Key offset  . . 0

   Low key 1 . . .                  High key 1  . .
    (hex)  2 . . .                   (hex)   2  . .
           3 . . .                           3  . .
           4 . . .                           4  . .
           5 . . .                           5  . .

VSAM Index Allocation:
   CI size  . . . . . 512      size of the index control intervals
   Allocation unit  . TRK      REC, KB, MB, TRK, or CYL

   Space . . . . .    Primary . . 1            Secondary . 1

   Volume serial(s) . STF203
   Device type(s) . . 3390
```

5. Press PF3 to return to the Data Set List panel.

6. Type `DEFINE` in the line commands area, or select the **Process** pull-down and select **Define**.

   The VSAM Define panel is displayed as shown in Figure 23-10.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 File Manager                        VSAM Define
 Command ===> _____________________________________________________________________
 Press ENTER to define the catalog entry or EXIT to cancel
 VSAM Catalog Entry:
    Data set name  . . 'REDBK2.TEST.VSAM'                                 
    Catalog ID . . . . 'SYS1.ICFCAT.VSTF2U0'                              
                                                                   More:     +
 VSAM Associations:
    VSAM data type . . KSDS       Expiration date  . (NONE)
    Data . . . . . . . 'REDBK2.TEST.VSAM.DATA'                            
    Index  . . . . . . 'REDBK2.TEST.VSAM.INDEX'                           
 VSAM Cluster Attributes:
    Key length . . . . 10         Key offset . . . . 0
    CI size  . . . . . 20480      size of the data control intervals
    Buffer space . . . 41472      buffer space to be allocated at open time
    Shr cross region . 1          cross system . . . 3          Reuse . . . . N
    Recovery . . . . . Y          Spanned  . . . . . N          Erase . . . . N
    Writecheck . . . . N
 VSAM Data Allocation:
    Allocation unit  . TRK        REC, KB, MB, TRK, or CYL
  F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Up        F8=Down
  F9=Swap     F10=Actions  F12=Cancel
```

Figure 23-10 VSAM Define panel

7. Change the data set name, as well as the data and the index names, to the new file's corresponding names, by typing over the existing information.

   Note: In the lab, if we did not erase the value in the Catalog ID field, we could not locate the file without explicitly pointing to the catalog and the volume. Have your system programmer validate the rules at your site with your storage management group during a post-installation review.

8. Modify any of the other file attributes as needed.
9. Press Enter to define your new VSAM file.

Reviewing this example

In this example, we created a new VSAM file with attributes based on an existing VSAM file.

Notice the following characteristics of this utility:

- All of this processing is performed in the foreground.
  - There is no option to perform the file allocation in batch.
  - This utility does not create batch JCL; nor does it create IDCAMS control cards.
- There is no DELETE associated with the DEFINE process.

  If the new VSAM file you want to create already exists, you will receive an error message after you press Enter.

23.1.7 How to initialize a VSAM file with low-value records

When you create a VSAM file for a CICS application, you usually need to initialize it with a low-value record. You probably create a control card (or sequential file) containing binary zeros that matches the record length of the file, so you can REPRO the record into the new file.

In this example, we use File Manager to perform that process, so it does not depend on different control cards for each file size.

To start, you need an empty VSAM file. You can use the method described previously or IDCAMS control cards.

Start the process by following these steps:

1. Access File Manager in your ISPF session.
2. Go to Data Create Utility (option **3.1**).
3. Enter the name of the new VSAM file.
4. Indicate the number of records to be created.
5. Specify a Fillchar of x'00' (binary zeros).

> **Tip:** Do not make the mistake of selecting a Fillchar of BIN, thinking it creates binary zeros — you *will* get binary data such as: `x'0102030405060708090A0B0C0D0E0F10...'`

6. Specify a Disposition of Old.
7. Specify the Copybook or template of None.
8. Select the option for Batch execution.

   When you are finished, your panel should resemble Figure 23-11.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
File Manager                 Data Create Utility
Command ===> ____________________________________________________________________
                                                                   More:     +
Output Partitioned, Sequential or VSAM Data Set:
   Data set name . . . . . 'REDBK2.TEST.PDHVSAM'
   Member  . . . . . . . . ________     (Blank or pattern for member list)
   Volume serial . . . . . ______       (If not cataloged)
   Record length . . . . . 80____       Optional record length for RECFM V
   Records . . . . . . . . 1_______     number of records
   Fillchar  . . . . . . . X'00'        char or hex value, AN, BIN, or RAND
   Key position  . . . . . _______      if sequence field desired
   Key length  . . . . . . 8            length from 1 to 9
   Key increment . . . . . 10______     increment value
Copybook or Template:
   Data set name . . . . . ____________________________________________
   Member  . . . . . . . . ________     (Blank or pattern for member list)
Processing Options:
   Disposition       Copybook or template          Enter "/" to select option
   2  1. Old/Reuse   3  1. Above                   _  Edit template
      2. Mod            2. Previous                _  Batch execution
 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 23-11 Data Create utility panel to initialize VSAM file with binary zeroes

Note: For a VSAM file, add only *one* record. Otherwise, duplicate keys will generate the following error message: `VSAM PUT RC X'08', Error Code X'0C'`.

9. Press Enter.

   The JCL for the batch job, using Data Set Generate (DSG) is displayed. It should resemble Example 23-5.

   Example 23-5 JCL for batch job

```
//FMNMAIN  EXEC PGM=FMNMAIN
//STEPLIB  DD DSN=FMN.V4R1M0.SFMNMOD1,DISP=SHR
//*        DD DSN=IGY.SIGYCOMP,DISP=SHR
//SYSPRINT DD SYSOUT=*
//FMNTSPRT DD SYSOUT=*
//SYSTERM  DD SYSOUT=*
//SYSIN    DD *
$$FILEM DSG DSNOUT=REDBK2.TEST.PDHVSAM,
$$FILEM FILLCHAR=X'00',
$$FILEM DISP=OLD,
$$FILEM NLRECS=1
```

10. Submit the batch job.

Save a copy of this JCL.

What is happening in this step

No additional data set is needed for this batch job, aside from the standard File Manager load library.

The File Manager program keyword DSG is used to invoke the Data Set Generate function. The output file is identified, along with the keywords, to indicate how the file should be loaded.

The fill character is specified as a hexadecimal zero, and the number of logical records is specified as one.

Reviewing the report output

The key portion of the batch job's output report is listed in Example 23-6.

Note: Each page in the report starts with the title, "IBM File Manager for z/OS and OS/390."

Example 23-6 Report of DSG low value record creation

```
IBM File Manager for z/OS
$$FILEM DSG DSNOUT=REDBK2.TEST.VSAMPDH,
$$FILEM FILLCHAR=X'00',
$$FILEM DISP=OLD,
$$FILEM NLRECS=1
1 record(s) written
```

Here you can see the input commands listed, followed by a message that states that the requested number of records were written to the output file.

File Manager functions used in this routine

A brief explanation of the File Manager function used in this routine follows.

DSG

Initializes VSAM data sets, sequential data sets, and PDS members.

You specify the output data set name, the disposition, the number of logical records, and the fill character.

To fill each byte of each record with data, specify one of the following:

char	To write a character, such as 0, in each byte
X'cc'	To write a binary character, such as X'04', in each byte
AN	To write alphanumeric characters (A to Z and 0 to 9)
BIN	To write binary characters (X'00' to X'FF')
RAND	To write random binary characters (X'00' to X'FF')

The default is a blank.

Modify the JCL for generic use

To reuse the code from this example, convert the JCL into a procedure. It can then be used to initialize all of your VSAM files. The next example shows the modifications we made.

In Example 23-7 we added the PROC statement and changed the DSG parameter DSNOUT to OUTPUT. This lets you use an override statement in the JCL to point to your file.

Note: We would place the SYSIN statements in a member of a control card library for a production batch job.

Example 23-7 DSG batch step converted to a proc

```
//DSGPROC  PROC
//FILEMAN  EXEC PGM=FMNMAIN
//STEPLIB  DD DSN=FMN.V4R1M0.SFMNMOD1,DISP=SHR
//*        DD DSN=IGY.SIGYCOMP,DISP=SHR
//DDOUT DD DSN=DSNOUT=REDBK2.TEST.VSAMPDH,DISP=SHR
//SYSPRINT DD SYSOUT=*
//FMNTSPRT DD SYSOUT=*
//SYSTERM  DD SYSOUT=*
//SYSIN    DD *
$$FILEM DSG OUTPUT=DDOUT,
$$FILEM FILLCHAR=x'00',
$$FILEM DISP=OLD,
$$FILEM NLRECS=1
```

23.1.8 How to populate this newly created file

We can now insert records into the newly created file by copying from an existing file, or by using a template to generate the data.

Copy data from QSAM to VSAM

Perform the following steps to copy data from QSAM to VSAM:

1. Fill the Copy Utility panel (option **3.3**) with the QSAM source file name as shown in Figure 23-12. Press Enter.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 File Manager                      Copy Utility
 Command ===> ____________________________________________________________________

 From Partitioned, Sequential or VSAM Data Set:
    Data set name . . . . . 'REDBK3.REDBOOK.PDPAK.COMPMAST'_____________
    Member  . . . . . . . . ________       (Blank or pattern for member list)
    Volume serial . . . . . ______
    Start key . . . . . . . ______________________________ key or slot
    Skip count  . . . . . . ________     number of records to be skipped
    Copy count  . . . . . . ALL_____     number of records to be copied
 From Copybook or Template:
    Data set name . . . . . ____________________________________________
    Member  . . . . . . . . __________     (Blank or pattern for member)
 Processing Options:
    Copybook/template usage     Enter "/" to select option
    2  1. Above                 _  Edit template
       2. None                  _  Use proc ________
       3. Create dynamic        _  Ignore length mismatch
                                _  Batch Execution

  F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
  F9=Swap     F10=Actions  F12=Cancel
```

Figure 23-12 The Copy Utility panel

2. Fill the next panel, as shown in Figure 23-13, with the name of the target VSAM file.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 Copy from REDBK3.REDBOOK.PDPAK.COMPMAST
 Command ===> ____________________________________________________________________

 To Partitioned, Sequential or VSAM Data Set:
    Data set name . . . . . 'REDBK2.TEST.PDHVSAM'____________________________
    Member  . . . . . . . . ________
    Volume serial . . . . . ______

 Processing Options:
    Disposition             Execution "/" options
    2  1. Old or Reuse      _  Replace members
       2. Mod





  F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
  F9=Swap     F10=Actions  F12=Cancel
```

Figure 23-13 Target VSAM file

3. We used the empty file created previously. If you name a file that does not exist, File Manager displays the VSAM Define panel. After selecting KSDS, you will see a panel similar to Figure 23-14.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 File Manager                        VSAM Define
 Command ===> ____________________________________________________________________
 Press ENTER to define the catalog entry or EXIT to cancel
 VSAM Catalog Entry:
    Data set name  . . 'REDBK2.TEST.PDHVSAM'                                    
    Catalog ID . . . . _                                                         
                                                                        More:     +
 VSAM Associations:
    VSAM data type . . KSDS       Expiration date  . ________
    Data . . . . . . . ____________________________________________
    Index  . . . . . . ____________________________________________
 VSAM Cluster Attributes:
    Key length . . . . 80___      Key offset . . . . _____
    CI size  . . . . . 4000_      size of the data control intervals
    Buffer space . . . ________   buffer space to be allocated at open time
    Shr cross region . _          cross system . . . _          Reuse . . . . _
    Recovery . . . . . _          Spanned  . . . . . _          Erase . . . . _
    Writecheck . . . . _
 VSAM Data Allocation:
    Allocation unit  . TRK        REC, KB, MB, TRK, or CYL
  F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Up        F8=Down
  F9=Swap     F10=Actions  F12=Cancel
```

Figure 23-14 VSAM Define panel

In addition to the online capabilities, you can also use the batch functions of File Manager.

4. After pressing Enter in the previous panel, you get confirmation as shown in Figure 23-15.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 File Manager                        Copy Utility
 Command ===> ____________________________________________________________________
 7 record(s) copied: 0 truncated: 0 fields truncated
 From Partitioned, Sequential or VSAM Data Set:
    Data set name . . . . . 'REDBK3.REDBOOK.PDPAK.COMPMAST'                     
    Member  . . . . . . . . ________       (Blank or pattern for member list)
    Volume serial . . . . . ______
    Start key . . . . . . . ______________________________ key or slot
    Skip count  . . . . . . 0_______      number of records to be skipped
    Copy count  . . . . . . ALL_____      number of records to be copied
 From Copybook or Template:
    Data set name . . . . . ______________________________________________
    Member  . . . . . . . . __________     (Blank or pattern for member)
 Processing Options:
    Copybook/template usage     Enter "/" to select option
    2  1. Above                 _  Edit template
       2. None                  _  Use proc ________
       3. Create dynamic        _  Ignore length mismatch
                                _  Batch Execution

  F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
  F9=Swap     F10=Actions  F12=Cancel
```

Figure 23-15 Copy completed

Note: File Manager, from Version 4 onwards, automatically uses the DFSORT COPY option for this, and similar, simple data set to data set copy actions, provided that DFSORT is available, and at the required maintenance level. This is because the input data set is not partitioned and the record format is not undefined.

You can now browse this data set to check that the content has correctly been loaded. Specify option **1** on the Main panel. The capability to browse without any template can be used, as shown in Figure 23-16.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 File Manager Browse REDBK2.TEST.PDHVSAM
 Command ===> _____________________________________________________ Scroll CSR
        Type KSDS     Key ____________________________ RBA __________ Format CHAR
        Col 1________
 RBA          Len    <===+====>----+----2----+----3----+----4----+----5----+----
 ****  Top of data   ****
 0            80     ............................................................
 80           90     Casey_Import_Export 00079.0000077.0000078.0000072.0000070.00
 170          90     Glass_and_Luget_plc 00019.0000022.0000025.0000020.0000016.00
 260          90     Headworth_Electrical00124.0000131.0000133.0000133.0000133.00
 350          90     IBM                 00163.0000163.0000162.0000160.0000161.00
 440          90     ShareSelect         00119.0000120.0000118.0000116.0000116.00
 530          90     SportSelect         00224.0000224.0000220.0000217.0000215.00
 620          90     Veck_Transport      00036.0000034.0000034.0000035.0000037.00
 ****  End of data   ****



  F1=Help      F2=Zoom      F3=Exit      F4=CRetriev  F5=RFind     F7=Up
  F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 23-16 Browse without template

However, it is nicer to look at it using a template as shown in Figure 23-17.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 File Manager Browse REDBK2.TEST.PDHVSAM
 Command ===> _____________________________________________________ Scroll CSR
        Type KSDS     Key ____________________________ RBA __________ Format TABL
                      Customer Name        Invoice Oct     Sept    Aug
 RBA          Len     #2                   #3      #4      #5      #6
                      AN 1:20              AN 21:5 AN 27:7 AN 35:7 AN 43:7
                      <---+----1----+----> <--->   <---+-> <---+-> <---+->
 ****  Top of data   ****
 0            80     ..................... .....   ....... ....... .......
 80           90     Casey_Import_Export  00079   0000077 0000078 0000072
 170          90     Glass_and_Luget_plc  00019   0000022 0000025 0000020
 260          90     Headworth_Electrical 00124   0000131 0000133 0000133
 350          90     IBM                  00163   0000163 0000162 0000160
 440          90     ShareSelect          00119   0000120 0000118 0000116
 530          90     SportSelect          00224   0000224 0000220 0000217
 620          90     Veck_Transport       00036   0000034 0000034 0000035
 ****  End of data   ****



  F1=Help      F2=Zoom      F3=Exit      F4=CRetriev  F5=RFind     F7=Up
  F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 23-17 Browse with a template

When you are browsing or editing a file, you can use commands such as SORT to present the data differently. The default sorting for a VSAM KSDS is the keys in ascending order (SORT KEY). You can sort the records according to the order of the data in a column range or a field if you use a template. Figure 23-18 presents the result of the `SORT #2 D`. (D is for descending.)

```
 Process    Options    Help
File Manager Browse REDBK2.TEST.PDHVSAM
Command ===> _                                                Scroll CSR
       Type KSDS     Key ___________________________ RBA _________ Format TABL
                    Customer Name        Invoice Oct      Sept     Aug
RBA          Len    #2                   #3      #4       #5       #6
                    AN 1:20              AN 21:5 AN 27:7  AN 35:7  AN 43:7
                    <---+----1----+----> <--->    <---+->  <---+->  <---+->
****  Top of data   ****
620          90     Veck_Transport       00036   0000034  0000034  0000035
530          90     SportSelect          00224   0000224  0000220  0000217
440          90     ShareSelect          00119   0000120  0000118  0000116
350          90     IBM                  00163   0000163  0000162  0000160
260          90     Headworth_Electrical 00124   0000131  0000133  0000133
170          90     Glass_and_Luget_plc  00019   0000022  0000025  0000020
80           90     Casey_Import_Export  00079   0000077  0000078  0000072
0            80     ..................... .....   .......  .......  .......
****  End of data   ****


 F1=Help      F2=Zoom      F3=Exit      F4=CRetriev  F5=RFind     F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 23-18 SORTed data

Generate data using a template

Instead of using existing data, we can load the new VSAM with computer-generated data, following the rules given in a template such as a copybook.

This time, use the Data Creation utility, option **3.1** on the Main panel, as shown in Figure 23-19.

```
_ Process    Options    Help
 File Manager                 Data Create Utility                Template saved
                                                                    More:     +
 Output Partitioned, Sequential or VSAM Data Set:
    Data set name . . . . . DEMOS.PDPAK.CUSTTEST
    Member  . . . . . . . . ________      (Blank or pattern for member list)
    Volume serial . . . . . ______        (If not cataloged)
    Record length . . . . . ______        Optional record length for RECFM V
    Records . . . . . . . . 10            number of records
    Fillchar  . . . . . . . _____         char or hex value, AN, BIN, or RAND
    Key position  . . . . . ______        if sequence field desired
    Key length  . . . . . . 8             length from 1 to 9
    Key increment . . . . . 10            increment value
 Copybook or Template:
    Data set name . . . . . 'DAVIN25.DEMOS.TEMPLATE'
    Member  . . . . . . . . CUST001
 Processing Options:
    Disposition        Copybook or template          Enter "/" to select option
    3  1. Old          1  1. Above                   _  Edit copybook or template
 Command ===> _
  F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
  F9=Swap     F10=Actions  F12=Cancel
```

Figure 23-19 Data Create panel

You can browse the result using option **1** on the Main panel as shown in Figure 23-20, which shows that there are default values for data generation in the template. Edit the template if necessary.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
File Manager                    Data Set Browse

RBA ___________    Key 'TEST ADTTEST ADTTEST ADTTEST'          Zoom Format SNGL
VOLSER SMS001   Type KSDS   DSNAME DAVIN25.DEMOS.PDPAK.CUSTTEST
                                                     Top Field is 1    of 14
RBA        Len
 #4 CUST-NM        : TEST ADTTEST ADTTEST ADTTEST ADTTEST ADTTEST ADTTEST ADTT
                     EST
 #5 KEYREC-DOT     : x
 #6 COMP-NM        : aaabbbcccdddeeefff
 #7 DEC-NO-SHARES  :  0000010
 #8 BUY-FROM       : Unknown
 #9 BUY-FROM-NO    : 0090
#10 BUY-TO         : AAABBBCC
#11 BUY-TO-NO      : 3192
#12 SELL-FROM      : Unknown1
#13 SELL-FROM-NO   : 0000
#14 SELL-TO        : Unknown2
Command ===> _____________________________________________________ Scroll PAGE
 F1=Help      F2=Zoom      F3=Exit      F4=CRetriev  F5=RFind     F7=Up
 F8=Down      F9=Swap     F12=Cancel
```

Figure 23-20 Computer generated record

The default values for the fields are mostly fixed, as listed in Table 23-2.

Table 23-2 Some default values used in the example

Field	Type	Default value
CUST-NM	AN	FX 'TEST ADT' repeat
KEYREC-DOT	AN	FX ‘x’
COMP-NM	AN	RO 'aaabbbcccdddeeefff'
DEC-NO-SHARES	PD	Init=10, Incr=7
BUY-FROM	AN	FX ‘Unknown’
BUY-FROM-NO	AN	RA '0123456789'

23.1.9 How to split a single file into constituent record types

There may be times when you need to take one or more of the record types in a multi-record file and segregate the records for additional processing. The recommended way to do this is by using a DFSORT procedure. We show batch job streams, using both DFSORT and REXX procedures, that take a file and split it into three record types. All other record types are ignored.

The required DFSORT procedure would be similar to that shown in Example 23-8.

Example 23-8 DFSORT procedure

```
//*
//FILEMGR  EXEC PGM=FMNMAIN
//STEPLIB  DD DSN=FMN.V4R1M0.SFMNMOD1,DISP=SHR
//*        DD DSN=IGY.SIGYCOMP,DISP=SHR
```

```
//REC01    DD DSN=REDBK2.TEST.SEQFILEA,DISP=OLD
//REC02    DD DSN=REDBK2.TEST.SEQFILEB,DISP=OLD
//REC03    DD DSN=REDBK2.TEST.SEQFILEC,DISP=OLD
//SYSPRINT DD SYSOUT=*
//FMNTSPRT DD SYSOUT=*
//SYSTERM  DD SYSOUT=*
//SYSIN    DD *
$$FILEM DSC DSNIN=REDBK2.TEST.SEQFILE,
$$FILEM POSITION=0,
$$FILEM DISP=MOD,
$$FILEM DSNOUT=REDBK2.TEST.SEQFILEX,
$$FILEM PROC=*
*FASTPROC
 OUTFIL FNAMES=REC01,INCLUDE=(1,2,CH,EQ,C'01')
 OUTFIL FNAMES=REC02,INCLUDE=(1,2,CH,EQ,C'02')
 OUTFIL FNAMES=REC03,INCLUDE=(1,2,CH,EQ,C'03')
/+
```

The same requirement can be met by using an in-stream REXX routine to process the records. The File Manager step of the batch job, shown in Example 23-9, illustrates this.

Example 23-9 File Manager step of batch job

```
//*
//FILEMGR  EXEC PGM=FMNMAIN
//STEPLIB  DD DSN=FMN.V4R1M0.SFMNMOD1,DISP=SHR
//*        DD DSN=IGY.SIGYCOMP,DISP=SHR
//REC01    DD DSN=REDBK2.TEST.SEQFILEA,DISP=OLD
//REC02    DD DSN=REDBK2.TEST.SEQFILEB,DISP=OLD
//REC03    DD DSN=REDBK2.TEST.SEQFILEC,DISP=OLD
//SYSPRINT DD SYSOUT=*
//FMNTSPRT DD SYSOUT=*
//SYSTERM  DD SYSOUT=*
//SYSIN    DD *
$$FILEM DSC DSNIN=REDBK2.TEST.SEQFILE,
$$FILEM POSITION=0,
$$FILEM DISP=MOD,
$$FILEM DSNOUT=REDBK2.TEST.SEQFILEX,
$$FILEM PROC=*
DDNAME = 'REC' || FLD(1,2)
IF NCO(FLD(1,2),1,2,3) THEN DO
   WRITE(DDNAME)
   RETURN 'DROP'
END
/+
```

What is happening in these jobs

The file SEQFILE is the input file that contains multiple record types. The default output file has the DD name SEQFILEX. Each of the record types we are interested in go into REC01, REC02, or REC03.

The File Manager program keyword DSC is used to invoke the Data Set Copy function. The input and output files are identified, and the keyword PROC=* is used to indicate that an in-stream routine is being supplied.

The result is that all type 01 records end up in REC01, type 02 records go to REC02, type 03 records go to REC03, and all other record types go to the file EXTRA.

Reviewing the report output

The key portion of the batch job's output report is listed in Example 23-10.

> **Note:** Each page in the report starts with the title, "IBM File Manager for z/OS and OS/390."

Example 23-10 Report of DSC multiple record split

```
IBM File Manager for z/OS and OS/390
DSC      WRITE summary report
--------------------------------------------------------------------------------
Total records written to REC01    = 20
Total records written to REC02    = 20
Total records written to REC03    = 15
IBM File Manager for z/OS and OS/390
67 record(s) read
12 record(s) copied: 0 truncated: 0 fields truncated
```

The first part contains the output of the record split operation (a copy). Notice that you do not have to do any extra programming to obtain the number of records sent to each file; File Manager does that automatically.

The last line contains the total number of records processed. In this case, 12 records did not meet any of the selection criteria, and were written to the default file (EXTRA).

File Manager external REXX functions used in this routine

A brief explanation of each of the File Manager external REXX functions that were used in this routine follows.

FLD Returns the value of a field from the current input record (INREC), starting at start_column, of length number_of_bytes, interpreted according to the specified type:

B: If the field is binary. If you specify B for the type, the length must be 2, 4, or 8.

If the field contains characters, the following conventions must be used:

C: If the field is packed decimal. This is the default.

P: If you specify P for type, length must be between 1 and 16 and the field is zoned decimal.

Z: If you specify Z for type, length must be between 1 and 32, or if the field contains a separate sign character, between 1 and 33.

NCO If the numeric value of any of the match arguments is equal to the numeric value of number, then NCONTAIN returns 1. Otherwise, NCONTAIN returns 0.

WRITE Writes a record to the specified data sets. If the WRITE function is successful, it returns a value of 0. If the WRITE function is unsuccessful, it raises the REXX syntax error condition.

RETURN In REXX, you can use the RETURN or EXIT instruction to leave a procedure. `RETURN` is preferred for performance reasons. You can optionally specify a character string as a parameter on the instruction. This character string is returned to the caller of the procedure.

DROP The character string DROP tells File Manager to not write the current record to the primary output data set.

23.1.10 Replace a string in a specific location in a file

If you need to unconditionally replace a string in one location of a file, you can use this utility.

The code to perform this function with File Manager is shown in Example 23-11.

Example 23-11 File Manager string replace batch step

```
//*
//* FILE MANAGER BATCH: REPLACE A STRING IN A SPECIFIC LOCATION
//*
//STEP01   EXEC PGM=FMNMAIN
//STEPLIB  DD DSN=FMN.V4R1M0.SFMNMOD1,DISP=SHR
//*        DD DSN=IGY.SIGYCOMP,DISP=SHR
//SYSPRINT DD  SYSOUT=*
//FMNTSPRT DD  SYSOUT=*
//SYSTERM  DD  SYSOUT=*
//EDITFILE DD  DISP=OLD,DSN=YOUR.FILE.TO.EDIT
//SYSIN    DD  *
$$FILEM DSU INPUT=EDITFILE,
$$FILEM     PROC=*
OUTREC=OVERLAY(‘vaLUE’,INREC,11)
/+
```

What is happening in this step

The File Manager program keyword `DSU` invokes the Data Set Update utility, which is only available in batch.

The utility reads records sequentially from the input file. When File Manager processes them, it uses two built-in REXX variables, INREC and OUTREC to refer to the input and output records.

In this case, we use a standard REXX function, OVERLAY, to indicate a string should be placed at a specific location. This is assigned to the output record that is written to the file, a portion of which is shown in Example 23-12.

Example 23-12 Output from string replace batch job

```
IBM File Manager for z/OS and OS/390
$$FILEM DSU DSNIN=DAVIN6.FILE.TO.EDIT,
$$FILEM     PROC=*
13 record(s) read
13 record(s) updated
```

23.1.11 Copy selected variably blocked records to another file

If you need to copy selected records from a production file to a test file, you can use this utility. In this case there are multiple criteria and there is only one output file. This example also demonstrates how File Manager processes new file allocation and variable blocked records.

The steps necessary to perform this function using REXX with File Manager follow.

1. Select option **3.3** from the main FM menu, producing a panel as shown in Figure 23-21.

```
 Process   Options   Help
File Manager                        Copy Utility
Command ===> _

From Partitioned, Sequential or VSAM Data Set:
   Data set name . . . . . 'REDBK2.TEST.SEQFILE'
   Member  . . . . . . . . ________       (Blank or pattern for member list)
   Volume serial . . . . . ______
   Start key . . . . . . . ________________________________ key or slot
   Skip count  . . . . . . 0_______      number of records to be skipped
   Copy count  . . . . . . ALL_____      number of records to be copied
From Copybook or Template:
   Data set name . . . . . ____________________________________________
   Member  . . . . . . . . ___________      (Blank or pattern for member)
Processing Options:
   Copybook/template usage     Enter "/" to select option
   2  1. Above                 _  Edit template
      2. None                  /  Use proc *_______
      3. Create dynamic        _  Ignore length mismatch
                               /  Batch Execution

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 23-21 Option selection from main File Manager panel

2. Select **Use proc** with the generic * and press Enter. You can now enter record selection criteria for the copy. In this example we are selecting only those records with either a C, G, L, M, or S in column 3, as shown in Figure 23-22.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       SYS03296.T105320.RA000.REDBK2.R0100810          Columns 00001 00072
Command ===> ____________________________________________ Scroll ===> CSR
Enter your REXX procedure statements
==MSG> -Warning- The UNDO command is not available until you change
==MSG>           your edit profile using the command RECOVERY ON.
''''''  *FASTPROC
''''''   INCLUDE COND=(3,1,SS,EQ,C'C,G,L,M,S')
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 23-22 Record selection criteria

3. Press Enter, followed by PF3. You can now enter the name of the file to which the data is to be copied. In this example, we are dynamically defining a new file (SEQFILEX) with the same characteristics as the input file. These steps are shown in Figure 23-23 to Figure 23-25.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
Copy from REDBK2.TEST.SEQFILE
Command ===> ____________________________________________________________________

To Partitioned, Sequential or VSAM Data Set:
   Data set name . . . . . 'REDBK2.TEST.SEQFILEX'____________________________
   Member  . . . . . . . . ________
   Volume serial . . . . . ______

Processing Options:
   Disposition              Execution "/" options
   2  1. Old or Reuse       _  Replace members
      2. Mod

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 23-23 Name of file to be copied to (Part 1 of 3)

```
 Process   Options   Help
 ------------------------------------------------------------------------------
Allocate REDBK2.TEST.SEQFILEX
Command ===> ____________________________________________________________________

New Data Set Organization:
   Select option          Instructions
   6 1. KSDS              The above data set does not exist.
     2. ESDS              To define or allocate a new data set select a data
     3. RRDS              set organization and press ENTER or press PF3/EXIT
     4. VRRDS             or PF12/CANCEL to return without allocation.
     5. LDS
     6. Non VSAM ------> For a new non VSAM data set only, enter a data set
                          name below to copy existing allocation attributes.

Existing Partitioned or Sequential Data Set:
   Data set name . . . . . 'REDBK2.TEST.SEQFILE'____________________________
   Volume serial . . . . . ______

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 23-24 Name of file to be copied to (Part 2 of 3)

```
 Process   Options   Help
 ------------------------------------------------------------------------------
Allocate REDBK2.TEST.SEQFILEX
Command ===> _____________________________________________________________________
                                                                        More:     +
Specify a model data set, SMS class names, or leave blank for defaults:
   Like data set . . _____________________________________________
   Volume serial . . STF203
   Data class  . . . ________      leave blank for default
   Storage class . . STF2CLS1      leave blank for default
   Management class  ________      leave blank for default

Space Requirements:
   Space unit  . . . TRKS          BLKS, TRKS, CYLS, KB, or MB
   Primary units . . 4             quantity of above units
   Secondary units . 1             quantity of above units
   Directory blocks  0             leave blank for SMS default
   Record format . . FB            if new format: U,F,V, or D, with B,S,A,M
   Record length . . 90
   Block size  . . . 27990         physical output block size
   Library type  . . ________      LIBRARY, PDS, or blank for default
   Release unused  . NO            enter YES to free unused space, else NO
 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 23-25 Name of file to be copied to (Part 3 of 3)

4. Since we selected Batch execution for this Copy job, we are presented with the JCL as shown in Example 23-13.

Example 23-13 JCL for batch

```
//FILEMGR  EXEC PGM=FMNMAIN
//STEPLIB  DD DSN=FMN.V4R1M0.SFMNMOD1,DISP=SHR
//*        DD DSN=IGY.SIGYCOMP,DISP=SHR
//SYSPRINT DD SYSOUT=*
//FMNTSPRT DD SYSOUT=*
//SYSTERM  DD SYSOUT=*
//SYSIN    DD *
$$FILEM DSC DSNIN=REDBK2.TEST.SEQFILE,
$$FILEM POSITION=0,
$$FILEM DISP=MOD,
$$FILEM DSNOUT=REDBK2.TEST.SEQFILEX,
$$FILEM PROC=*
*FASTPROC
 INCLUDE COND=(3,1,SS,EQ,C'C,G,L,M,S')
/+
```

5. Submitting this job, we see the successful execution report as shown in Example 23-14.

Example 23-14 Successful completion

```
IBM File Manager for z/OS
$$FILEM DSC DSNIN=REDBK2.TEST.SEQFILE,
$$FILEM POSITION=0,
$$FILEM DISP=MOD,
$$FILEM DSNOUT=REDBK2.TEST.SEQFILEX,
$$FILEM PROC=*
67 record(s) read
20 record(s) successfully copied
```

6. Alternatively, we could have used a REXX procedure to obtain the same result. For this method, we coded the procedure as shown in Figure 23-26. Remember, though, that the

performance when processing large data sets is significantly better if DFSORT can be used.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       SYS03296.T111511.RA000.REDBK2.R0100812          Columns 00001 00072
Command ===> ____________________________________________ Scroll ===> CSR
Enter your REXX procedure statements
==MSG> -Warning- The UNDO command is not available until you change
==MSG>           your edit profile using the command RECOVERY ON.
'''''' IF CO(FLD(3,1),C,G,L,M,S) THEN
''''''    EXIT
'''''' ELSE
''''''    EXIT 'DROP'
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
''''''
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 23-26 REXX procedure

7. Following this through, we see JCL created as shown in Example 23-15.

Example 23-15 JCL created

```
//FILEMGR  EXEC PGM=FMNMAIN
//STEPLIB  DD DSN=FMN.V4R1M0.SFMNMOD1,DISP=SHR
//*        DD DSN=IGY.SIGYCOMP,DISP=SHR
//SYSPRINT DD SYSOUT=*
//FMNTSPRT DD SYSOUT=*
//SYSTERM  DD SYSOUT=*
//SYSIN    DD *
$$FILEM DSC DSNIN=REDBK2.TEST.SEQFILE,
$$FILEM POSITION=0,
$$FILEM DISP=MOD,
$$FILEM DSNOUT=REDBK2.TEST.SEQFILEX,
$$FILEM PROC=*
IF CO(FLD(3,1),C,G,L,M,S) THEN
   RETURN
ELSE
   RETURN 'DROP'
/+
```

Successful completion of the job is shown in Example 23-16, with identical results. Note that for a job such as this, particularly when working with a large file, DFSORT performance will be significantly better than that of REXX.

Example 23-16 Successful completion

```
IBM File Manager for z/OS
$$FILEM DSC DSNIN=REDBK2.TEST.SEQFILE,
$$FILEM POSITION=0,
$$FILEM DISP=MOD,
```

```
$$FILEM DSNOUT=REDBK2.TEST.SEQFILEX,
$$FILEM PROC=*
67 record(s) read
20 record(s) copied: 0 truncated: 0 fields truncated
```

What is happening in these jobs

The File Manager program keyword DSC is used to invoke the data set copy function. The input and output files are identified, and the keyword PROC is used to indicate that an in-stream routine, either DFSORT or REXX, is being supplied.

DFSORT We know that DFSORT is being used by the presence of the *FASTPROC statement. The routine is built using standard DFSORT program control statements, as described in the *DFSORT R14 Application Programming Guide* SC26-7050-05. The INCLUDE statement uses a substring comparison test to search for one of the listed values in column 3 of the input records. If found, the record is copied to the output file; all other records are bypassed.

REXX The first line of the routine checks the contents of the record (using the function FLD) starting in position 3 for a length of 1 in order to see if it matches one of the listed values. If it does, the second line of the routine writes out the records to the output file. Otherwise, the fourth line ignores the records.

Notice that in both examples the Data Control Block (DCB) information for the new file is copied from the input file.

23.1.12 Search for a string in all members of a PDS

If you need to determine which members of a PDS contain a particular string, you can use this utility.

The code to perform this function with File Manager is shown in Example 23-17.

Example 23-17 File Manager string find in a PDS batch step

```
//*
//* FILE MANAGER BATCH: SEARCH FOR STRING
//*
//STEP01   EXEC PGM=FMNMAIN
//STEPLIB  DD DSN=FMN.V4R1M0.SFMNMOD1,DISP=SHR
//*        DD DSN=IGY.SIGYCOMP,DISP=SHR
//SYSPRINT DD  SYSOUT=*
//FMNTSPRT DD  SYSOUT=*
//SYSTERM  DD  SYSOUT=*
//DDIN     DD  DISP=SHR,DSN=YOUR.CHANGE.MGMT.UAT.JCLLIB
//SYSIN    DD  *
$$FILEM FCH INPUT=DDIN,MEMBER=*,
$$FILEM     PROC=*
   IF CO(INREC,'UNIT=CART') | ,
      CO(INREC,'UNIT=TAPE') THEN
      RETURN
   ELSE
      RETURN 'DROP'
/+
```

What is happening in this step

The File Manager utility FCH is used to invoke the Find/Change function.

The file, DDIN (the default input file for the FCH function) is the PDS you want to search.

The first two lines check for one of two strings. The fifth line ignores any records that do not contain the strings.

Reviewing the report output

The key portion of the batch job's output report is listed in Example 23-18.

Example 23-18 Output from string find in a PDS batch job

```
IBM File Manager for z/OS and OS/390
$$FILEM FCH INPUT=DDIN,MEMBER=*,
$$FILEM     PROC=*
IBM File Manager for z/OS and OS/390
Record-#  Find/Change Listing DSN:DAVIN6.WORK.JCL

 FABSERCH                    ---------- STRING(S) FOUND ----------

     12s             IF=(1,0,C'UNIT=CART'),
     13s           ORIF=(1,0,C'UNIT=TAPE')

 FMBSERCH                    ---------- STRING(S) FOUND ----------

     14s    IF CO(INREC,'UNIT=CART') | ,
     15s       CO(INREC,'UNIT=TAPE') THEN DO

 IECD01                      ---------- STRING(S) FOUND ----------

    833s //            DISP=(,CATLG,DELETE),UNIT=CART,EXPDT=99000,
    862s //            DISP=(,CATLG,DELETE),UNIT=CART,EXPDT=99000,

 ISBSERCH                    ---------- STRING(S) FOUND ----------

      9s SRCHFOR  'UNIT=CART'
     10s SRCHFOR  'UNIT=TAPE'

 TESTME                      ----------- member in use -----------

  ------- Find/Change summary section ------------------
  Records found: 8 Records processed: 2744
  Members w/recs: 5 Members wo/recs: 42

  ------- Find/Change statement section ----------------

  IF CO(INREC,'UNIT=CART') | ,
  CO(INREC,'UNIT=TAPE') THEN
  RETURN
  ELSE
  RETURN 'DROP'
```

Each of the members in which either one of the strings was found is listed. The lines on which the strings were found are displayed.

Notice that our test file is still in use; no search was performed on this member (otherwise, the string would have been found there as well).

The summary statistics appear at the end of the report, along with a display of the search commands.

23.1.13 Multiple find

The **FIND utility** command (option **3.6**) allows you to specify more than one string to search for, optionally with a limit on the range of columns to be searched. The command can be entered as shown in Example 23-19.

Example 23-19 FIND utility command

```
FIND string1 AND string2 OR string3 col1 col2
F string1 & string2 | string3 col1 col2
```

Note: This does not apply to the FIND command in Edit/Browse mode.

This is a free-form FIND. If you do not specify a column range, the total length of the record is used. There are some limitations to this utility; use REXX procedures for more complex processing such as:

- Searching for a string in one field AND/OR another string in another field.
- Combining FIND and CHANGE in the same command.

Restriction: You do not have the option of specifying a template and field reference instead of the starting and ending columns.

When you request the online FIND, you get the results in the SRCHFOR.LIST file specified on the panel, as shown in Example 23-20.

Example 23-20 Result of combined FIND '01' | '03' 1 2

```
********************************** Top of Data **********************************
IBM File Manager for z/OS and OS/390
Record-#  Find/Change Listing DSN:DAVINR1.DEMOS.TESTFILE

 FMNCDATA                  ---------- STRING(S) FOUND ----------

     1  01Grant Sutherland    ...-.Í.........................................
     2  01Andrew Astle        .M...Í.........................................
     3  01Graham Purdie       .h.;.Ç.........................................
     4  01Bill Soper          .....Ç.........................................
     5  01Tyrone Dalais       .O...-.........................................
     6  01Rod Turner          .Ö.Ã...........................................
     7  01Clive Nealon        .u.Ã...........................................
     8  01Jim Alexander       .ì.>...........................................
     9  01Silvano Prez        .....-.........................................
    10  01Don Pharoah         .....Ä.........................................
    11  01John Levrington     .è...Å.........................................
    12  01Liz Rushton         .è...Ã.........................................
    13  01Bob McCormack       .µ...Î.........................................
    14  01Keith Stewart       .....ø.........................................
    15  01John Nicholls       .è...Ñ.........................................
    16  01Peter van Dyke      .µ...e.........................................
```

```
    17  01Anna Waghorn        .d...-.........................................
    18  01Mike Moriarty       .....d.........................................
    19  01Merrill Bani        ...-.ì.........................................
    20  01Angus King          ...-...........................................

  ------- Find/Change summary section -----------------
  Records found: 20 Records processed: 40
  Members w/recs: 1 Members wo/recs: 0
  Search cols: 1:2 Longest line: 80

  Edit options in effect: CAPS:OFF

  Search argument: 01
  Search argument: 03
******************************** Bottom of Data ********************************
```

The FIND corresponding procedure is shown in Example 23-21.

Example 23-21 FIND procedure

```
$$FILEM FCH ,
$$FILEM MEMBER=*,
$$FILEM DSNIN=DAVINR1.DEMOS.TESTFILE
BNDS 1 2
CAPS OFF
find '01' | '03' 1 2
/+
```

Result of a batch FIND command

You can see the result of the FIND command with SDSF. The new command is presented in Figure 23-27.

```
 SDSF OUTPUT DISPLAY DAVINR1M JOB05150  DSID   102 LINE 0       COLS 02- 81
 COMMAND INPUT ===> _                                         SCROLL ===> CSR
********************************* TOP OF DATA *********************************
IBM File Manager for z/OS and OS/390


 DEFAULT SET FUNCTION VALUES:

PRINTOUT=SYSPRINT  HEADERPG=YES  ASCII=BOTH     RECLIMIT=(1,*)
PRINTLEN=132       PAGESKIP=NO   PAD=OFF        EOD=/*
PAGESIZE=60        DATAHDR=YES   DBCSPRT=OFF    LANGUAGE=ENGLISH
PRTTRANS=ON        DUMP=UPDOWN   TAPELBL=SL     CYLHD=ABSOLUTE
SMFNO=000

IBM File Manager for z/OS and OS/390
$$FILEM FCH ,
$$FILEM MEMBER=*,
$$FILEM DSNIN=DAVINR1.DEMOS.TESTFILE

 FMNCDATA                    ---------- STRING(S) FOUND ----------
```

Figure 23-27 SDSF output of batch FIND command

You can specify an output file in the SYSPRINT DD card when you request the batch execution of FIND.

23.1.14 QSAM allocation using a model

In many utilities, when you specify a QSAM output file that does not exist, File Manager helps you fill in the information required for the allocation. A new panel for new data set allocation is displayed, as shown in Figure 23-28. You can specify an existing data set from which to take the allocation specifications.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 Allocate REDBK2.TEST.SEQFILEK
 Command ===> ____________________________________________________________________

 New Data Set Organization:
    Select option         Instructions
    6 1. KSDS             The above data set does not exist.
      2. ESDS             To define or allocate a new data set select a data
      3. RRDS             set organization and press ENTER or press PF3/EXIT
      4. VRRDS            or PF12/CANCEL to return without allocation.
      5. LDS
      6. Non VSAM ------> For a new non VSAM data set only, enter a data set
                          name below to copy existing allocation attributes.


 Existing Partitioned or Sequential Data Set:
    Data set name . . . . . 'REDBK2.TEST.SEQFILE'
    Volume serial . . . . . ______



  F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
  F9=Swap     F10=Actions  F12=Cancel
```

Figure 23-28 Allocate a new data set

The allocation panel is filled with the values taken from this existing data set, as shown in Figure 23-29.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 Allocate REDBK2.TEST.SEQFILEK                          Command is not active
 Command ===> ____________________________________________________________________
                                                                   More:     +
 Specify a model data set, SMS class names, or leave blank for defaults:
    Like data set . . ____________________________________________
    Volume serial . . ______
    Data class  . . . ________     leave blank for default
    Storage class . . ________     leave blank for default
    Management class  ________     leave blank for default

 Space Requirements:
    Space unit  . . . TRKS         BLKS, TRKS, CYLS, KB, or MB
    Primary units . . 4____        quantity of above units
    Secondary units . 1____        quantity of above units
    Directory blocks  0____        leave blank for SMS default
    Record format . . FB__         if new format: U,F,V, or D, with B,S,A,M
    Record length . . 90___
    Block size  . . . 27990        physical output block size
    Library type  . . _______      LIBRARY, PDS, or blank for default
    Release unused  . NO__         enter YES to free unused space, else NO
  F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
  F9=Swap     F10=Actions  F12=Cancel
```

Figure 23-29 Allocation based on existing data set

As proof, the characteristics of the source data set are shown in Figure 23-30. These were obtained from ISPF option **3.2**.

```
 Process   Options   Help
File Manager                  Utility Functions
                              Data Set Information
Command ===> _
                                                                More:     +
Data Set Name . . . . : REDBK2.TEST.SEQFILE

General Data                            Current Allocation
 Management class . . : **None**         Allocated tracks  . : 4
 Storage class  . . . : STF2CLS1         Allocated extents . : 1
  Volume serial . . . : STF203
  Device type . . . . : 3390
 Data class . . . . . : **None**        Current Utilization
  Organization  . . . : PS               Used tracks . . . . : 1
  Record format . . . : FB               Used extents  . . . : 1
  Record length . . . : 90
  Block size  . . . . : 27990
  1st extent tracks . : 4
  Secondary tracks  . : 1
  Data set name type  :                  SMS Compressible  . : NO
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 23-30 Characteristics of the existing data set used as source

23.1.15 Work with VTOC

Follow these steps to work with VTOC:

1. Select option **3** (Utilities, on the FM main menu), then option **5** (VTOC). This takes you to the Display VTOC panel. From here you can display the volumes (option V), as shown in Figure 23-31.

```
 Process   Options   Help
File Manager              Volume Selection/Summary
Command ===> _                                                  Scroll CSR
    <----- Volume Data --------> <-VTOC Data-> <-------- Free Space --------->
Sel Volser Devno Unit Stat %Used Size %Used Ix DSCBs  Cyls   Max Tracks    Max
_   STF203  0383 3390 SMS     32   15    17  Y   620  2270  2269  34079  34035
_   STF204  0384 3390 SMS     14   15    12  Y   658  2864  2862  42998  42930
_   STF206  0386 3390 SMS     17   15    11  Y   667  2759  2715  41441  40729
_   STF208  0388 3390 SMS     49   15     9  Y   681  1712  1710  25728  25660
_   STF209  0389 3390 SMS     23   15    10  Y   676  2580  2552  38750  38286
_   STF20A  038A 3390 SMS     33   15    10  Y   677  2240  2222  33644  33330
_   STF20C  038C 3390 SMS     50   15    10  Y   675  1683  1683  25284  25259
_   STF20D  038D 3390 SMS     26   15     8  Y   687  2483  2457  37275  36855
_   STF20F  038F 3390 SMS     18   15    16  Y   633  2724  2724  40896  40860
_   STF210  0390 3390 SMS     37   15    15  Y   640  2100  2100  31508  31500
_   STF211  0391 3390 SMS      8   15     9  Y   685  3071  3060  46108  45908
   ****  End of data  ****

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Up        F8=Down
 F9=Swap     F12=Cancel
```

Figure 23-31 List of volumes for using VTOC

2. You can select one volume here to view the list of the files it contains. If you want the list for all volumes, go back to the Display VTOC panel and enter a generic name in the Data Set Name field as shown in Figure 23-32.

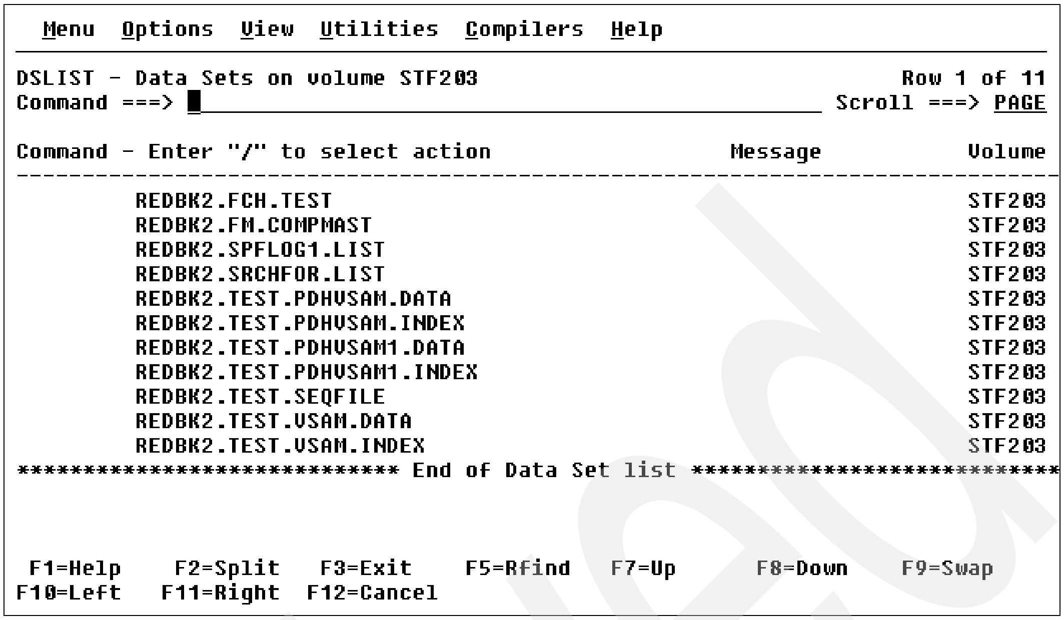
```
 Process   Options   Help
 ------------------------------------------------------------------------------
File Manager                      Display VTOC                      Line 1 of 11
Command ===> _____________________________________________________ Scroll CSR
Unit 0383 VOLSER STF203  3390 with 3339 cyls, 15 trks/cyl, 58786 bytes/trk

        - Data Set Name sorted by NAME  -- Ext              Begin-end
Command 1...5...10...15...20...25...30..... seq Volume   Cyl-hd   Cyl-hd Tracks
        REDBK2.FCH.TEST                       0 STF203   882  1   882  1      1
        REDBK2.FM.COMPMAST                    0 STF203  1020 13  1020 14      2
        REDBK2.SPFLOG1.LIST                   0 STF203  1065  0  1065  7      8
        REDBK2.SRCHFOR.LIST                   0 STF203   873  0   873 14     15
        REDBK2.TEST.PDHVSAM.DATA              0 STF203   882  2   882  2      1
        REDBK2.TEST.PDHVSAM.INDEX             0 STF203   882  3   882  3      1
        REDBK2.TEST.PDHVSAM1.DATA             0 STF203   882 14   882 14      1
        REDBK2.TEST.PDHVSAM1.INDEX            0 STF203  1038 14  1038 14      1
        REDBK2.TEST.SEQFILE                   0 STF203  1045  6  1045  9      4
        REDBK2.TEST.VSAM.DATA                 0 STF203   846 13   846 13      1
        REDBK2.TEST.VSAM.INDEX                0 STF203   846 14   846 14      1
******************************** End of data ********************************


 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F5=RFind     F6=Process
 F7=Up        F8=Down      F9=Swap     F10=Actions  F11=RgtLeft  F12=Cancel
```

*Figure 23-32 List of * files for STF203 using VTOC*

Note the differences in the information given by this data set list display and that supplied by File Manager DSLIST (option **3.4**) or the ISPF DSLIST (option **3.4**), as shown in Figure 23-33 and Figure 23-34.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
File Manager                      Data Set List                     Line 1 of 37
Command ===> _____________________________________________________ Scroll CSR
Catalog ID ''                                                    Types ALL
          --- Data Set Name --- sorted by NAME    ---   Entry  Prim  M Created
Command  1...5...10...15...20...25...30...35...40....   type  volume V YYYY.DDD
         REDBK2                                         ALIAS
         REDBK2.DMON.SYSIN                              NVSAM STF20F   2003.295
         REDBK2.DMON.SYSPRINT                           NVSAM STF20F   2003.295
         REDBK2.DSNSPFT                                 NVSAM STF211   2003.276
         REDBK2.FCH.TEST                                NVSAM STF203   2003.288
         REDBK2.FM.CNTL                                 NVSAM STF20F   2003.283
         REDBK2.FM.COMPMAST                             NVSAM STF203   2003.295
         REDBK2.FMN.LIST                                NVSAM STF20F   2003.294
         REDBK2.ISPPROF                                 NVSAM STF210   2003.272
         REDBK2.SPFLOG1.LIST                            NVSAM STF203   2003.296
         REDBK2.SPF100.OUTLIST                          NVSAM STF20A   2003.283
         REDBK2.SRCHFOR.LIST                            NVSAM STF203   2003.283
         REDBK2.TEST.COMPMAST                           NVSAM STF208   2003.295
         REDBK2.TEST.PDHVSAM                            KSDS           2003.290
           REDBK2.TEST.PDHVSAM.DATA                     DATA  STF203   2003.290
 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F5=RFind     F6=Process
 F7=Up        F8=Down      F9=Swap     F10=Actions  F11=RgtLeft  F12=Cancel
```

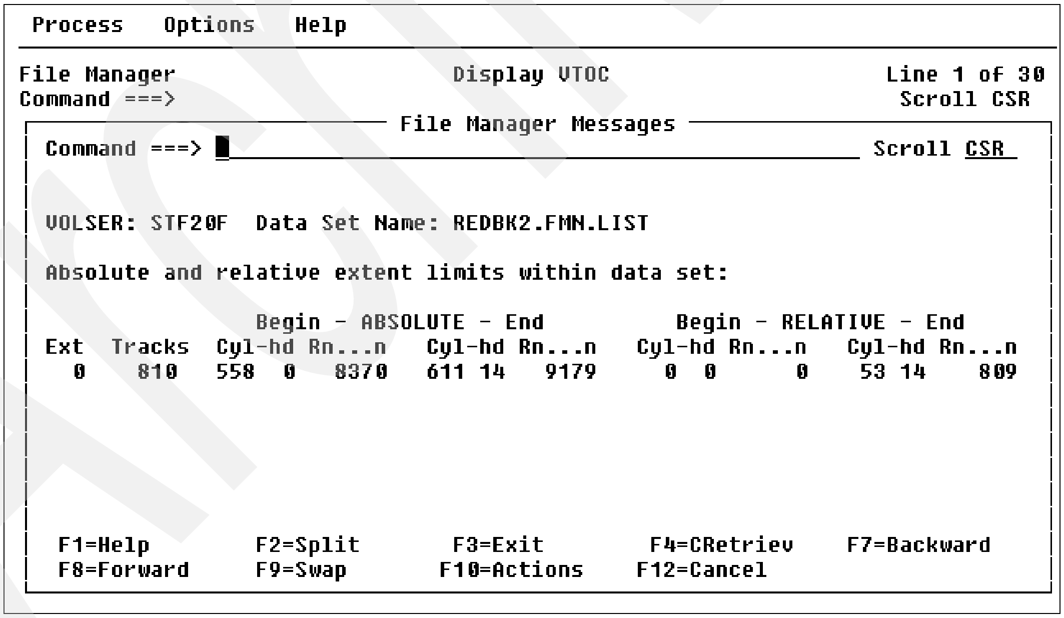
Figure 23-33 File Manager DSLIST



```
  Menu  Options  View  Utilities  Compilers  Help
------------------------------------------------------------------------------
DSLIST - Data Sets on volume STF203                                 Row 1 of 11
Command ===> _______________________________________________ Scroll ===> PAGE

Command - Enter "/" to select action                  Message          Volume
-------------------------------------------------------------------------------
         REDBK2.FCH.TEST                                               STF203
         REDBK2.FM.COMPMAST                                            STF203
         REDBK2.SPFLOG1.LIST                                           STF203
         REDBK2.SRCHFOR.LIST                                           STF203
         REDBK2.TEST.PDHVSAM.DATA                                      STF203
         REDBK2.TEST.PDHVSAM.INDEX                                     STF203
         REDBK2.TEST.PDHVSAM1.DATA                                     STF203
         REDBK2.TEST.PDHVSAM1.INDEX                                    STF203
         REDBK2.TEST.SEQFILE                                           STF203
         REDBK2.TEST.VSAM.DATA                                         STF203
         REDBK2.TEST.VSAM.INDEX                                        STF203
***************************** End of Data Set list ****************************



 F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
F10=Left   F11=Right  F12=Cancel
```

Figure 23-34 ISPF DSLIST

3. Turning attention back to the File Manager Display VTOC file list, you can enter commands against a specific file. To get the list of the available commands, refer to the on-line help. Figure 23-35 illustrates the result of issuing the EXTENTS command against file REDBK2.FMN.LIST.



```
 Process   Options   Help
------------------------------------------------------------------------------
 File Manager                    Display VTOC                      Line 1 of 30
 Command ===>                                                        Scroll CSR
  ----------------------------- File Manager Messages ---------------------------
 | Command ===> ________________________________________________ Scroll CSR_  |
 |                                                                            |
 |                                                                            |
 | VOLSER: STF20F  Data Set Name: REDBK2.FMN.LIST                             |
 |                                                                            |
 | Absolute and relative extent limits within data set:                       |
 |                                                                            |
 |                Begin - ABSOLUTE - End          Begin - RELATIVE - End      |
 | Ext   Tracks  Cyl-hd Rn...n   Cyl-hd Rn...n   Cyl-hd Rn...n   Cyl-hd Rn...n |
 |   0     810   558  0   8370   611 14   9179     0  0      0     53 14    809 |
 |                                                                            |
 |                                                                            |
 |                                                                            |
 |                                                                            |
 |                                                                            |
 |  F1=Help       F2=Split       F3=Exit       F4=CRetriev    F7=Backward     |
 |  F8=Forward    F9=Swap       F10=Actions   F12=Cancel                      |
  ----------------------------------------------------------------------------
```

Figure 23-35 Results of EXTENTS command

23.1.16 Get information about a load module

It may be useful to get some information about an existing load module. Use option **3** (Utilities) then option **10** (Loadlib) to get to the panel shown in Figure 23-36.

```
 Process   Options   Help
 ------------------------------------------------------------------------
File Manager                Load Module Information
Command ===> ___________________________________________________________________

Input:
   Data set name . . . . . 'FMN.V4R1M0.SFMNMOD1'
   Member  . . . . . . . . ________      Blank or pattern for member list
   Volume serial . . . . . ______        If not cataloged

Processing Options:
   Order CSECTs by                        Output to
   1   1. Address                         1   1. Display
       2. Name                                2. Printer



 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 23-36 Utilities and Loadlib options selected

You can specify the module name in this panel or press Enter and select the module you want from the resultant list.

For the module selected, you get the components, their size, and the short name (if available) or the program number of the compiler used, as shown in Figure 23-37.

```
 Process   Options   Help
 ------------------------------------------------------------------------
File Manager                Load Module Information                Line 1 of 8
Command ===> _____________________________________________________ Scroll PAGE

 Load Library - 'FMN.V4R1M0.SFMNMOD1'
 Load Module -  FMNMOD16     Linked 2003.260   by 5695PMB01

 Name      Type Address Size    A/RMODE Compiler 1 Date 1   Compiler 2 Date 2
 --------- ---- ------- ------- ------- ---------- -------- ---------- --------
 FMN$$CPR   SD   000000   01E8   31/ANY PL/X-390   2003.237 569623400  2003.237
  FMN$$CPR   LD  000000
 FMNBELOW   SD   0001E8   1908   31/ 24 PL/X-390   2003.238 569623400  2003.238
  FMNBELOW   LD  0001E8
 FMNAMSIO   SD   001AF0   2A58   31/ 24 PL/X-390   2003.237 569623400  2003.237
  FMNAMSIO   LD  001AF0
  FMNAMSI    LD  001B0A
  FMNAMSO    LD  001B10
 ********************************* End of data *********************************



 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 23-37 Load module information

23.1.17 Dynamic template

When a COBOL or PL/I layout is not available, you can use the unformatted template. It allows you to define a record structure you can use to map the data. The following steps describe how to use the template to format your data.

1. Select option **3** (None) on the browse or edit panel. You are presented with unformatted data as shown in Figure 23-38.

```
 Process   Options   Help
File Manager Browse REDBK3.REDBOOK.PDPAK.COMPMAST
Command ===> _                                                Scroll CSR
                                  Record 0        Col 1       Format CHAR
----+----1----+----2----+----3----+----4----+----5----+----6----+----7----+---
****  Top of data  ****
Casey_Import_Export 00079.0000077.0000078.0000072.0000070.0000065.0000063.00000
Glass_and_Luget_plc 00019.0000022.0000025.0000020.0000016.0000020.0000022.00000
Headworth_Electrical00124.0000131.0000133.0000133.0000133.0000137.0000138.00001
IBM                 00163.0000163.0000162.0000160.0000161.0000159.0000156.00001
ShareSelect         00119.0000120.0000118.0000116.0000116.0000112.0000110.00001
SportSelect         00224.0000224.0000220.0000217.0000215.0000215.0000215.00002
Veck_Transport      00036.0000034.0000034.0000035.0000037.0000039.0000042.00000
****  End of data  ****

 F1=Help      F2=Zoom      F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 23-38 Result of an unformatted browse

2. Select option **4** (Create dynamic) on the browse or edit panel. You are presented with the panel shown in Figure 23-39, where you can specify the characteristics of the record layout. To add field names, use PF11 from this screen.

```
 Process   Options   Help
File Manager Edit Dynamic Template                          Line 1 of 16
Command ===> _                                              Scroll CSR

1 Edit template field attributes and selection

Cmd Con (  Start  Length Type  Op  Value                              )
*** ****  Top of data  ****
___     _     1     20  AN   ___ ____________________________________ _
___ AND _    +0      5  AN   ___ ____________________________________ _
___ AND _    +1      7  AN   ___ ____________________________________ _
___ AND _    +1      7  AN   ___ ____________________________________ _
___ AND _    +1      7  AN   ___ ____________________________________ _
___ AND _    +1      7  AN   ___ ____________________________________ _
___ AND _    +1      7  AN   ___ ____________________________________ _
___ AND _    +1      7  AN   ___ ____________________________________ _
___ AND _    +1      3  AN   ___ ____________________________________ _
___ AND _    +0      3  AN   ___ ____________________________________ _
___ AND _ ______ ______ __   ___ ____________________________________ _
___ AND _ ______ ______ __   ___ ____________________________________ _
___ AND _ ______ ______ __   ___ ____________________________________ _
 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F5=RFind     F7=Up
 F8=Down      F9=Swap     F10=RunTemp  F11=Fldname  F12=Cancel
```

Figure 23-39 Dynamic template generation

3. Figure 23-40 presents the sample data sorted and presented with this dynamic template, including the addition of individual field names. Note that field numbers are also added.

```
 Process   Options   Help
File Manager Browse REDBK3.REDBOOK.PDPAK.COMPMAST                 Template saved
Command ===> _                                                      Scroll CSR
                                                   Record 0         Format TABL
Company Name          Share Value Value-1 Value-2 Value-3 Value-4 Value-5 Value
#2                    #3          #4      #5      #6      #7      #8      #9  +
AN 1:20               AN 21:5     AN 27:7 AN 35:7 AN 43:7 AN 51:7 AN 59:7 AN 67
<---+----1----+----> <--->        <---+-> <---+-> <---+-> <---+-> <---+-> <---+
****  Top of data  ****
Casey_Import_Export   00079       0000077 0000078 0000072 0000070 0000065 00000
Glass_and_Luget_plc   00019       0000022 0000025 0000020 0000016 0000020 00000
Headworth_Electrical  00124       0000131 0000133 0000133 0000133 0000137 00001
IBM                   00163       0000163 0000162 0000160 0000161 0000159 00001
ShareSelect           00119       0000120 0000118 0000116 0000116 0000112 00001
SportSelect           00224       0000224 0000220 0000217 0000215 0000215 00002
Veck_Transport        00036       0000034 0000034 0000035 0000037 0000039 00000
****  End of data  ****

 F1=Help      F2=Zoom      F3=Exit      F4=CRetriev  F5=RFind     F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 23-40 Dynamic template usage

4. You can edit this template to add a condition. Type the TE command and press Enter. You can specify your condition here or type 1 on the command line, and press Enter to display again the dynamic template panel as shown in Figure 23-41.

```
 Process   Options   Help
File Manager Edit Dynamic Template                              Line 1 of 10
Command ===> _                                                   Scroll CSR

Cmd Con (  Start  Length Type  Op  Value                                   )
*** ****  Top of data  ****
___     _     1      20  AN    __  ____________________________________ _
___ AND _    +0       5  AN    >   00100_________________________________ _
___ AND _    +1       7  AN    __  ____________________________________ _
___ AND _    +1       7  AN    __  ____________________________________ _
___ AND _    +1       7  AN    __  ____________________________________ _
___ AND _    +1       7  AN    __  ____________________________________ _
___ AND _    +1       7  AN    __  ____________________________________ _
___ AND _    +1       7  AN    __  ____________________________________ _
___ AND _    +1       3  AN    __  ____________________________________ _
___ AND _    +0       3  AN    __  ____________________________________ _
*** ****  End of data  ****

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F5=RFind     F7=Up
 F8=Down      F9=Swap     F10=RunTemp  F11=Fldname  F12=Cancel
```

Figure 23-41 Adding a condition to a dynamic template

5. You can alternatively specify the selection criteria by field name as shown in Figure 23-42.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
File Manager Edit Dynamic Template                                Line 1 of 10
Command ===> _____________________________________________________ Scroll CSR

Cmd Con ( Field Name           Op  Value
*** ****  Top of data  ****
___       _ Company Name       ___ ______________________________________
___   AND _ Share Value       >__ 00100 ___________________________________
___   AND _ Value-1            ___ ______________________________________
___   AND _ Value-2            ___ ______________________________________
___   AND _ Value-3            ___ ______________________________________
___   AND _ Value-4            ___ ______________________________________
___   AND _ Value-5            ___ ______________________________________
___   AND _ Value-6            ___ ______________________________________
___   AND _ Buy                ___ ______________________________________
___   AND _ Sell               ___ ______________________________________
*** ****  End of data  ****

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F5=RFind     F7=Up
 F8=Down      F9=Swap     F10=RunTemp  F11=Fldname  F12=Cancel
```

Figure 23-42 Adding a condition to a dynamic template using field name

Either way, the result is shown in Figure 23-43.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
File Manager Browse REDBK3.REDBOOK.PDPAK.COMPMAST
Command ===> ______________________________________________________ Scroll CSR
                                                  Record 0         Format TABL
Company Name         Share Value Value-1 Value-2 Value-3 Value-4 Value-5 Value
#2                   #3          #4      #5      #6      #7      #8      #9  +
AN 1:20              AN 21:5     AN 27:7 AN 35:7 AN 43:7 AN 51:7 AN 59:7 AN 67
<---+----1----+----> <--->       <---+-> <---+-> <---+-> <---+-> <---+-> <---+
****  Top of data  ****
- - - - - - - - - - - - - - - - - - - - - 2 Line(s) not selected
Headworth_Electrical 00124       0000131 0000133 0000133 0000133 0000137 00001
IBM                  00163       0000163 0000162 0000160 0000161 0000159 00001
ShareSelect          00119       0000120 0000118 0000116 0000116 0000112 00001
SportSelect          00224       0000224 0000220 0000217 0000215 0000215 00002
- - - - - - - - - - - - - - - - - - - - - 1 Line(s) not selected
****  End of data  ****

 F1=Help      F2=Zoom      F3=Exit      F4=CRetriev  F5=RFind     F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 23-43 Result of template with selection

23.1.18 Compare data sets using templates

For this example, we created a sequential copy of the sample data set VEHICLE1 and changed (using the edit function) the company name for three records (vehicle model changed from Verada to Veranda) as shown in Figure 23-44.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 File Manager Browse TSS05.TESTDATA(VEHICLE1)
 Command ===> _____________________________________________________ Scroll CSR
                                                  Record 0         Format TABL
 VEHICLE-TYPE VEHICLE-MAKE         VEHICLE-MODEL        MODEL-CODE VEHICLE-SUB-
 #2           #3                   #4                   #5         #6          +
 AN 1:1       AN 2:20              AN 22:20             AN 42:6    AN 48:20
 -            <---+----1----+----> <---+----1----+----> <---+>     <---+----1--
 ****  Top of data  ****
 C            Mitsubishx           Verada               KF         Ei
 C            Mitsubishi           Verada               KH         Ei
 C            Mitsubishi           Verada               KH         Ei
 C            Subaru               Outback              2Gen
 C            Subaru               Outback              3Gen
 C            Subaru               Outback              3Gen
 C            Subaru               Outback              3Gen
 C            Toyota               Camry                MCV20R     Conquest
 C            Toyota               Camry                MCV20R     Conquest
 C            Toyota               Camry                MCV20R     CSi
 C            Toyota               Camry                SXV20R     CSi
 C            Toyota               Camry                SXV20R     CSi
  F1=Help      F2=Zoom      F3=Exit      F4=CRetriev  F5=RFind     F7=Up
  F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 23-44 The data set that was modified

Select the compare utility: option **3** (Utilities), then option **11** (Compare).

Specify both files names (and templates) that you want to compare, or select them in lists, then specify the characteristics of the comparison as shown in Figure 23-45.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 File Manager                  Compare Utility
 Command ===> _____________________________________________________________

 Compare Options:
    Compare type            Synchronization         Listing type
    1  1. Record            1  1. One-to-one         1  1. Summary
       2. Formatted            2. Read-ahead           2. Delta
                               3. Keyed                3. Matching
                                                       4. Long
                                                       5. None

 Processing Options:                     Listing Options:
    Enter "/" to select option              Enter "/" to select option
    _  Edit template mapping                _  Wide listing
    /  Clear print data set                 _  Show hex chars
                                            /  Highlight changes
                                            _  Show field attributes


  F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
  F9=Swap     F10=Actions  F12=Cancel
```

Figure 23-45 Comparison characteristics

Example 23-22 presents the result of the operation.

Example 23-22 Results

```
IBM File Manager for z/OS
* * * * Record comparison with one-to-one synchronization
* * New data set: REDBK2.FM(VEHICLE1)
* *               using template TSS05.BASE.TEMPLATE(VEHICLE)
* * Old data set: TSS05.TESTDATA(VEHICLE1)
* *               using template TSS05.BASE.TEMPLATE(VEHICLE)

Comparison summary:

              Old data set records processed:         20
              New data set records processed:         20
                      Matching records found:         17
                Old data set records deleted:          3
               New data set records inserted:          3
                       Total changed records:          3
                              Paired changes:          3
    Non-paired old data set record deletions:          0
   Non-paired new data set record insertions:          0

                           Synchronization: One-to-one
                           Comparison type: Record
                              Listing type: Summary

********************************* End of data ********************************
```

24

File Manager/DB2

This chapter describe the capabilities of File Manager for DB2 Data and how to use them.

24.1 Overview of FM/DB2

File Manager for DB2 Data (FM/DB2) is a powerful set of utility functions for editing, browsing, printing, copying, and maintaining DB2 data. It also provides utilities for listing DB2 objects, managing DB2 privileges, generating JCL to run DB2 standalone utilities, exporting and importing DB2 tables to or from QSAM or VSAM data sets, creating data to populate DB2 tables, and prototyping SQL SELECT statements.

On the Primary Options Menu shown in Figure 24-1, you can choose which DB2 SSID to connect. FM/DB2 will establish a connection with a chosen DB2 system. If you do not specify a value, FM/DB2 will present you with the DB2 Subsystem Selection panel. Place a / or S against the required DB2 system in the list displayed.

```
Process   Options   Utilities   Help
FM/DB2                        Primary Option Menu
Command ===>

0  Settings      Set processing options                User ID . : REDBK3
1  Browse        Browse DB2 table or view              System ID : STLABF2
2  Edit          Edit DB2 table                        Appl ID . : FMN2
3  Utilities     Perform utility functions             Version . : 4.1.0
4  SQL           Prototype, execute and analyze SQL    Terminal  : 3278A
5  DB2I          Start DB2 Interactive                 Screen  . : 1
X  Exit          Terminate FM/DB2                      Date  . . : 2003/10/31
                                                       Time  . . : 13:47

                                                       DB2 SSID  . DB1B
                                                       SQL ID  . . REDBK3
```

Figure 24-1 Primary Options Menu

24.2 How to edit a table

This section describes how to perform a DB2 edit using FM.

1. Select option **2** in the Primary Option Menu. The DB2 edit panel is displayed. Fill in owner and name of the table (wildcards are allowed), and press Enter.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------
FM/DB2                              DB2 Edit
Command ===> _____________________________________________________________

Specify the DB2 Object:
   Location  . . . . . _________________        Database . . ________ (optional)
   Owner . . . . . . . REDBK3 __                Table space  ________ (optional)
   Name  . . . . . . . EMP _________________

Template:
   Data set name . . . ____________________________________________
   Member  . . . . . . ________

Processing Options:
   Template usage                      Enter "/", "A"lways to select option
   3  1. Above                         _  Edit options
      2. Previous                      _  Edit template
      3. Generate from table           _  Re-edit template
      4. Generate/Replace
```

Figure 24-2 DB2 Edit panel

The Table Edit screen is displayed as shown in Figure 24-3.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------
FM/DB2                              Table Edit
Command ===> ____________________________________________________________ Scroll PAGE
LOCATION                      NAME REDBK3.EMP                            Format TABL
       EMPNO  FIRSTNME      MIDINIT LASTNAME        WORKDEPT PHONENO HIREDATE
       #1     #2            #3      #4              #5       #6      #7
       CH(6)  VARCHAR(12)   CH(1)   VARCHAR(15)     CH(3)    CH(4)   DATE
       <---+> <---+----1-> -       <---+----1----> <->      <-->    <---+---->
000001 000010 CHRISTINE<    I       HAAS<           A00      3978    1965-01-01
000002 000110 VINCENZO<     G       LUCCHESI<       A00      3490    1958-05-16
000003 000120 SEAN<                 O'CONNELL<      A00      2167    1963-12-05
000004 200010 DIAN<         J       HEMMINGER<      A00      3978    1965-01-01
000005 200120 GREG<                 ORLANDO<        A00      2167    1972-05-05
000006 000020 MICHAEL<      L       THOMPSON<       B01      3476    1973-10-10
000007 000030 SALLY<        A       KWAN<           C01      4738    1975-04-05
000008 000130 DOLORES<      M       QUINTANA<       C01      4578    1971-07-28
000009 000140 HEATHER<      A       NICHOLLS<       C01      1793    1976-12-15
000010 200140 KIM<          N       NATZ<           C01      1793    1976-12-15
000011 000060 IRVING<       F       STERN<          D11      6423    1973-09-14
000012 000150 BRUCE<                ADAMSON<        D11      4510    1972-02-12
000013 000160 ELIZABETH<    R       PIANKA<         D11      3782    1977-10-11
 F1=Help      F2=Zoom      F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 24-3 Table Edit panel

2. To insert a new row in the table, you simply put an **I** on the line as shown in Figure 24-4 and press Enter.

```
 Process   Options   Utilities   Help
FM/DB2                              Table Edit
Command ===> _____________________________________________________ Scroll PAGE
LOCATION                   NAME REDBK3.EMP                        Format TABL
       EMPNO  FIRSTNME     MIDINIT LASTNAME        WORKDEPT PHONENO HIREDATE
       #1     #2           #3      #4              #5       #6      #7
       CH(6)  VARCHAR(12)  CH(1)   VARCHAR(15)     CH(3)    CH(4)   DATE
       <---+> <---+----1-> -       <---+----1----> <->      <-->    <---+---->
000000 ****  Top of data  ****
000001 000010 CHRISTINE<   I       HAAS<           A00      3978    1965-01-01
i0002 000110 VINCENZO<    G       LUCCHESI<       A00      3490    1958-05-16
000003 000120 SEAN<                O'CONNELL<      A00      2167    1963-12-05
000004 200010 DIAN<        J       HEMMINGER<      A00      3978    1965-01-01
000005 200120 GREG<                ORLANDO<        A00      2167    1972-05-05
000006 000020 MICHAEL<     L       THOMPSON<       B01      3476    1973-10-10
000007 000030 SALLY<       A       KWAN<           C01      4738    1975-04-05
000008 000130 DOLORES<     M       QUINTANA<       C01      4578    1971-07-28
000009 000140 HEATHER<     A       NICHOLLS<       C01      1793    1976-12-15
000010 200140 KIM<         N       NATZ<           C01      1793    1976-12-15
000011 000060 IRVING<      F       STERN<          D11      6423    1973-09-14
000012 000150 BRUCE<               ADAMSON<        D11      4510    1972-02-12
```

Figure 24-4 Inserting rows

After pressing Enter, a new line opens as shown in Figure 24-5.

```
 Process   Options   Utilities   Help
FM/DB2                              Table Edit
Command ===> _____________________________________________________ Scroll PAGE
LOCATION                   NAME REDBK3.EMP                        Format TABL
       EMPNO  FIRSTNME     MIDINIT LASTNAME        WORKDEPT PHONENO HIREDATE
       #1     #2           #3      #4              #5       #6      #7
       CH(6)  VARCHAR(12)  CH(1)   VARCHAR(15)     CH(3)    CH(4)   DATE
       <---+> <---+----1-> -       <---+----1----> <->      <-->    <---+---->
000000 ****  Top of data  ****
000001 000010 CHRISTINE<   I       HAAS<           A00      3978    1965-01-01
000002 000110 VINCENZO<    G       LUCCHESI<       A00      3490    1958-05-16
000003         <                   <
000004 000120 SEAN<                O'CONNELL<      A00      2167    1963-12-05
000005 200010 DIAN<        J       HEMMINGER<      A00      3978    1965-01-01
000006 200120 GREG<                ORLANDO<        A00      2167    1972-05-05
000007 000020 MICHAEL<     L       THOMPSON<       B01      3476    1973-10-10
000008 000030 SALLY<       A       KWAN<           C01      4738    1975-04-05
000009 000130 DOLORES<     M       QUINTANA<       C01      4578    1971-07-28
000010 000140 HEATHER<     A       NICHOLLS<       C01      1793    1976-12-15
000011 200140 KIM<         N       NATZ<           C01      1793    1976-12-15
000012 000060 IRVING<      F       STERN<          D11      6423    1973-09-14
```

Figure 24-5 Row inserted

3. Enter your information as in the sample in Figure 24-6.

```
 Process   Options   Utilities   Help
FM/DB2                              Table Edit
Command ===> _____________________________________________________ Scroll PAGE
LOCATION                      NAME REDBK3.EMP                     Format TABL
       EMPNO  FIRSTNME     MIDINIT LASTNAME        WORKDEPT PHONENO HIREDATE
       #1     #2           #3      #4              #5       #6      #7
       CH(6)  VARCHAR(12)  CH(1)   VARCHAR(15)     CH(3)    CH(4)   DATE
       <---+> <---+----1-> -       <---+----1----> <->      <-->    <---+---->
000000 ****  Top of data  ****
000001 000010 CHRISTINE<   I       HAAS<           A00      3978    1965-01-01
000002 000110 VINCENZO<    G       LUCCHESI<       A00      3490    1958-05-16
000003 00234  umberto      h       McScott         a00      6767    1928-09-17
000004 000120 SEAN<                O'CONNELL<      A00      2167    1963-12-05
000005 200010 DIAN<        J       HEMMINGER<      A00      3978    1965-01-01
000006 200120 GREG<                ORLANDO<        A00      2167    1972-05-05
000007 000020 MICHAEL<     L       THOMPSON<       B01      3476    1973-10-10
000008 000030 SALLY<       A       KWAN<           C01      4738    1975-04-05
000009 000130 DOLORES<     M       QUINTANA<       C01      4578    1971-07-28
000010 000140 HEATHER<     A       NICHOLLS<       C01      1793    1976-12-15
000011 200140 KIM<         N       NATZ<           C01      1793    1976-12-15
000012 000060 IRVING<      F       STERN<          D11      6423    1973-09-14
 F1=Help      F2=Zoom     F3=Exit     F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down     F9=Swap    F10=Left     F11=Right    F12=Cancel
```

Figure 24-6 Results of row inserted

4. If you want to delete rows from the table, use the line command **D**. In Figure 24-7 we show how to use the block delete using **DD** on lines 42 and 44.

```
 Process   Options   Utilities   Help
FM/DB2                              Table Edit
Command ===> _____________________________________________________ Scroll PAGE
LOCATION                      NAME REDBK3.EMP                     Format TABL
       EMPNO  FIRSTNME     MIDINIT LASTNAME        WORKDEPT PHONENO HIREDATE
       #1     #2           #3      #4              #5       #6      #7
       CH(6)  VARCHAR(12)  CH(1)   VARCHAR(15)     CH(3)    CH(4)   DATE
       <---+> <---+----1-> -       <---+----1----> <->      <-->    <---+---->
000039 000330 WING<                LEE<            E21      2103    1976-02-23
000040 000340 JASON<       R       GOUNOT<         E21      5698    1947-05-05
000041 200330 HELENA<              WONG<           E21      2103    1976-02-23
dd0042 200340 ROY<         R       ALONZO<         E21      5698    1947-05-05
000043        <                    <
dd0044 00234  umberto<     h       McScott<        a00      6767    1928-09-17
000045 ****  End of data  ****
 F1=Help      F2=Zoom     F3=Exit     F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down     F9=Swap    F10=Left     F11=Right    F12=Cancel
```

Figure 24-7 Deleting rows

5. Press Enter; you will see the result shown in Figure 24-8.

```
 Process   Options   Utilities   Help
FM/DB2                              Table Edit
Command ===> _____________________________________________________ Scroll PAGE
LOCATION                    NAME REDBK3.EMP                       Format TABL
       EMPNO  FIRSTNME     MIDINIT LASTNAME        WORKDEPT PHONENO HIREDATE
       #1     #2           #3      #4              #5       #6      #7
       CH(6)  VARCHAR(12)  CH(1)   VARCHAR(15)     CH(3)    CH(4)   DATE
       <---+> <---+----1-> -       <---+----1----> <->      <-->    <---+---->
000039 000330 WING<                LEE<            E21      2103    1976-02-23
000040 000340 JASON<       R       GOUNOT<         E21      5698    1947-05-05
000041 200330 HELENA<              WONG<           E21      2103    1976-02-23
000042 ****  End of data  ****

 F1=Help      F2=Zoom      F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 24-8 Rows deleted

6. If you need to back out your changes, use the command CAN as shown in Figure 24-9.

```
 Process   Options   Utilities   Help
FM/DB2                              Table Edit
Command ===> can__________________________________________________ Scroll PAGE
LOCATION                    NAME REDBK3.EMP                       Format TABL
       EMPNO  FIRSTNME     MIDINIT LASTNAME        WORKDEPT PHONENO HIREDATE
       #1     #2           #3      #4              #5       #6      #7
       CH(6)  VARCHAR(12)  CH(1)   VARCHAR(15)     CH(3)    CH(4)   DATE
       <---+> <---+----1-> -       <---+----1----> <->      <-->    <---+---->
000039 000330 WING<                LEE<            E21      2103    1976-02-23
000040 000340 JASON<       R       GOUNOT<         E21      5698    1947-05-05
000041 200330 HELENA<              WONG<           E21      2103    1976-02-23
000042 ****  End of data  ****

 F1=Help      F2=Zoom      F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 24-9 Cancel command entered

7. Press Enter. A confirmation pop-up panel is displayed as shown in Figure 24-10.

```
                          Confirm Cancel
 Command ===> _____________________________________________

 Data has been changed.                                          Scroll PAGE
                                                                 Format TABL
 Instructions:                                                 ENO HIREDATE
                                                                   #7
    Press ENTER key to confirm cancel request.                )  DATE
    (All changes will be discarded)                              <---+---->
                                                                 1976-02-23
    Enter END or EXIT command to return to edit or view          1947-05-05
    session.                                                     1976-02-23

  F1=Help       F2=Split      F3=Exit       F7=Backward
  F8=Forward    F9=Swap      F12=Cancel                        F6=RChange
                                                              F12=Cancel
```

Figure 24-10 Confirm Cancel command

8. After confirming the cancel request by pressing Enter, confirmation of successful roll-back is given as shown in Figure 24-11.

```
 Process   Options   Utilities   Help
FM/DB2                              DB2 Edit                    Rollback issued
Command ===> ____________________________________________________________________

Specify the DB2 Object:
   Location  . . . . . ________________        Database . . ________ (optional)
   Owner . . . . . . . REDBK3___               Table space  ________ (optional)
   Name  . . . . . . . EMP_________________

Template:
   Data set name . . . ____________________________________________
   Member  . . . . . . ________

Processing Options:
   Template usage                        Enter "/", "A"lways to select option
   3  1. Above                           _  Edit options
      2. Previous                       _  Edit template
      3. Generate from table            _  Re-edit template
      4. Generate/Replace

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 24-11 Confirmation of roll-back

24.3 Utility function to create a table

The Utility Functions panel shown in Figure 24-12 is displayed when you select **3**, **Utilities** from the Primary Option Menu. This panel lists the utilities you can use to manipulate DB2 data.

```
 Process   Options   Utilities   Help
FM/DB2                          Utility Functions
Command ===> 2

1  Print         Print DB2 table or view
2  Objects       Create and drop DB2 objects
3  Copy          Copy data within DB2
4  Object List   Display and process DB2 object lists
5  Privileges    Manage DB2 privileges
6  Import        Import sequential or VSAM data into DB2
7  Export        Export DB2 data to sequential or VSAM data set
8  Create        Create DB2 test data
9  Utilities     DB2 utility job generation
10 Audit trail   Print audit trail report
11 Print browse  Browse FM/DB2 print data set

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 24-12 Utility Function panel

In the following example we describe how to make a copy of an existing table.

1. Select **2 Create and drop DB2 objects** on the Utility Functions panel and press Enter. The panel DB2 Object Functions is displayed as shown in Figure 24-13.

```
 Process   Options   Utilities   Help
FM/DB2                       DB2 Object Functions
Command ===> C

      C Create object                        D Drop object

Processing Options:
   Object Type
   3   1. Database         5.  Alias             9.  Function
       2. Table space      6.  Index             10. Stored procedure
       3. Table            7.  Synonym           11. Trigger
       4. View             8.  Distinct type     12. Auxiliary Table (Create)

   Enter "/" to select option
   /  Confirm object drop

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 24-13 DB2 Object Functions panel

2. Select **C** for Create object and **3** for Table, and press Enter. The panel Create Table is now displayed as shown in Figure 24-14.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------
FM/DB2                            Create Table
Command ===> ___________________________________________________________
                                                                  More:     +
New Table:
   Owner . . . .  REDBK2__          (optional)
   Name  . . . .  EMP_____________  (required)
   Database  . .  ________          (optional)
   Table Space .  ________          (optional)

Model Table/View:  (optional)       Usage:
   Owner . . . .  REDBK3__          1 1. Generate LIKE clause
   Name  . . . .  EMP_____________    2. Load table information

Table Creation:
   Creation Options              (Use option 1 before options 2, 4-9)
   1  1. Columns                 (normally required, invalid with model LIKE)
      2. Nulls/default values    (optional)
      3. Table options           (editproc,validproc,more; optional)
      4. Unique Constraints      (primary/unique keys; optional)
      5. Referential Constraints (foreign keys; optional)
      6. Check constraints       (optional)
      7. Procedure exits         (optional)
```

Figure 24-14 Create Table panel

As shown on this panel, enter the following data for the new table:

– Owner - **REDBK3**

– Name - **EMP**

Also enter information for the model table:

– Owner - **REDBK2**

– Name - **EMP**

3. Enter **9** in the field Creation Options of the Create Table panel shown in Figure 24-15 and press Enter. This causes the statement to execute and the table to be created.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------
FM/DB2                            Create Table
Command ===> _____________________________________________________________
                                                                   More:   -
   Table Space .  ________          (optional)

Model Table/View:  (optional)       Usage:
   Owner . . . .  REDBK3            1 1. Generate LIKE clause
   Name  . . . .  EMP                 2. Load table information

Table Creation:
   Creation Options                 (Use option 1 before options 2, 4-9)
   9  1. Columns                    (normally required, invalid with model LIKE)
      2. Nulls/default values       (optional)
      3. Table options              (editproc,validproc,more; optional)
      4. Unique Constraints         (primary/unique keys; optional)
      5. Referential Constraints    (foreign keys; optional)
      6. Check constraints          (optional)
      7. Procedure exits            (optional)
      8. Generate values            (optional)
      9. Create the table
 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 24-15 Table created confirmation

4. To see the created table, select option **4** (Object List) as shown in Figure 24-16 and press Enter.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------
FM/DB2                          Utility Functions
Command ===> 4____________________________________________________________

1  Print         Print DB2 table or view
2  Objects       Create and drop DB2 objects
3  Copy          Copy data within DB2
4  Object List   Display and process DB2 object lists
5  Privileges    Manage DB2 privileges
6  Import        Import sequential or VSAM data into DB2
7  Export        Export DB2 data to sequential or VSAM data set
8  Create        Create DB2 test data
9  Utilities     DB2 utility job generation
10 Audit trail   Print audit trail report
11 Print browse  Browse FM/DB2 print data set




 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 24-16 Utility Functions panel

5. The Object List Utility is now displayed as shown in Figure 24-17. Enter **REDBK3** in the owner field (wildcard is also allowed) and press Enter.

```
 Process   Options   Utilities   Help
FM/DB2                        Object List Utility
Command ===> ____________________________________________________________
                                                                  More:     +
   blank Display object list                 P Print object list

Object Identification Criteria:
   Location  . . . . . . . ______________     Enter * for list
   Owner . . . . . . . . . REDBK2
   Name  . . . . . . . . . ________________
   Database/collect/schema ________________

Additional Selection Criteria:
   Column  . . . . . . . . ________________ Enter * to list catalog columns
   Operator  . . . . . . . ___________
   Value . . . . . . . . . ________________________________________________
                           ________________________________________________

Processing Options:
   Object Type
   3  1. Database          6.  Synonym            11. Schema
 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 24-17 Object List Utility panel

6. The Tables, Views and Aliases panel is now displayed as shown in Figure 24-18. Put the cursor in the Cmd field and enter a question mark (?); a list of commands you can perform from this field is shown as in Figure 24-19.

```
 Process   Options   Utilities   Help
FM/DB2                    Tables, Views and Aliases            Row 1 to 1 of 1
Command ===> ___________________________________________________ Scroll PAGE

Location:

Cmd   Name               Owner    T DB Name  TS Name    Cols        Rows Checks
      *                  *        * *        *          *           *      *

?     EMP                REDBK2   T DSNDB04  EMPA1J7C     14          -1      0
****************************** END OF DB2 DATA ******************************

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 24-18 Tables, Views and Aliases panel

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------
FM/DB2                     Object List Line Commands         Row 14 to 26 of 42
Command ===> _____________________________________________________ Scroll PAGE

Type S against the required line command and press Enter.

Sel Command  Description

_   CRA      Create alias
_   CRX      Create index
_   CS       Describe columns
_   D        Show database
_   DR       Drop object
_   E        Edit table
_   FK       Show foreign keys
_   G        Grant privileges
_   GEN      Generate SQL for table from DB2 catalog
_   I        Details about table/view
_   LAB      Create a label
_   P        Show privileges
_   PA       Show parent tables
 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 24-19 Object List Lines Commands panel

24.4 Copy data to a table

Perform the following steps to copy data to a table:

1. From the Utility Functions panel, select **3** as shown in Figure 24-20 and press Enter.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------
FM/DB2                        Utility Functions
Command ===> 3____________________________________________________________

1  Print         Print DB2 table or view
2  Objects       Create and drop DB2 objects
3  Copy          Copy data within DB2
4  Object List   Display and process DB2 object lists
5  Privileges    Manage DB2 privileges
6  Import        Import sequential or VSAM data into DB2
7  Export        Export DB2 data to sequential or VSAM data set
8  Create        Create DB2 test data
9  Utilities     DB2 utility job generation
10 Audit trail   Print audit trail report
11 Print browse  Browse FM/DB2 print data set





 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 24-20 Utility Functions panel

2. The Copy Utility panel shown in Figure 24-21 is displayed. Define the object you want to copy, for example:
   – Owner - **REDBK3**

– Table - **EMP**

– Copy count - **ALL**

Press Enter.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------
FM/DB2                            Copy Utility
Command ===> ____________________________________________________________

From DB2 Object:
   Location  . . . . . _______________        Database . . ________ (optional)
   Owner . . . . . . . REDBK3__               Table space  ________ (optional)
   Name  . . . . . . . EMP_______________

   Copy count  . . . . ALL_____       Number of rows to copy

From Template:
   Data set name . . . ____________________________________________
   Member  . . . . . . ________

Processing Options:
   Template usage                     Enter "/", "A"lways to select option
   3  1. Above                        _  Edit template
      2. Previous
      3. Generate from table
      4. Generate/Replace
 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 24-21 Copy Utility panel

3. The To DB2 Object panel displays as shown in Figure 24-22. Define the owner and name as shown and press Enter.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------
Copy from REDBK3.EMP
Command ===> ____________________________________________________________

To DB2 Object:
   Location  . . . . . _______________        Database . . ________ (optional)
   Owner . . . . . . . REDBK2__               Table space  ________ (optional)
   Name  . . . . . . . EMP_______________



To Template:
   Data set name . . . ____________________________________________
   Member  . . . . . . ________

Processing Options:
   Template usage                     Enter "/", "A"lways to select option
   3  1. Above                        _  Edit copy options
      2. Previous                     _  Edit template mapping
      3. Generate from table          _  Batch execution
      4. Generate/Replace
 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 24-22 DB2 Object panel

Copy processing starts. When it is finished, you get a message indicating how many rows have been copied, as shown in Figure 24-23.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------------
FM/DB2                          Copy Utility                 45 row(s) copied
Command ===> _____________________________________________________________________

From DB2 Object:
   Location  . . . . . _______________        Database . . ________ (optional)
   Owner . . . . . . . REDBK3__               Table space  ________ (optional)
   Name  . . . . . . . EMP______________

   Copy count  . . . . ALL____       Number of rows to copy

From Template:
   Data set name . . . ________________________________________________
   Member  . . . . . . ________

Processing Options:
   Template usage                    Enter "/", "A"lways to select option
   3  1. Above                       _  Edit template
      2. Previous
      3. Generate from table
      4. Generate/Replace
 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 24-23 Copy confirmation

25

File Manager/IMS

This chapter describes how to use File Manager to manipulate IMS data.

You can use File Manager/IMS to:

- Browse, edit, or print data in an IMS database
- Extract data from, load data into, or erase data from, an IMS database
- Format segments according to record structures defined in COBOL or PL/I copy books
- Access databases via DL/I or BMP processing, and belonging to an IMS DC, DBCTL, or CICS region

For many tasks, you can use the File Manager IMS template and view features. These allow you to define a logical view of a data set based on a COBOL or PL/I copybook. Associating a view with a data set lets you define which fields and records you want to work with, how the fields are displayed, and which segments are displayed.

25.1 Navigating an IMS database

This section demonstrates some of the navigation functionality that can be used in browsing an IMS database.

Start by establishing the edit/browse options using option **0.7** from the FM/IMS Primary Option Menu. The Edit/Browse Options panel is shown in Figure 25-1.

```
 Process   Options   Help
 ------------------------------------------------------------------------
 FM/IMS                        Edit/Browse Options
 Command ===> ___________________________________________________________

 Enter "/" to select option
 /  Translate data in data area to uppercase on input
 _  Show field reference number of SNGL display
 /  Show field type and length values on SNGL display
 /  Use * (any segment type) as default on NEXT and PREVIOUS commands
 /  Set SHOW SUP ON, exposing suppressed groups as default
 _  Set SCOPE DBREC, limiting segments to within a database record
 _  Set AUTOSAVE ON, issuing a SAVE after checkpoint frequency in options 0.8

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 25-1 Edit/Browse Options panel

After selecting database browse, enter the name of the database you wish to browse, along with the associated DBD name, and indicate that you will browse in DLI mode as shown in Figure 25-2. Although not shown here, we also selected PSB=DYNAMIC in option **0.8** from the Primary Option Menu, and Fetch DSNames from profile and Save DSNames in profile in option **0**.

```
 Process   Options   Help
FM/IMS                  Browse Entry Panel (Dynamic PSB)
Command ===> _

DBD:
   Data set name . . . 'FMN.V4R1M0.DBDLIB'
   Member  . . . . . . DJ1G

IMS:
   Subsystem Name  . . ____    (If BMP)
   AGN Name  . . . . . ________

Processing Options:
   Region Type           Enter "/" to select option
   1  1. DLI             _  Secondary index
      2. BMP             /  Use view
                         _  Use dynalloc DB dsnames

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 25-2 Browse Entry (Dynamic PSB) panel

Press Enter to view details about the data sets associated with database DJ1G as shown in Figure 25-3.

```
 Process   Options   Help
FM/IMS             Browse : Database Data Set Specification
Command ===> _                                                  Scroll PAGE
Press ENTER to confirm usage of the specified data set(s)
Database DJ1G

DBD name  ddname    Data set name
DJ1E      DJ1E      'FMN.V4R1M0.DB.DJ1E'
DJ2E      DJ2E      'FMN.V4R1M0.DB.DJ2E'
DJ2F      DJ2F      'FMN.V4R1M0.DB.DJ2F'
DJ3E      DJ3E      'FMN.V4R1M0.DB.DJ3E'
DJ3F      DJ3F      'FMN.V4R1M0.DB.DJ3F'
DJ1F      DJ1F      'FMN.V4R1M0.DB.DJ1F'
**** End of data ****

Processing Options:
   Fetch dsnames from            Enter "/" to select option
   1  1. Profile                 /  Save dsnames in profile
      2. Dynamic Allocation data set

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 25-3 Database Data Set Specification

As shown in Figure 25-4, choose to use a template without specifying a view as you browse the database.

```
 Process   Options   Help
FM/IMS                    Browse : View Specification     Command is not active
Command ===> _____________________________________________________________

Database DJ1G

Template:
   Data set name . . . 'TSS05.IMS.TEMPLATE'_____________________________

View:
   Data set name . . . ____________________________________________
   Member  . . . . . . ________
                                 ▌
Processing Options:
   View usage                          Enter "/" to select option
   1  1. New                           _  Save view on return
      2. Existing
      3. None


 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 25-4 View Specification panel

Press Enter; the Database Positioning panel is returned. Select the SHIRE segment as shown in Figure 25-5.

```
 Process   Options   Help
FM/IMS                   Browse : Database Positioning
Command ===> ____________________________________________________ Scroll PAGE

Database DJ1G                        Key sequence                Format TABL
View     Temporary                          Key
Cmd SXE Level     Segment   Description     len  Key value
s▌_ SX  1         SHIRE                     20  BROOME_____________
___  X   2        SHIRENP                    0
___  X   2        SHRLSUB                   18  ..................
___  X    3       SUBLSTR                   28  ............................
___  X    3       LETRBOX                    4  ....
   **** End of data ****


 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F5=Key >=    F7=Backward
 F8=Forward   F9=Swap     F10=Actions  F12=Cancel
```

Figure 25-5 Database Positioning panel

Press Enter; you can now see details about the segments associated with your database as shown in Figure 25-6. Scroll through this using PF keys.

```
 Process   Options   Help
FM/IMS                      Browse : IMS Database DJ1G
Command ===> ____________________________________________________ Scroll PAGE
                                      SHOW SUP ON  Scope ALL    Format TABL
                     SHIRE-NAME       SHIRE-TYPE SHIRE-CODE1 SHIRE-CODE2
Cmd Level Segment    #3               #4                  #5          #6
                     AN 1:20          AN 21:1        ZD 22:3     ZD 25:5
                     <---+----1----+----> -                <-->      <---+>
                     ****   Top of window   ****
___  1     SHIRE     BROOME           1                    1         998
___   2    SHIRENP   No key
=LG  2     SHRLSUB   Key=BROOME
=LG   3    SUBLSTR   Key=AARONS                DR
=LG   3    SUBLSTR   Key=ANNE                  ST
=LG   3    SUBLSTR   Key=ANTHEOUS              WY
=LG   3    SUBLSTR   Key=ANTINA                WY
=LG   3    SUBLSTR   Key=ARCHER                ST
=LG   3    SUBLSTR   Key=ASAHI                 ST
=LG   3    SUBLSTR   Key=AURORA                ST
=LG   3    SUBLSTR   Key=BAGOT                 ST
=LG   3    SUBLSTR   Key=BARDWELL              ST
 F1=Help      F2=Format    F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 25-6 Database DJ1G

Cursor-select the segment with key ANTINA and press PF2; this returns the details about this segment as shown in Figure 25-7. Note that Format has changed from TABL to SNGL.

```
 Process   Options   Help
FM/IMS                      Browse : IMS Database DJ1G            Length error
Command ===> ____________________________________________________ Scroll PAGE
                                      SHOW SUP ON  Scope ALL    Format SNGL
Segment  SUBLSTR                   Level: 3
Concatenated key value:  BROOME            BROOME            ANTINA
Current type is SUBURB-LINK-STREET                  Top Field is 1    of 7
Field                    Typ  Len   Data
SUB-STREET-NAME          AN    18 K ANTINA
SUFFIX                   AN     4 K
DIRECTION                AN     1 K
STREET                   AN    18   WY   ANTINA
SUFFIX                   AN     4
DIRECTION                AN     1
EMERGENCY-PHONE          AN     4
****  End of record  ****

 F1=Help      F2=Format    F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap     F10=Previous F11=Next     F12=Cancel
```

Figure 25-7 Database DJ1G detail

Enter **`next shire`** on the command line and press Enter; this moves you to the next SHIRE segment in the database. As shown in Figure 25-8, this is DENMARK.

```
 Process   Options   Help
FM/IMS                        Browse : IMS Database DJ1G
Command ===> _____________________________________________________ Scroll PAGE
                                        SHOW SUP ON  Scope ALL   Format SNGL
Segment  SHIRE                        Level: 1
Concatenated key value:  DENMARK
Current type is SHIRE-TOWN                          Top Field is 1    of 9
Field          Typ  Len   Data
SHIRE-NAME     AN    20 K DENMARK
SHIRE-TYPE     AN     1   1
SHIRE-CODE1    ZD     3    305
SHIRE-CODE2    ZD     5      244
SHIRE-YEAR     BI     2     6280
SHIRE-BRICK    PD     8       606363187610
SHIRE-WOODEN   PD     8      2034233224332
SHIRE-SCHOOL   PD     4    3020312
SHIRE-GOLF     BI     2    12608
****  End of record  ****


 F1=Help      F2=Format    F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap     F10=Previous F11=Next     F12=Cancel
```

Figure 25-8 Next SHIRE segment

Change the Format to CHAR and return to the database positioning panel using PF3 as shown in Figure 25-9.

```
 Process   Options   Help
FM/IMS                   Browse : Database Positioning
Command ===> ____________________________________________________ Scroll PAGE

Database DJ1G                        Key sequence                Format CHAR
View     Temporary                        Key
Cmd SXE Level     Segment   Description     len  Key value
___  SX  1        SHIRE                     20  DENMARK
x__   X  2        SHIRENP                    0
x__   X  2        SHRLSUB                   18  ..................
x__   X   3       SUBLSTR                   28  ............................
x__   X   3       LETRBOX                    4  ....
   **** End of data ****


 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F5=Key >=    F7=Backward
 F8=Forward   F9=Swap     F10=Actions  F12=Cancel
```

Figure 25-9 Database positioning panel

Deselect all segments except the SHIRE segment by placing an X against each. Deselection could also be accomplished by:

- Entering X4 against SHIRENP
- Entering XX against SHIRENP and XX against LETRBOX
- Entering X* against SHIRENP

Returning to the database positioning panel and entering **E** against the SHIRE segment, you receive the Segment Layout Selection panel as shown in Figure 25-11.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
FM/IMS                  View : Segment Layout Selection
Command ===> ____________________________________________________________ Scroll PAGE

View     Temporary
Template TSS05.IMS.TEMPLATE(DJ1G)
Segment  SHIRE

Cmd   Sel Segment layout name                                   Offset Length
           ****  Top of data  ****
E__   *   SHIRE                                                  0      53
___   *   SHIRE-TOWN                                             0      53
___   *   SHIRE-CITY                                             0      53
           ****  End of data  ****

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F5=RFind     F6=Describe
 F7=Up        F8=Down      F9=Swap     F12=Cancel
```

Figure 25-10 Segment Layout Selection panel

Specifying **E** in the Cmd field enables you to enter Field Selection Criteria for the SHIRE segment as shown in Figure 25-11.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
FM/IMS                  View : Segment Layout     SSA expression generated
Command ===> ____________________________________________________________ Scroll PAGE
View  Temporary                                              Segment SHIRE
Template TSS05.IMS.TEMPLATE(DJ1G)
Offset 0
Cmd Seq SHC Ref Rdf Field name                          Key Type  Start Length
                    ****  Top of data  ****
___ ___         1     1 SHIRE                                AN      1     53
___ ___         2      2 SHIRE-KEY                        K   AN      1     20
___ ___         3       3 SHIRE-NAME                      K   AN      1     20
___ ___         4      2 SHIRE-TYPE                           AN     21      1
___ ___         5      2 SHIRE-CODE1                          ZD     22      3
___ ___         6      2 SHIRE-CODE2                          ZD     25      5
___ ___         7      2 SHIRE-ROAD                           PD     30      8
___ ___         8      2 SHIRE-RIVER                          PD     38      8
----------------------- Layout Identification Criteria -----------------------
#4='0'
-------------------------- Field Selection Criteria ----------------------------
#3<G
 ------------------------------------------------------------------------------
 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F5=RFind     F6=Describe
 F7=Up        F8=Down      F9=Swap     F11=Headings F12=Cancel
```

Figure 25-11 SHIRE segment layout

Through similar steps, you can specify selection criteria for the SHIRE-TOWN and SHIRE-CITY segments as shown in Figure 25-12 and Figure 25-13.

```
 Process   Options   Help
FM/IMS                        View : Segment Layout     SSA expression generated
Command ===> _                                                   Scroll PAGE
View  Temporary                                              Segment SHIRE
Template TSS05.IMS.TEMPLATE(DJ1G)
Offset 0
Cmd Seq SHC Ref Rdf Field name                          Key Type  Start Length
                    ****  Top of data  ****
___ ___        1     1 SHIRE-TOWN                          AN      1     53
___ ___        2      2 SHIRE-KEY                       K  AN      1     20
___ ___        3       3 SHIRE-NAME                     K  AN      1     20
___ ___        4      2 SHIRE-TYPE                          AN     21      1
___ ___        5      2 SHIRE-CODE1                         ZD     22      3
___ ___        6      2 SHIRE-CODE2                         ZD     25      5
___ ___        7      2 SHIRE-YEAR                          BI     30      2
___ ___        8      2 SHIRE-BRICK                         PD     32      8
----------------------- Layout Identification Criteria -----------------------
#4='1'
-------------------------- Field Selection Criteria --------------------------
#3>=G & #3<=K
 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F5=RFind     F6=Describe
 F7=Up        F8=Down      F9=Swap     F11=Headings F12=Cancel
```

Figure 25-12 SHIRE-TOWN segment layout

```
 Process   Options   Help
FM/IMS                        View : Segment Layout
Command ===> _                                                   Scroll PAGE
View  Temporary                                              Segment SHIRE
Template TSS05.IMS.TEMPLATE(DJ1G)
Offset 0
Cmd Seq SHC Ref Rdf Field name                          Key Type  Start Length
                    ****  Top of data  ****
___ ___        1     1 SHIRE-CITY                          AN      1     53
___ ___        2      2 SHIRE-KEY                       K  AN      1     20
___ ___        3       3 SHIRE-NAME                     K  AN      1     20
___ ___        4      2 SHIRE-TYPE                          AN     21      1
___ ___        5      2 SHIRE-CODE1                         ZD     22      3
___ ___        6      2 SHIRE-CODE2                         ZD     25      5
___ ___        7      2 SHIRE-MAJOR                         BI     30      2
___ ___        8      2 SHIRE-MAIN-ATTRACTION               AN     32     20
----------------------- Layout Identification Criteria -----------------------
#4='2'
-------------------------- Field Selection Criteria --------------------------
#3>K
 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F5=RFind     F6=Describe
 F7=Up        F8=Down      F9=Swap     F11=Headings F12=Cancel
```

Figure 25-13 SHIRE-CITY segment layout

Returning to the Segment Layout Selection screen, you now see the short message `Retrieving with qual SSA` in the top right-hand corner, meaning that you have completed entry of field selection criteria for the SHIRE segment as shown in Figure 25-14.

```
 Process   Options   Help
FM/IMS                 View : Segment Layout Selectio Retrieving with qual SSA
Command ===> _                                                   Scroll PAGE

View     Temporary
Template TSS05.IMS.TEMPLATE(DJ1G)
Segment  SHIRE

Cmd   Sel Segment layout name                                  Offset Length
          ****  Top of data  ****
___    *  SHIRE                                                  0     53
___    *  SHIRE-TOWN                                             0     53
___    *  SHIRE-CITY                                             0     53
          ****  End of data  ****

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F5=RFind     F6=Describe
 F7=Up        F8=Down      F9=Swap     F12=Cancel
```

Figure 25-14 Completed SHIRE segment field selection

At this point, let's say you decide that you still need to be more granular with the selection. Return to the Database Positioning panel and reselect the LETRBOX segment as shown in Figure 25-15.

```
 Process   Options   Help
FM/IMS                 Browse : Database Positioning
Command ===> _                                                   Scroll PAGE

Database DJ1G                      Key sequence                  Format CHAR
View C   Temporary                        Key
Cmd SXE Level     Segment   Description   len  Key value
___ SXE 1         SHIRE                   20  DENMARK
___       2       SHIRENP                  0
___       2       SHRLSUB                 18  ..................
___         3     SUBLSTR                 28  ............................
X__         3     LETRBOX                  4  ....
   **** End of data ****

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F5=Key >=    F7=Backward
 F8=Forward   F9=Swap     F10=Actions  F12=Cancel
```

Figure 25-15 Database Positioning pane

The SHIRE segment is again selected (**S** in the Cmd field), giving a display starting with the first segment (ESPERANCE) that satisfies our selection criteria. This is shown in Figure 25-16 and Figure 25-17.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
FM/IMS                    Browse : Database Positioning
Command ===> ____________________________________________________________ Scroll PAGE

Database DJ1G                          Key sequence                    Format CHAR
View C   Temporary                            Key
Cmd SXE Level     Segment   Description     len  Key value
s__ SXE 1         SHIRE                      20  DENMARK____________
___      2        SHIRENP                     0
___      2        SHRLSUB                    18  ..................
___       3       SUBLSTR                    28  ............................
___   X   3       LETRBOX                     4  ....
   **** End of data ****

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F5=Key >=    F7=Backward
 F8=Forward   F9=Swap     F10=Actions  F12=Cancel
```

Figure 25-16 SHIRE segment selected

```
 Process   Options   Help
 ------------------------------------------------------------------------------
FM/IMS                    Browse : IMS Database DJ1G
Command ===> _____________________________________________________ Scroll PAGE
                        SHOW SUP ON  Scope ALL    Col 1_________ Format CHAR
Cmd Level Segment      ----+----1----+----2----+----3----+----4----+----5----+---
                       ****   Top of window   ****
___ 1     SHIRE        ESPERANCE           060401116.......................
___ 1     SHIRE        EXMOUTH             080600190.......................
___ 1     SHIRE        PERTH               212406576..KINGS PARK          .r
___   3   LETRBOX     6892EAST PERTH PLB
___   3   LETRBOX     6902LEEDERVILLE PLB A
___   3   LETRBOX     6903LEEDERVILLE PLB B
___   3   LETRBOX     6931MAYLANDS PLB
___   3   LETRBOX     6915MOUNT HAWTHORN PLB
___   3   LETRBOX     6906NORTH PERTH PLB
___   3   LETRBOX     6865NORTHBRIDGE PLB
___   3   LETRBOX     6001PERTH PLB
___   3   LETRBOX     6831PERTH ST GEORGES TCE PLB
___   3   LETRBOX     6832PERTH EAST ST GEORGES TCE PLB
___   3   LETRBOX     6837PERTH PLB A1 - A48
___   3   LETRBOX     6838PERTH PLB B49 - B96
 F1=Help      F2=Format    F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap      F10=Left     F11=Right    F12=Cancel
```

Figure 25-17 ESPERANCE segment

To add further selection criteria, edit the LETRBOX segment by entering an **E** against it in the Database Positioning panel. You can now enter the selection expression for the LETRBOX segment, as shown in Figure 25-18.

```
 Process   Options   Help
FM/IMS                        View : Segment Layout     Retrieving with qual SSA
Command ===> _                                                   Scroll PAGE
View  Temporary                                                Segment LETRBOX
Template TSS05.IMS.TEMPLATE(DJ1G)
Offset 0
Cmd Seq SHC Ref Rdf Field name                          Key Type  Start Length
                    ****  Top of data  ****
___ ___       1     1 LETTER-BOX                         AN      1     34
___ ___       2      2 LETTER-BOX-KEY                  K  AN      1      4
___ ___       3       3 POST-CODE                      K  AN      1      4
___ ___       4      2 LOCATION                         AN      5     30
                    ****  End of data  ****

------------------------- Field Selection Criteria --------------------------
#3>6903 & #4>=PERTH
 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F5=RFind     F6=Describe
 F7=Up        F8=Down      F9=Swap     F11=Headings F12=Cancel
```

Figure 25-18 LETRBOX segment selected

Returning to the Database Positioning panel, enter **S** against the SHIRE segment, noting that selection starts from the ESPERANCE SHIRE segment, as shown in Figure 25-19.

```
 Process   Options   Help
FM/IMS                   Browse : Database Positioning
Command ===> _                                                   Scroll PAGE

Database DJ1G                        Key sequence                  Format CHAR
View C   Temporary                          Key
Cmd SXE Level     Segment   Description     len  Key value
s_  SXE 1         SHIRE                      20  ESPERANCE
___       2       SHIRENP                     0
___       2       SHRLSUB                    18  ..................
___        3      SUBLSTR                    28  ............................
___   XE   3      LETRBOX                     4  ....
   **** End of data ****

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F5=Key >=    F7=Backward
 F8=Forward   F9=Swap     F10=Actions  F12=Cancel
```

Figure 25-19 Database Positioning panel

This selection criteria has resulted in just three segments being returned, as shown in Figure 25-20.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
FM/IMS                        Browse : IMS Database DJ1G            REXX required
Command ===> _____________________________________________________ Scroll PAGE
                          SHOW SUP ON  Scope ALL    Col 1_________ Format CHAR
Cmd Level Segment    ----+----1----+----2----+----3----+----4----+----5----+---
                     ****   Top of window   ****
___ 1     SHIRE      PERTH                212406576..KINGS PARK          .r
___   3   LETRBOX    6979VICTORIA PARK PLB
___   3   LETRBOX    6981VICTORIA PARK EAST PLB
                     ****  End of database  ****
 F1=Help      F2=Format    F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 25-20 Three segments returned

This example has illustrated just some of the versatility of database navigation possible with FM/IMS.

25.2 Template processing

Templates provide the power to obtain different views of your data by using the structure defined in a copybook to present the data in a file. This section includes examples.

Note: The tutorial in the *IBM File Manager for z/OS User's Guide and Reference,* SC18-7792 contains the best explanation of the creation of a template from a copybook.

25.2.1 It really does remember the copybook

The Processing Options section appears on most of the File Manager panels. In Figure 25-21, it is shown in the lower left-hand portion of the Edit Entry Panel.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
File Manager                    Edit Entry Panel

Input Partitioned, Sequential or VSAM Data Set:
   Data set name . . . . . 'FMN.SFMNSAM1'                                  
   Member  . . . . . . . . FMNCDATA      Blank or pattern for member list
   Volume serial . . . . . ______        If not cataloged
   Starting position . . . ________________________________ Key,Slot,RBA,Record

Copybook or Template:
   Data set name . . . . . 'DAVINR1.DEMOS.TEMPLATE'                        
   Member  . . . . . . . . VSAM00 ___    Blank or pattern for member list

Processing Options:
   Copybook/template usage            Enter "/" to select option
   1  1. Above                        _  Edit template
      2. Previous                     _  Create audit trail
      3. None
      4. Create dynamic
```

Figure 25-21 Processing options section on File Manager Edit Entry Panel

Whenever you use a copybook (or a template) to process a data set, and select **Above** as a value for the processing options, File Manager saves the relation information in a member of your ISPF profile.

That information can be used at a later time. For example, let's say you want to edit a data set, but you do not recall the name of the copybook. If you select **Previous,** you can leave the Copybook Data Set Name and Member fields blank. File Manager will supply them based on the retained profile information.

Note: If the member is no longer in the data set, or if the data set no longer exists, you will receive one of two messages:

- `No matching member name`
- `Data set not found`

Even with this drawback, this is still a useful feature for development efforts.

25.2.2 Using the SELECT command in edit templates

You can dynamically edit templates to control which fields are displayed or printed in a data set browse or edit session.

In Figure 25-22, start by initiating a browse of the VEHICLE1 data set using the VEHICLE template.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
File Manager                    Browse Entry Panel
Command ===> _____________________________________________________________________

Input Partitioned, Sequential or VSAM Data Set:
   Data set name . . . . . 'TSS05.TESTDATA'
   Member  . . . . . . . . VEHICLE1     Blank or pattern for member list
   Volume serial . . . . . ______        If not cataloged
   Starting position . . . ______________________________ Key,Slot,RBA,Record

Copybook or Template:
   Data set name . . . . . 'TSS05.BASE.TEMPLATE'
   Member  . . . . . . . . VEHICLE___    Blank or pattern for member list

Processing Options:
   Copybook/template usage            Enter "/" to select option
   3  1. Above                        _  Edit template
      2. Previous
      3. None
      4. Create dynamic

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
 F9=Swap     F10=Actions  F12=Cancel
```

Figure 25-22 Browse Entry Panel

Data from the file is displayed as shown in Figure 25-23, formatted according to the saved template. Access the template edit function by entering **TE** on the primary command line.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
File Manager Browse TSS05.TESTDATA(VEHICLE1)
Command ===> TE_____________________________________________________ Scroll CSR
                                 Record 0_______ Col 1________ Format CHAR
----+----1----+----2----+----3----+----4----+----5----+----6----+----7----+---
****  Top of data  ****
CMitsubishx          Verada              KF    Ei                  199700011BBM
CMitsubishi          Verada              KH    Ei                  199900041ALW
CMitsubishi          Verada              KH    Ei                  200000011APL
CSubaru              Outback             2Gen                      199600019IT-
CSubaru              Outback             3Gen                      199800021AIU
CSubaru              Outback             3Gen                      199900011BDM
CSubaru              Outback             3Gen                      200000021BCN
CToyota              Camry               MCV20RConquest            199900011ALM
CToyota              Camry               MCV20RConquest            200000011ARM
CToyota              Camry               MCV20RCSi                 199800021AIE
CToyota              Camry               SXV20RCSi                 199800011ABX
CToyota              Camry               SXV20RCSi                 199900011ANC
TFord                Courier             CO                        199600011AXX
TFord                Transit             TU                        199800011ABX
THolden              Rodeo               RR                        199900011ABD
 F1=Help      F2=Zoom      F3=Exit      F4=CRetriev  F5=RFind     F6=RChange
 F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 25-23 Template Edit command

After selecting **E** in the Template Edit panel, the next display you see is the Field Selection/Edit panel, in the default state of no fields selected. In this situation, File Manager regards all fields as being selected. Enter an **S** against VEHICLE-TYPE and press Enter. This field is now selected, as shown in Figure 25-24.

```
 Process   Options   Help
File Manager                  Field Selection/Edit
Command ===> _____________________________________________________ Scroll CSR

1 Specify Record Identification by field.
2 Specify Record Selection      by field.
Offset _____0
Cmd Seq SHC Ref Rdf Field Name                          Type  Start Length
                    ****  Top of data  ****
___ ___         1     1 VEHICLE-REC                      AN      1   8475
___ ___ S       2      2 VEHICLE-TYPE                    AN      1      1
___ ___         3      2 VEHICLE-MAKE                    AN      2     20
___ ___         4      2 VEHICLE-MODEL                   AN     22     20
---------------------- Record Identification Criteria -------------------------

------------------------- Record Selection Criteria ---------------------------

 F1=Help      F2=Split     F3=Exit      F4=CRetriev  F5=RFind     F6=Describe
 F7=Up        F8=Down      F9=Swap     F10=RunTemp  F11=Headings F12=Cancel
```

Figure 25-24 Field Selection Edit panel

Using PF10, run the template again. Now only the VEHICLE-TYPE field is shown in Figure 25-25.

```
 Process   Options   Help
File Manager Browse TSS05.TESTDATA(VEHICLE1)
Command ===> _____________________________________________ Scroll CSR
                                          Record 0_______ Format TABL
VEHICLE-TYPE MODEL-CODE VEHICLE-SUB-MODEL    YEAR-OF-MANUFACTURE
#2           #5         #6                                 #7
AN 1:1       AN 42:6    AN 48:20                        ZD 68:4
-            <---+>     <---+----1----+---->              <--->
****  Top of data  ****
C            KF         Ei                                1997
C            KH         Ei                                1999
C            KH         Ei                                2000
C            2Gen                                         1996
C            3Gen                                         1998
C            3Gen                                         1999
C            3Gen                                         2000
C            MCV20R     Conquest                          1999
C            MCV20R     Conquest                          2000
C            MCV20R     CSi                               1998
C            SXV20R     CSi                               1998
C            SXV20R     CSi                               1999
 F1=Help      F2=Zoom      F3=Exit      F4=CRetriev  F5=RFind     F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 25-25 Results of PF10 RunTemp

If you now return to template edit and specify SELECT ALL ON, you see in Figure 25-26 that all fields are now selected.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 File Manager                 Field Selection/Edit
 Command ===> _____________________________________________________ Scroll CSR

 1 Specify Record Identification by field.
 2 Specify Record Selection      by field.
 Offset _____0
 Cmd Seq SHC Ref Rdf Field Name                                Type  Start Length
                     ****  Top of data  ****
 ___ ___ S     1     1 VEHICLE-REC                             AN      1   8475
 ___ ___ S     2      2 VEHICLE-TYPE                           AN      1      1
 ___ ___ S     3      2 VEHICLE-MAKE                           AN      2     20
 ___ ___ S     4      2 VEHICLE-MODEL                          AN     22     20
 ---------------------- Record Identification Criteria ------------------------
 ------------------------------------------------------------------------------

 ------------------------ Record Selection Criteria -----------------------------
 ------------------------------------------------------------------------------
 ------------------------------------------------------------------------------
 ------------------------------------------------------------------------------
 ------------------------------------------------------------------------------
  F1=Help      F2=Split     F3=Exit      F4=CRetriev  F5=RFind     F6=Describe
  F7=Up        F8=Down      F9=Swap     F10=RunTemp  F11=Headings F12=Cancel
```

Figure 25-26 Field Selection/Edit panel

Using PF10 to run the template selection, you see in Figure 25-27 that all the default fields are displayed.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 File Manager Browse TSS05.TESTDATA(VEHICLE1)
 Command ===> _____________________________________________________ Scroll CSR
                                                   Record 0_______ Format TABL
 VEHICLE-REC
 #1                                                                          +
 AN 1:8475
 <---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+---
 ****  Top of data  ****
 CMitsubishx          Verada              KF    Ei                  199700011BB
 CMitsubishi          Verada              KH    Ei                  199900041AL
 CMitsubishi          Verada              KH    Ei                  200000011AP
 CSubaru              Outback             2Gen                      199600019IT
 CSubaru              Outback             3Gen                      199800021AI
 CSubaru              Outback             3Gen                      199900011BD
 CSubaru              Outback             3Gen                      200000021BC
 CToyota              Camry               MCV20RConquest            199900011AL
 CToyota              Camry               MCV20RConquest            200000011AR
 CToyota              Camry               MCV20RCSi                 199800021AI
 CToyota              Camry               SXV20RCSi                 199800011AB
 CToyota              Camry               SXV20RCSi                 199900011AN
  F1=Help      F2=Zoom      F3=Exit      F4=CRetriev  F5=RFind     F7=Up
  F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 25-27 Results of PF10 RunTemp

Returning to template edit, deselect VEHICLE-REC and VEHICLE-TYPE, by entering an **S** against each, and then re-run the template. The results are shown in Figure 25-28 with all default fields except for VEHICLE-TYPE.

```
 Process   Options   Help
File Manager Browse TSS05.TESTDATA(VEHICLE1)
Command ===> _____________________________________________ Scroll CSR
                                          Record 0       Format TABL
VEHICLE-MAKE         VEHICLE-MODEL        MODEL-CODE VEHICLE-SUB-MODEL
#3                   #4                   #5         #6
AN 2:20              AN 22:20             AN 42:6    AN 48:20
<---+----1----+----> <---+----1----+----> <---+>     <---+----1----+---->
****  Top of data  ****
Mitsubishx           Verada               KF         Ei
Mitsubishi           Verada               KH         Ei
Mitsubishi           Verada               KH         Ei
Subaru               Outback              2Gen
Subaru               Outback              3Gen
Subaru               Outback              3Gen
Subaru               Outback              3Gen
Toyota               Camry                MCV20R     Conquest
Toyota               Camry                MCV20R     Conquest
Toyota               Camry                MCV20R     CSi
Toyota               Camry                SXV20R     CSi
Toyota               Camry                SXV20R     CSi
 F1=Help      F2=Zoom      F3=Exit      F4=CRetriev  F5=RFind     F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 25-28 Results of PF10 RunTemp

25.2.3 How to process COPY REPLACING statements

For applications that have source programs containing the construct shown in Example 25-1, File Manager has a feature that enables the processing of COPY REPLACING statements.

Example 25-1 Source code with a COPY REPLACING statement

```
01  CANCEL-IN-RECORD.
    COPY COPYSMP1 REPLACING ==:CUSTIO:== BY == CANCEL-IN ==.
```

In this example, use File Manager to create a template with the substituted values in the following manner:

1. Review the source statements in your application program to ascertain the value of the pseudo-text and the string that should be substituted.
2. Access File Manager in your ISPF session.
3. Go to option **7** (Templates).

   The Template Workbench panel is displayed.
4. Enter the copybook data set name and member.
5. Enter the template data set name and member.
6. Select the **Options** pull-down.
7. Type **6** to adjust your File Manager processing options, as shown in Figure 25-29, and press Enter.

```
 Process    Options    Help
File Man   6 _1. Print settings
             2. System settings
CC Creat     3. Tape processing options          eld/record in template
CM Creat     4. Job card specifications          template from copybook
MC Map f     5. Compiler language selection      m template
             6. COBOL compiler specifications
Copybook     7. PL/I compiler specifications
   Data      8. Edit/Browse options
   Membe     9. ISPF settings
Template
   Data set name . DEMOS.TEMPLATE
   Member  . . . . VSAM01
Model Template:
   Data set name .
   Member  . . . .
```

Figure 25-29 Selecting options before creating a template

8. On the Set Processing Options panel, adjust the COBOL Replacing Options by doing the following:
   a. Enter the From string (the pseudo-text) that is found in the copybook.
   b. Enter the To string (the string) that should be placed in the source program at compile time.

   Your panel should look similar to the one in Figure 25-30.

```
 Process    Options    Help
File Manager            Set COBOL Processing Options
                                                                  More:     +
Additional SYSLIB data sets:
1.  ____________________________________________
2.  ____________________________________________
3.  ____________________________________________
4.  ____________________________________________
5.  ____________________________________________
6.  ____________________________________________
7.  ____________________________________________
8.  ____________________________________________
9.  ____________________________________________
10. ____________________________________________

COBOL Replacing Options:
   From string                          To string
1. ==:CUSTIO:==                    by ==CANCEL-IN==
2. ________________________________ by ________________________________
3. ________________________________ by ________________________________
4. ________________________________ by ________________________________
Command ===> ____________________________________________________________
```

Figure 25-30 COBOL copy replacing options in File Manager

In this case, the pseudo-text `==:CUSTIO:==` is replaced by the string `==CANCEL-IN==`.

Tip: You must include the equal signs (delimiters) for the File Manager processing to work. If you do not, the compilation of the copybook will fail with a return code of 8.

9. Press PF3 to save these settings.
10. Type **cc** on the command line of the Template Workbench panel, and press Enter to create the template.

You can have up to five sets of values specified in your profile. The substitution of values is based on the strings found in your copybook.

25.2.4 How to build a template for multi-record file layouts

Some application programs use files that have multiple record types. In certain instances, the record structures are contained in several copybooks. If this is the case at your site, you need to use a little bit of ingenuity to map your files.

File Manager displays data if a record structure matches a copybook file layout. For File Manager to perform this task with different members from the same data set, you must provide a single point of reference.

To do that, create a new copybook; and using a valid COBOL construct, include the other copybooks in it using the COPY command. Example 25-2 shows what this would look like.

Example 25-2 Copybook with nested multiple copybooks

```
****  INCLUDE MULTIPLE COPY MEMBERS IN ONE LOCATION.
****  THIS WILL LET YOU CREATE A TEMPLATE IN FILE MANAGER
****  FOR A MULTI-RECORD FILE.

 01 HEADER-REC.
    COPY CPY01.
 01 DETAIL-T01.
    COPY CPY02.
 01 DETAIL-T02.
    COPY CPY03.
 01 DETAIL-T03.
    COPY CPY04.
 01 TRAILER-REC.
    COPY CPY05.
```

In this example, our data file contains a header record, three detail records, and a trailer record. Five copybooks define the entire structure of the file.

File Manager can now manipulate this one copybook and create a template for it like any other copybook. You now have the ability to view or edit the entire file without any errors.

25.3 Hints and tips

This section presents some information we discovered during the preparation of this redbook, which might be useful to you.

Additional Tip: The ISPF settings to jump to the action bar can cause you to exit File Manager when the last cursor field is completed and you press Enter. Reset the ISPF settings to avoid this situation.

25.3.1 System programmer notes

In this section we present several items that system programmers should keep in mind after they install File Manager.

ISPF skeleton modification

Customize the job card file tailoring skeleton FMNFTJOB for your site.

The code in this skeleton, shown in Example 25-3, determines if any of the Set Processing Options (option **0**) values are filled in for a job card. If none of them are, then a job card is created dynamically from this skeleton.

Example 25-3 ISPF skeleton FMNFTJOB

```
)CM
)CM Description: File Manager ISPF file tailoring job card skeleton.
)CM
)CM Copyright:
)CM
)CM  Licensed Materials - Property of IBM
)CM
)CM  5655-L26 File Manager for z/OS Version 4
)CM
)CM  (C) Copyright IBM Corp. 2000, 2003.
)CM  All Rights Reserved
)CM
)CM  US Government Users Restricted Rights - Use, duplication
)CM  or disclosure restricted by GSA ADP Schedule Contract
)CM  with IBM Corp.
)CM
)CM Modifications:
)CM
)SET FMNJCGEN = 0
)SEL &FMNPJC1 = &Z && &FMNPJC2 = &Z && &FMNPJC3 = &Z && &FMNPJC4 = &Z
//&ZUSER.B JOB &ZACCTNUM,
//             &ZUSER,MSGCLASS=A,
//             NOTIFY=&ZUSER,CLASS=A,
//             MSGLEVEL=(1,1)
)CM Set flag to indicate job card information was generated
)SET FMNJCGEN = 1
)ENDSEL
)SEL &FMNJCGEN ¬= 1
&FMNPJC1
&FMNPJC2
&FMNPJC3
&FMNPJC4
)ENDSEL
//*
```

Pay particular attention to the CLASS and MSGCLASS parameters in the JOB card. We found that message class A does not produce any output at this site. It took several phone calls and e-mails to verify the product was working correctly. After we changed the message class to H, we saw the output we expected.

Application programmer ISPF profile alert

File Manager creates an ISPF profile member called FMNTMHST to keep track of the data set-to-copybook (or data set-to-template) association. If your site's application programmers are typical, they manipulate dozens of data sets as part of their work effort. Each association causes this table to grow (by approximately 100 bytes).

If the ISPF profile data set does not have enough space, at some point some program product (not necessarily File Manager) will not be able to update a table.

You should review the default attributes of the ISPF profile data set that is created for new application programmers and make any necessary changes. Also, consider reviewing the size of existing profile data sets to see if they are approaching either a directory block or extent limit.

APF authorization

The *IBM File Manager for z/OS Customization Guide*, GC18-7790, has some interesting notes regarding File Manager and APF authorization.

The message displayed on the ISPF screen when the VER command is issued always indicates that File Manager is not running APF authorized, because it cannot run APF authorized under ISPF.

You can determine if File Manager is APF authorized by executing a batch job and including the following input statement:

```
$$FILEM VER
```

25.3.2 Look out for your PF keys

While you are in option **1** (Data Set Browse) or in option **2** (Data Set Edit), you cannot use PF2 to split the screen. The key list assigned to these File Manager options changes the value of PF2 to ZOOM. We found this to be annoying (even frustrating) at times, especially because we do not display our PF key settings.

> **Note:** Our recommendation for new users (even though we know it uses up valuable screen real estate) is to turn on the function key display until you get used to the product. To do that, use either one of the following commands:
>
> ```
> FKA ON
> PFSHOW ON
> ```

If you want to split your screen at that time, use the START command.

We discovered that File Manager has 29 different key lists. You need to be aware of this as you navigate through the product.

25.3.3 What to do when a copybook fails to compile

Sometimes when you create a template, File Manager is unable to compile the copybook. When that happens, an error panel is displayed as shown in Figure 25-31.

```
                          Compilation Errors
 _
 F
    The compile of the copybook produced a return code -  12
 C  Select one of the following options:
 C
 M  1. View the compilation listing
    2. Abort processing
 C  3. Retry (recompile copybook)
    4. Retry (with alternate compiler)
    5. Ignore the error and continue processing
 T

 M

    Command ===> _
```

Figure 25-31 File Manager Compilation Errors panel

We recommend that you always review the compilation listing. This way you can quickly judge how much work is involved in correcting the error.

After you select option **1**, the compilation listing is displayed using the Print Browse function as shown in Figure 25-32.

To locate the start of your copybook, issue the FIND command with your copybook as the string. Then issue the RFIND command. (There is no LAST operand for the FIND command in the Print Browse function).

```
  Menu  Utilities  Compilers  Help
 ------------------------------------------------------------------------
 BROWSE    SYS02291.T082904.RA000.DAVINR1.R0101392    Line 00000069 Col 001 080
  000007          _         Data Division.
  000008                    Working-Storage Section.
  000010                       COPY "FMNCCPY3".
  000011C                   01 REC-TYPE01.
  000012C                      03 REC-TYPE              PIC XX.
  000013C                      03 REC-ID REDEFINES REC-TYPE  PIC XX.
  000014C                      03 NAME                   PIC X(20).
  000015C                      03 EMPLOYEE-NO            PIC 9(4) BINARY.
  000016C                      03 AGE                    PIC 9(4) BINARY.
  000017C                      03 SALARY                 PIC 9(7) PACKED-DECIMAL.
  000018C                      03 MONTH    PIC 9(8) BINARY OCCURS 12 TIMES.
  000019C                      03 FILLER                 PIC XX.
  000020C                   01 REC-TYPE02.
  000021C                      05 REC-TYPE               PIC XY.

==000021==> IGYDS1160-S Character(s) "Y" were found as "PICTURE" string characte
                          "PICTURE" string of "S9(1)" was assumed.

  000022C                      05 NAME                   PIC X(20).
  000023C                      03 JOB-TITLE              PIC X(14).
 Command ===> ____________________________________________ Scroll ===> CSR
```

Figure 25-32 File Manager copybook compile listing at the point of error

Review the listing to determine what changes are necessary. Press PF3 twice to return to the Template Workbench panel.

25.3.4 Record structure defined in source application program

If the record structure of one of your application files is contained in an application source program, File Manager cannot use it.

You must extract the record layout and create a copybook to perform any file manipulation with File Manager.

We suggest a certain amount of caution when doing this. You should use whatever resources are available to you so that you do not create multiple versions of the same file layout.

25.3.5 Watch out for that bad disposition

Every time we used option **3.1** (Data Create Utility), the value in the Disposition field automatically reverted to `Mod`. This may have been related to the Systems Managed Storage (SMS) rules at this site.

Depending on your file structure, this might not matter to you. Otherwise, you will have more data in your file than you may have intended. You just have to be careful.

25.4 Copybook and include

File Manager can process PL/I Includes as well as COBOL copybooks as shown in Example 25-4 and Example 25-5.

Example 25-4 COBOL copybook

```
01 REC-TYPE01.
          03 REC-TYPE              PIC XX.
          03 NAME                  PIC X(20).
          03 EMPLOYEE-NO           PIC 9(4) BINARY.
          03 AGE                   PIC 9(4) BINARY.
          03 SALARY                PIC 9(7) PACKED-DECIMAL.
          03 MONTH    PIC 9(8) BINARY OCCURS 12 TIMES.
          03 FILLER                PIC XX.
       01 REC-TYPE02.
          05 REC-TYPE              PIC XX.
          05 NAME                  PIC X(20).
          03 JOB-TITLE             PIC X(14).
          03 ADDR1                 PIC X(20).
          03 ADDR2                 PIC X(20).
          03 POSTCODE              PIC X(4).
```

Example 25-5 PL/I Include

```
  DECLARE 1 REC_TYPE01,
            2 REC_TYPE        CHAR(2),
            2 NAME            CHAR(20),
            2 EMPLOYEE_NO     BINARY FIXED(8),
            2 AGE             BINARY FIXED(8),
            2 SALARY          DECIMAL FIXED(7),
            2 MONTH           DIM(12) BINARY FIXED(16),
            2 FILLER          CHAR(2);
```

```
DECLARE 1 REC_TYPE02,
          2 REC_TYPE        CHAR(2),
          2 NAME            CHAR(20),
          2 JOB_TITLE       CHAR(14),
          2 ADDR1           CHAR(20),
          2 ADDR2           CHAR(20),
          2 POSTCODE        CHAR(4);
```

File Manager produces the same template with either of these two descriptions.

26

Test data and File Manager

When an application is tested, data is needed. There could be several types of data required for testing, for example, sequential and VSAM data sets, data from DB2 and IMS databases, mostly existing on DASD. The data might contain dummy values generated according to templates, or it could be extracted from production data. Moreover, some relations between data from different sources may exist.

In this chapter, the capability to create test data with File Manager for z/OS and its DB2 and IMS components is described. Also, some information regarding a new tool, File Export for z/OS is presented.

Some panels are intentionally omitted from the series of figures presented in this chapter. Refer to the corresponding product documentation to fill in the gaps.

26.1 Creating MVS data using File Manager and copybooks

Using File Manager on line or in a batch mode, the user can create data from scratch using the File Manager data generator, which allows you to create data based on record layout.

26.1.1 Data creation on line

To generate data on line follow these steps:

1. Select option **3** Utilities on the main panel of File Manager, and then select option **1** on the Utility Functions panel, as shown in Figure 26-1.

```
 Process   Options   Help
 ------------------------------------------------------------------------
 File Manager                  Utility Functions
 Command ===> _________________________________________________________

 0  DBCS           Set DBCS data format for print
 1  Create         Create data
 2  Print          Print data
 3  Copy           Copy data
 4  Dslist         Catalog services
 5  VTOC           Work with VTOC
 6  Find/Change    Search for and change data
 7  AFP            Browse AFP data
 8  Storage        Browse user storage
 9  Printdsn       Browse File Manager print data set
 10 Loadlib        View load module information
 11 Compare        Compare data
 12 Audit trail    Print audit trail report
 13 Copybook       View and Print
```

Figure 26-1 Utility Functions panel of File Manager

2. Fill in the required and optional fields, for example, the name of the target data set, the number of records, and the name of the copybook or template, as shown in Figure 26-2.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
File Manager                  Data Create Utility
Command ===> _____________________________________________________________________
                                                                        More:     +
Output Partitioned, Sequential or VSAM Data Set:
   Data set name . . . . . FMN1.PDS _____________________________________
   Member  . . . . . . . . TEST1___      (Blank or pattern for member list)
   Volume serial . . . . . ______        (If not cataloged)
   Record length . . . . . 144___        Optional record length for RECFM V
   Records . . . . . . . . 5_______      number of records
   Fillchar  . . . . . . . _____         char or hex value, AN, BIN, or RAND
   Key position  . . . . . _______       if sequence field desired
   Key length  . . . . . . 8             length from 1 to 9
   Key increment . . . . . 10______      increment value
Copybook or Template:
   Data set name . . . . . 'DELAHAY.FMN1.TEMPLATE'                        ____
   Member  . . . . . . . . MASTER___     (Blank or pattern for member list)
Processing Options:
   Disposition       Copybook or template          Enter "/" to select option
   1  1. Old/Reuse   1  1. Above                   /  Edit template
      2. Mod            2. Previous                _  Batch execution
   ISPF Packing         3. None
   1  1. None           4. Create dynamic
```

Figure 26-2 Specifying target data set and template for data generation

3. If the target data set does not exist, File Manager guides you through its creation as shown in Figure 26-3.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
Allocate SERGEI.FMN1.PDS
Command ===> _____________________________________________________________________

New Data Set Organization:
   Select option          Instructions
   6 1. KSDS              The above data set does not exist.
     2. ESDS              To define or allocate a new data set select a data
     3. RRDS              set organization and press ENTER or press PF3/EXIT
     4. VRRDS             or PF12/CANCEL to return without allocation.
     5. LDS
     6. Non VSAM ------> For a new non VSAM data set only, enter a data set
                          name below to copy existing allocation attributes.


Existing Partitioned or Sequential Data Set:
   Data set name . . . . . 'delahay.fmn2.export'                          ____
   Volume serial . . . . . ______
```

Figure 26-3 Creating new data set using model

4. For a non-VSAM data set, the user can specify a model, as shown in Figure 26-3, so File Manager can use it for allocation attributes as shown in Figure 26-4.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 Allocate SERGEI.FMN1.PDS
 Command ===> ____________________________________________________________________

 Specify a model data set, SMS class names, or leave blank for defaults:
    Like data set . . ____________________________________________
    Volume serial . . ______
    Data class  . . . ________     leave blank for default
    Storage class . . ________     leave blank for default
    Management class  ________     leave blank for default

 Space Requirements:
    Space unit  . . . CYLS         BLKS, TRKS, CYLS, KB, or MB
    Primary units . . 1____        quantity of above units
    Secondary units . 1____        quantity of above units
    Directory blocks  10___        leave blank for SMS default
    Record format . . FB__         if new format: U,F,V, or D, with B,S,A,M
    Record length . . 144__
    Block size  . . . 27936        physical output block size
    Library type  . . _______      LIBRARY, PDS, or blank for default
    Release unused  . NO_          enter YES to free unused space, else NO
    Expiration date . ________     yyyy.ddd, blank for default
```

Figure 26-4 Allocation attributes for target data set

5. File Manager presents an opportunity to select the fields to be included in the target data set, as shown in Figure 26-5. The fields with default values are marked by a character **C** in the SHC column.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 File Manager                Field Selection/Edit                 Line 1 of 9
 Command ===> ___________________________________________________ Scroll PAGE

 1 Id : #2='01'                                                            +
 2 Sel: ___________________________________________________________________ +
 Offset ____0
 Cmd Seq SHC Ref Field Name                          Picture  Type Start Length
                 ****  Top of data  ****
 ___ ___       1 1 REC-TYPE01                                 AN      1     80
 ___ ___   S C 2  2 REC-TYPE                         XX       AN      1      2
 ___ ___       3  2 REC-ID REDEFINES REC-TYPE        XX       AN      1      2
 ___ __1 SHC   4  2 NAME                             X(20)    AN      3     20
 ___ ___   S C 5  2 EMPLOYEE-NO                      9(4)     BI     23      2
 ___ ___   S   6  2 AGE                              9(4)     BI     25      2
 ___ ___   S   7  2 SALARY                           9(7)     PD     27      4
 ___ ___   S   8  2 MONTH OCCURS 12 TIMES            9(8)     BI     31      4
 ___ ___   S   9  2 FILLER                           XX       AN     79      2
                 ****  End of data  ****
```

Figure 26-5 Field selection panel

6. The user can edit field definitions to specify default values for data generation. When done, File Manager displays the confirmation message, as shown in Figure 26-6.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
File Manager                 Data Create Utility          5 record(s) written
Command ===> _____________________________________________________________________
                                                                   More:     +
Output Partitioned, Sequential or VSAM Data Set:
   Data set name . . . . . FMN1.PDS                                 
   Member  . . . . . . . . TEST1___     (Blank or pattern for member list)
   Volume serial . . . . . ______       (If not cataloged)
   Record length . . . . . 144____      Optional record length for RECFM V
   Records . . . . . . . . 5_______     number of records
   Fillchar  . . . . . . . _____        char or hex value, AN, BIN, or RAND
   Key position  . . . . . _______      if sequence field desired
   Key length  . . . . . . 8            length from 1 to 9
   Key increment . . . . . 10______     increment value
Copybook or Template:
   Data set name . . . . . 'DELAHAY.FMN1.TEMPLATE'                  
   Member  . . . . . . . . MASTER___    (Blank or pattern for member list)
Processing Options:
   Disposition       Copybook or template          Enter "/" to select option
   1  1. Old/Reuse   1  1. Above                    _  Edit template
      2. Mod            2. Previous                 _  Batch execution
   ISPF Packing         3. None
   1  1. None           4. Create dynamic
```

Figure 26-6 Data is generated

7. The results are presented in Figure 26-7. The user has to update the generated records with pertinent values, unless the generated values are acceptable.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
File Manager Browse SERGEI.FMN1.PDS(TEST1)
Command ===> ____________________________________________________ Scroll PAGE
                                          Record 0_______ Format TABL
NAME                 REC-TYPE EMPLOYEE-NO     AGE   SALARY    MONTH(1)
#4                   #2               #5      #6       #7          #8
AN 3:20              AN 1:2       BI 23:2 BI 25:2  PD 27:4     BI 31:4
<---+----1----+----> <>           <---+>  <---+> <---+--> <---+----1>
****  Top of data  ****
ABCDEFGHIJKLMNOPQRST 01              1000       0        0           0
BCDEFGHIJKLMNOPQRSTU 01              1010       0        0           0
CDEFGHIJKLMNOPQRSTUV 01              1020       0        0           0
DEFGHIJKLMNOPQRSTUVW 01              1030       0        0           0
EFGHIJKLMNOPQRSTUVWX 01              1040       0        0           0
****  End of data  ****
```

Figure 26-7 Generated data

26.1.2 Data creation in a batch mode

A sample procedure to generate data in a batch mode is presented in Example 26-1. This job will create 5 records in the output data set according to the field definitions in the specified File Manager template (TCOUT parameter).

Example 26-1 Generating data in batch mode

```
//*
//FILEMGR  EXEC PGM=FMNMAIN
//STEPLIB  DD DSN=ADT5.FMN.SFMNMOD1,DISP=SHR
//SYSPRINT DD SYSOUT=*
//FMNTSPRT DD SYSOUT=*
//SYSTERM  DD SYSOUT=*
//SYSIN    DD *
$$FILEM DSG DSNOUT=DELAHAY.FMN1.PDS(TEST1),
$$FILEM TCOUT=DELAHAY.FMN1.TEMPLATE(MASTER),
$$FILEM DISP=OLD,
$$FILEM PACK=NONE,
$$FILEM NLRECS=5
```

26.1.3 Field attributes for template-based data creation

The opportunity to define default values to be used in generated data is available to the user when File Manager presents the Field Attributes panel, shown in Figure 26-8.

Numeric values

For numeric fields, the creation attributes are the start and end values, the increment and the cycle.

```
 Process   Options   Help
 ------------------------------------------------------------------------
File Manager                    Field Attributes
Command ===> ___________________________________________________________

Fixed Attributes:
          Field name  EMPLOYEE-NO
                Type  BI
               Start  23
              Length  2

Use Attributes:
             Heading  ____________________
        Output width  _____
       Leading zeros  NO_          YES or NO

Create Attributes:
         Start value  1000_____________________
           End value  ______________________
           Increment  10_____________________
               Cycle  __________
```

Figure 26-8 Attributes for numeric fields

Table 26-1 contains samples illustrating how the values of these attributes affect the resulting values.

Table 26-1 Attributes for numeric fields

Start	End	Increment	Cycle	Generated values
1,000		10		1000, 1010, 1020, 1030 ...
1,000	1,020	5		1000, 1005, 1010, 1015, 1020, 1020, 1020 ...
1,000		10	3	1000, 1000, 1000, 1010, 1010 ...

Alphanumeric values

For alphanumeric fields, there is a different set of creation attributes, for example, the filler, the action, the start character, the pattern and the pattern repetition indicator, as shown in Figure 26-9.

```
 Process   Options   Help
 ------------------------------------------------------------------------
File Manager                    Field Attributes
Command ===> ___________________________________________________________

Fixed Attributes:
          Field name  REC-TYPE
                Type  AN
               Start  1
              Length  2

Use Attributes:
             Heading  ____________________
        Output width  _____

Create Attributes:
              Filler  _______
              Action  FX
     Start character  _
             Pattern  '01'____________________________________________
 Repeat user pattern  ___          YES or NO
```

Figure 26-9 Attributes for alphanumeric fields

Table 26-2 contains samples illustrating how the values of these attributes affect the resulting values.

Table 26-2 Attributes for alphanumeric fields

Filler	Action	Pattern	Generated values
	FX	'01'	'01' in all records
?			'??????' In all records, filling the field
	FX	AN	'ABCDEF' for all records
	RP	AN	Rippling effect ABCDEF BCDEFG CDEFGH DEFGHI EFGHIJ

The available values for the action field are as follows:

FX The contents of the field are to remain fixed.

RA The contents of the field are to be generated from characters selected at random from the specified pattern. A different set of characters is selected for each record.

RO The contents of the field are to be rolled. The pattern you specify is shifted left one byte for each record until the first non-blank character of the pattern is in the first byte of the field. Then, the pattern is shifted right one byte for each output record until it returns to its original position in the field. RO can only be used with a user-supplied pattern, not with an IBM-supplied pattern. The pattern must start with

a blank, or the result is the same as FX. The roll only occurs within the length of the pattern.

RP The contents of the field are to be rippled. The pattern you specify is shifted left one byte for each record and the truncated character is appended to the right-hand end of the pattern.

SL The contents of the field are to be shifted left for each record. The pattern you specify is shifted left one character and filled to the right with a space. When all characters in the pattern have been shifted out, the original pattern is restored and the process is repeated.

SR The contents of the field are to be shifted right for each record. The pattern you specify is shifted right one character and filled to the left with a space. When all characters in the pattern have been shifted out, the original pattern is restored and the process is repeated.

TL The contents of the field are to be truncated on the left for each record. The pattern you specify is left truncated (the leftmost character replaced with a space) one character at a time until all characters in the pattern have been truncated. Then, the original pattern is restored and the process is repeated.

TR The contents of the field are to be truncated on the right for each record. The pattern you specify is right truncated (the rightmost character replaced with a space) one character at a time until all characters in the pattern have been truncated. Then, the original pattern is restored and the process is repeated.

WV The contents of the field are to be waved. The pattern you specify is shifted left one byte for each record until the first non-blank character of the pattern is in the first byte of the field. Then, the original pattern is restored and the process repeated. WV can only be used with a user-supplied pattern, not with an IBM-supplied pattern. The pattern must start with a blank, or the result is the same as FX. The roll only occurs within the length of the pattern.

The IBM-supplied patterns are:

AL Alphabetic - characters A-Z, inclusive

AN Alphanumeric - characters A-Z, 0-9 inclusive

CO Collating sequence - all printable characters in the range X'40' - X'F9'

26.2 Copying MVS data using File Manager

When copying data using File Manager, the user can:

- Select records and fields using Boolean expressions
- Map fields from the input data set to the output data set
- Reformat and generate data while copying

26.2.1 Data copy on line

Perform the following steps to copy data on line:

1. Select option **3,** Copy data, from the File Manager Utility Functions panel. Fill in the fields about the source data set and the template or copybook to be used, as shown in Figure 26-10.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 File Manager                      Copy Utility
 Command ===> ____________________________________________________________________
                                                                        More:     +
 From Partitioned, Sequential or VSAM Data Set:
    Data set name . . . . . 'ADTOOLS.FILE.MGR50.SFMNSAM1'
    Member  . . . . . . . . FMNCDATA       (Blank or pattern for member list)
    Volume serial . . . . . ______
    Start key . . . . . . . ______________________________ key or slot
    Skip count  . . . . . . 0_______      number of records to be skipped
    Copy count  . . . . . . ALL_____      number of records to be copied
 From Copybook or Template:
    Data set name . . . . . 'SLITVAK.FMN1.TEMPLATE'
    Member  . . . . . . . . ___________     (Blank or pattern for member)
 Processing Options:
    Copybook/template     Enter "/" to select option
    1  1. Above           _  Edit template          _  Advanced member selection
       2. None            _  Batch execution        _  Skip member name list
       3. Create dynamic  _  Use proc _______       _  REXX member selection: P
                          /  Ignore length mismatch _  Exclusive (DISP=OLD)
  F1=Help      F2=Split     F3=Exit      F4=CRetriev  F7=Backward  F8=Forward
  F9=Swap     F10=Actions  F12=Cancel
```

Figure 26-10 Specifying source data set and template for copying

2. Do the same thing for the target data set, as shown in Figure 26-11.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 Copy from ADTOOLS.FILE.MGR50.SFMNSAM1(FMNCDATA)
 Command ===> ____________________________________________________________________
                                                                        More:     +
 To Partitioned, Sequential or VSAM Data Set:
    Data set name . . . . . 'SLITVAK.FMN1.DATA'
    Member name (or mask) . FMNCDAT1
    Volume serial . . . . . ______

 To Copybook/Template From:  SLITVAK.FMN1.TEMPLATE(FMNCCPY)
    Data set name . . . . . 'SLITVAK.FMN1.TEMPLATE'
    Member  . . . . . . . . COPY______      (Blank or pattern for member)

 Processing Options:
   Copybook/template usage     Disposition           Enter "/" to select option
   1  1. Above                 1  1. Old or Reuse    _  Replace members
      2. None                    2. Mod              /  Edit template mapping
      3. Create dynamic
   ISPF Packing
   1  1. Asis
      2. Pack
      3. Unpack
```

Figure 26-11 Specifying target data set and template for copying

3. Edit the templates if there is a requirement to specify selection criteria; edit the template mapping if there are differences between source and target formats as shown in Figure 26-12.

```
 Process   Options   Help
File Manager                  Record Type Mapping                  Line 1 of 2
Command ===> ____________________________________________________ Scroll PAGE

To          SLITVAK.FMN1.TEMPLATE(COPY)
From        SLITVAK.FMN1.TEMPLATE(FMNCCPY)

Cmd To Field Name                 Len   From Field Name                     Len
    ****  Top of data  ****
s__ REC-TYPE01                     88 | REC-TYPE01                          80
___ REC-TYPE02                     80 | REC-TYPE02                          80
    ****  End of data  ****
```

Figure 26-12 Record type mapping panel

4. The user can change the correspondence between fields and edit new fields to specify default values for data creation as shown in Figure 26-13.

```
 Process   Options   Help
File Manager                  Field Mapping                        Line 1 of 9
Command ===> ____________________________________________________ Scroll PAGE

To          SLITVAK.FMN1.TEMPLATE(COPY)
From        SLITVAK.FMN1.TEMPLATE(FMNCCPY)

Cmd Lv To Field Name       Type  Len   Lv From Field Name             Type  Len
       ****  Top of data  ****
___   1 REC-TYPE01          AN    88 |  1 REC-TYPE01                   AN    80
___   2 REC-TYPE            AN     2 |  2 REC-TYPE                     AN     2
___   2 NAME                AN    20 |  2 NAME                         AN    20
s__   2 EMPLOYEE-NO         BI     2 |  2 EMPLOYEE-NO                  BI     2
___   2 AGE                 BI     2 |  2 AGE                          BI     2
e__   2 SERIAL-NBR          ZD     8 |
___   2 SALARY              PD     4 |  2 SALARY                       PD     4
___   2 MONTH               BI     4 |  2 MONTH                        BI     4
___   2 FILLER              AN     2 |  2 FILLER                       AN     2
       ****  End of data  ****
```

Figure 26-13 Field mapping panel

5. The user can specify a specific relationship between source and target fields, as shown in Figure 26-14.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
File Manager                  From Field Mapping
Command ===> ___________________________________________________________ Scroll PAGE
To   template/copybook : SLITVAK.FMN1.TEMPLATE(COPY)
From template/copybook : SLITVAK.FMN1.TEMPLATE(FMNCCPY)
To field . . . . . . . : #4 EMPLOYEE-NO
From field . . . . . . : #5 EMPLOYEE-NO
Corresponding(Y/N) . . : N               (Auto map for group items).

Sel   Ref Lvl From Field                                      Typ Start Length
              ****  Top of data  ****
___     D --  Delete "From field"
___     1  1  REC-TYPE01                                      AN      1     80
___     2  2  REC-TYPE                                        AN      1      2
___     3  2  REC-ID                                          AN      1      2
___     4  2  NAME                                            AN      3     20
s__     5  2  EMPLOYEE-NO                                     BI     23      2
___     6  2  AGE                                             BI     25      2
___     7  2  SALARY                                          PD     27      4
___     8  2  MONTH(12)                                       BI     31      4
___     9  2  FILLER                                          AN     79      2
              ****  End of data  ****
```

Figure 26-14 Source and target fields mapping

6. Creation attributes for a target field can be specified, as shown in Figure 26-15.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
File Manager                   Field Attributes
Command ===> ___________________________________________________________________

Fixed Attributes:
          Field name  SERIAL-NBR
                Type  ZD
               Start  27
              Length  8

Use Attributes:
             Heading  ____________________
        Output width  _____
       Leading zeros  NO_          YES or NO

Create Attributes:
         Start value  100000_________________
           End value  ____________________
           Increment  10____________________
               Cycle  __________
```

Figure 26-15 Creation attributes specified for target field

7. File Manager displays the confirmation message when execution is finished, as shown in Figure 26-16.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 File Manager                        Copy Utility
 Command ===> _______________________________________________________________

 From Partitioned, Sequential or VSAM Data Set:
    Data set name . . . . . 'ADTOOLS.FILE.MGR50.SFMNSAM1'
    Member  . . . . . . . . FMNCDATA       (Blank or pattern for member list)
    Volume serial . . . . . ______
    Start key . . . . . . . ______________________________ key or slot
    Skip count  . . . . . . 0_______      number of records to be skipped
    Copy count  . . . . . . ALL_____      number of records to be copied
 From Copybook or Template:
    Data set name . . . . . 'SLITVAK.FMN1.TEMPLATE'
    Member  . . . . . . . . FMNCCPY___      (Blank or pattern for member)
 Processing Options:
    Copybook/template     Enter "/" to select option
    1  1. Above           _  Edit template          _  Advanced member selection
       2. None            _  Batch execution        _  Skip member name list
       3.                                                          lection: P
          | 40 record(s) copied: 20 truncated: 0 fields truncated |  P=OLD)
                                                                     ord counts
```

Figure 26-16 File Manager confirmation message

26.2.2 Data copy in a batch mode

The user can copy data using File Manager also in a batch mode. In this case the user can concatenate several steps.

Example 26-2 shows how to process SMF records with LRECL=32767 that File Manager cannot process in its present level. This example takes advantage of using DFSORT. This job will copy in the data set DD01O only the records containing SYS0.

Example 26-2 File Manager batch extraction

```
//*********************************************************************
//* This step is here to truncate SMF records to an acceptable length
//*********************************************************************
//FILTEMP EXEC  PGM=SORT
//SYSPRINT DD SYSOUT=*
//SYSMSG   DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SORTIN   DD DISP=SHR,DSN=INPUT.SMF
//SORTOUT  DD DISP=(,PASS),DSN=&&SORT01X,
//            DCB=(BUFNO=50),UNIT=SORT,
//            SPACE=(CYL,(50,30),RLSE),
//            BLKSIZE=32760,RECFM=VB,LRECL=32756
//SYSIN    DD *
OPTION COPY
OUTFIL FNAMES=SORTOUT
/*
//*********************************************************************
//* File Manager processing of SMF truncated records
//*********************************************************************
//FILTEMP1 EXEC  PGM=FILEMGR
//SYSPRINT DD SYSOUT=*
//DD01     DD DISP=SHR,DSN=&&SORT01X
//DD01O    DD DISP=(,CATLG,DELETE),DSN=CONVERTED.SMF,
//            DCB=(BUFNO=50),UNIT=SYSDA,
//            SPACE=(CYL,(50,30),RLSE),
//            BLKSIZE=32760,RECFM=VB,LRECL=32756
```

```
//DDUMMY   DD SYSOUT=*
//SYSIN    DD *
$$FILEM DSC  INPUT=DD01,OUTPUT=DDUMMY,
$$FILEM      PROC=*
IF CO(INREC,'SYS0') THEN WRITE(DD010)
RETURN DROP
/+
//*******************************************************************
```

If you prefer DFSORT programming instead of File Manager, do not forget that the user can use templates with File Manager, but cannot use them with DFSORT.

Table 26-3 shows the SYSIN contents for the same selection when using File Manager and DFSORT.

Table 26-3 Selection criteria sample in File Manager and DFSORT

File Manager	DFSORT

`IF SUBSTR(INREC,74,1) = '-' &`
`  SUBSTR(INREC,31,1) = 'E' &`
`  SUBSTR(INREC,40,2) <> 'R0'`
`  THEN WRITE(DD010)`
	`OUTFIL FNAMES=DD010,`
	`  INCLUDE=(74,1,CH,EQ,C'-',`
	`  AND,31,1,CH,EQ,C'E',`
	`  AND,40,2,CH,NE,C'R0')`

Tip: The best way to avoid having to use DFSORT to truncate the SMF records is to specify the records' length when dumping them, as shown in Example 26-3.

Example 26-3 SMF dump with fixed LRECL

```
//SMFDUMP EXEC PGM=IFASMFDMP
//INDD1     DD DISP=SHR,DSN=your_smf_save_file
//OUTDD1    DD DISP=(NEW,CATLG),UNIT=SYSDA,
//             SPACE=(CYL,10,5),VOL=SER=your_disk,
//             LRECL=32760,RECFM=VBS,BLKSIZE=0,
//             DSN=your_smf_working_file
//SYSPRINT  DD SYSOUT=*
//SYSIN     DD *
INDD(INDD1,OPTIONS(DUMP))
OUTDD(OUTDD1,TYPE(110))
/*
```

26.3 Creating DB2 test data using File Manager

The user can create DB2 data using the File Manager DB2 Component referred to here as File Manager/DB2.

26.3.1 DB2 data creation on line

Perform the following steps to create DB2 data on line:

1. From the File Manager/DB2 Utility Functions panel shown in Figure 26-17 select option **8**, which is specifically meant for test data creation.

```
 Process   Options   Utilities   Help
FM/DB2 (DB0A)                 Utility Functions
Command ===> _____________________________________________________

1  Print         Print DB2 table or view
2  Objects       Create and drop DB2 objects
3  Copy          Copy data within DB2
4  Object List   Display and process DB2 object lists
5  Privileges    Manage DB2 privileges
6  Import        Import sequential or VSAM data into DB2
7  Export        Export DB2 data to sequential or VSAM data set
8  Create        Create DB2 test data
9  Utilities     DB2 utility job generation
10 Audit trail   Print audit trail report
11 Print browse  Browse FM/DB2 print data set
```

Figure 26-17 Utility Functions panel of File Manager/DB2

2. On the next panel, shown in Figure 26-18, specify the target DB2 object. This target object must exist prior to data generation; there is no option to create it presented at this point.

```
 Process   Options   Utilities   Help
FM/DB2 (DB0A)               Data Create Utility
Command ===> _____________________________________________________

Specify the DB2 Object:
   Location  . . . . . _______________        Database . . ________ (optional)
   Owner . . . . . . . DELAHAY_____           Table space  ________ (optional)
   Name  . . . . . . . TOTO_______________

   Create Count  . . . 6_______       Number of rows to create

DB2 Template:
   Data set name . . . ____________________________________________
   Member  . . . . . . ________

Processing Options:
   Template usage                     Enter "/", "A"lways to select option
   3  1. Above                        _  Edit template
      2. Previous                     _  Batch execution
      3. Generate from table
      4. Generate/Replace
```

Figure 26-18 Data create utility panel of File Manager/DB2

3. The user can edit the corresponding template, as shown in Figure 26-19.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------------
FM/DB2 (DB0A)                Column Selection/Edit                  Line 1 of 14
Command ===> _____________________________________________________ Scroll PAGE

------- Row Selection Criteria ---- (Use SQL/PF4 for full screen edit) -------
1 Sel: ____________________________________________________________________ +

Cmd Seq SHC CL# Column name          Data type(length)  Null Default  Order A/D
                ****  Top of data  ****
___ ___ S     1 EMPNO                CHARACTER(6)            None
___ ___ S     2 FIRSTNME             VARCHAR(12)             None
___ ___ S     3 MIDINIT              CHARACTER(1)            None
___ ___ S     4 LASTNAME             VARCHAR(15)             None
___ ___ S     5 WORKDEPT             CHARACTER(3)        Y   Null
___ ___ S     6 PHONENO              CHARACTER(4)        Y   Null
___ ___ S     7 HIREDATE             DATE                Y   Null
___ ___ S     8 JOB                  CHARACTER(8)        Y   Null
___ ___ S     9 EDLEVEL              SMALLINT            Y   Null
___ ___ S    10 SEX                  CHARACTER(1)        Y   Null
___ ___ S    11 BIRTHDATE            DATE                Y   Null
___ ___ S    12 SALARY               DECIMAL(9,2)        Y   Null
___ ___ S    13 BONUS                DECIMAL(9,2)        Y   Null
___ ___ S    14 COMM                 DECIMAL(9,2)        Y   Null
```

Figure 26-19 Column selection/edit panel

4. The user can specify creation criteria for the fields. Column attributes for a character field are shown in Figure 26-20.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------------
FM/DB2 (DB0A)                Column Attributes
Command ===> ______________________________________________________________

Fixed Attributes:
         Column name  WORKDEPT
                Type  CHARACTER(3)

Use Attributes:
            Heading  ____________________
       Output width  _____

Create Attributes:
             Filler  _______
             Action  __
     Start character  _
            Pattern  ____________________________________________________
 Repeat user pattern  ___          YES or NO
```

Figure 26-20 Column attributes for character field

5. Column attributes for a numeric field are shown in Figure 26-21.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------
FM/DB2 (DB0A)                  Column Attributes
Command ===> ____________________________________________________________

Fixed Attributes:
         Column name  EDLEVEL
                Type  SMALLINT

Use Attributes:
              Heading  ____________________
         Output width  ____
        Leading zeros  NO_           YES or NO

Create Attributes:
          Start value  ________________________
            End value  ________________________
            Increment  ________________________
                Cycle  __________
```

Figure 26-21 Column attributes for numeric field

6. Figure 26-22 shows the column attributes for a date field, which is specific for File Manager/DB2.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------
FM/DB2 (DB0A)                  Column Attributes
Command ===> ____________________________________________________________

Fixed Attributes:
         Column name  HIREDATE
                Type  DATE
              Format  yyyy-mm-dd

Use Attributes:
              Heading  ____________________
         Output width  ____

Create Attributes:
           Date value  ____________________________________________
                             (Use = for date now, * for runtime date)
            Increment  0_________
```

Figure 26-22 Column attributes for date field

Note: The date field is specific to the DB2 component. It does not exist in the MVS component.

7. A fragment of the result table is shown in Figure 26-23.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------------
FM/DB2 (DB0A)                      Table Browse
Command ===> _____________________________________________________ Scroll PAGE
TABLE DELAHAY.TOTO                                                Format TABL
EMPNO  FIRSTNME     MIDINIT LASTNAME        WORKDEPT PHONENO HIREDATE   JOB
#1     #2           #3      #4              #5       #6      #7         #8   +
CH(6)  VARCHAR(12)  CH(1)   VARCHAR(15)      CH(3)    CH(4)   DATE       CH(8)
<---+> <---+----1-> -       <---+----1----> <->      <-->    <---+----> <---+-
****  Top of data  ****
       <                    <               _        _       _          _
       <                    <               _        _       _          _
       <                    <               _        _       _          _
       <                    <               _        _       _          _
       <                    <               _        _       _          _
       <                    <               _        _       _          _
       <                    <               _        _       _          _
       <                    <               _        _       _          _
       <                    <               _        _       _          _
       <                    <               _        _       _          _
       <                    <               _        _       _          _
       <                    <               _        _       _          _
       <                    <               _        _       _          _
       <                    <               _        _       _          _
```

Figure 26-23 Generated rows

26.3.2 DB2 data creation in batch mode

DB2 data can be created using File Manager/DB2 also in a batch mode. The sample procedure is presented in Example 26-4. This job will fill the table TOTO with 6 records according to the column definitions from the catalog.

Example 26-4 Sample procedure to generate DB2 data

```
//FMNDB2   EXEC PGM=FMNDB2,PARM=('SSID=DB0A,SQID=DELAHAY')
//STEPLIB  DD DSN=DB2CUST0.DB0A.SDSNEXIT,DISP=SHR
//         DD DSN=DSN710.SDSNLOAD,DISP=SHR
//         DD DSN=DSNDB0A.RUNLIB.LOAD,DISP=SHR
//SYSPRINT DD SYSOUT=*
//FMNTSPRT DD SYSOUT=*
//SYSTERM  DD SYSOUT=*
//SYSIN    DD *
$$FILEM D2G OBJOUT="DELAHAY"."TOTO",
$$FILEM ROWS=6
```

26.4 Copying DB2 data using File Manager

When copying DB data, the user can:

- Select rows
- Map columns in a source DB2 table to columns in a target DB2 table
- Reformat and generate data while copying

26.4.1 DB2 data copy on line

Perform the following steps to copy DB2 data on line:

1. Select option **3,** Copy data within DB2, from the File Manager/DB2 Utility Functions panel. Fill in the fields about the source object and the template to be used, as shown in Figure 26-24.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------
FM/DB2 (DB0A)                      Copy Utility
Command ===> ____________________________________________________________

From DB2 Object:
   Location  . . . . . ________________        Database . . ________ (optional)
   Owner . . . . . . . fmndb2__                Table space  ________ (optional)
   Name  . . . . . . . emp________________

   Copy count  . . . . ALL_____       Number of rows to copy

From Template:
   Data set name . . . fmn2.template_________________________________
   Member  . . . . . . emp_____

Processing Options:
   Template usage                     Enter "/", "A"lways to select option
   3  1. Above                        _  Edit template
      2. Previous
      3. Generate from table
      4. Generate/Replace
```

Figure 26-24 Source DB2 object and template

2. Do the same thing for the target object, as shown in Figure 26-25.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------
Copy from FMNDB2.EMP
Command ===> ____________________________________________________________

To DB2 Object:
   Location  . . . . . ________________        Database . . ________ (optional)
   Owner . . . . . . . DELAHAY_                Table space  ________ (optional)
   Name  . . . . . . . EMP________________



To Template:
   Data set name . . . ___________________________________________
   Member  . . . . . . ________

Processing Options:
   Template usage                     Enter "/", "A"lways to select option
   3  1. Above                        _  Edit copy options
      2. Previous                     /  Edit template mapping
      3. Generate from table          _  Batch execution
      4. Generate/Replace
```

Figure 26-25 Target DB2 object and template

3. If required, the user can edit the templates to specify selection criteria and edit the template mapping if there are differences between source and target formats as shown in Figure 26-26.

```
 Process   Options   Help
FM/DB2 (DB0A)                  Template Mapping                    Line 1 of 14
Command ===> ____________________________________________________ Scroll PAGE

To          DELAHAY.EMP
From        FMNDB2.EMP

Cmd CL# To Column name  Datatype  NI   CL# From Column name        Datatype  NI
        ****  Top of data  ****
___   1 EMPNO           CH(6)     NN |   1 EMPNO                   CH(6)     NN
___   2 FIRSTNME        VC(12)    NN |   2 FIRSTNME                VC(12)    NN
S__   3 MIDINIT         CH(1)     NN |   3 MIDINIT                 CH(1)     NN
___   4 LASTNAME        VC(15)    NN |   4 LASTNAME                VC(15)    NN
E__   5 WORKDEPT        CH(3)        |   5 WORKDEPT                CH(3)
___   6 PHONENO         CH(4)        |   6 PHONENO                 CH(4)
___   7 HIREDATE        DATE         |   7 HIREDATE                DATE
___   8 JOB             CH(8)        |   8 JOB                     CH(8)
___   9 EDLEVEL         SMINT        |   9 EDLEVEL                 SMINT
___  10 SEX             CH(1)        |  10 SEX                     CH(1)
___  11 BIRTHDATE       DATE         |  11 BIRTHDATE               DATE
___  12 SALARY          DEC(9,2)     |  12 SALARY                  DEC(9,2)
___  13 BONUS           DEC(9,2)     |  13 BONUS                   DEC(9,2)
___  14 COMM            DEC(9,2)     |  14 COMM                    DEC(9,2)
```

Figure 26-26 Template mapping panel of File Manager/DB2

4. The user can also change the correspondence between fields and edit new fields to specify default values for data creation.

   A sample of a specific relationship between fields is shown in Figure 26-27.

```
 Process   Options   Utilities   Help
FM/DB2 (DB0A)                 From Column Mapping
Command ===> ____________________________________________ Scroll PAGE
To DB2 object . . : DELAHAY.EMP
From DB2 object . : FMNDB2.EMP
To column . . . . : #3 MIDINIT
From column . . . : #3 MIDINIT

Sel   Cl#  From Column            Data type(length)
           ****  Top of data  ****
___     D  Delete "From column"
___     1  EMPNO                 CHARACTER(6)
___     2  FIRSTNME              VARCHAR(12)
S__     3  MIDINIT               CHARACTER(1)
___     4  LASTNAME              VARCHAR(15)
___     5  WORKDEPT              CHARACTER(3)
___     6  PHONENO               CHARACTER(4)
___     7  HIREDATE              DATE
___     8  JOB                   CHARACTER(8)
___     9  EDLEVEL               SMALLINT
___    10  SEX                   CHARACTER(1)
___    11  BIRTHDATE             DATE
___    12  SALARY                DECIMAL(9,2)
```

Figure 26-27 File Manager/DB2 column mapping panel

5. The File Manager/DB2 Column Attributes panel where creation attributes can be defined is shown in Figure 26-28.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------------
FM/DB2 (DB0A)                  Column Attributes
Command ===> ____________________________________________________________________

Fixed Attributes:
         Column name  WORKDEPT
                Type  CHARACTER(3)

Use Attributes:
              Heading  ____________________
         Output width  _____

Create Attributes:
               Filler  _______
               Action  __
     Start character  _
              Pattern  ____________________________________________________
 Repeat user pattern  ___           YES or NO
```

Figure 26-28 File Manager/DB2 column attributes panel

6. File Manager/DB2 displays the confirmation message when data is copied, as shown in Figure 26-29.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------------
FM/DB2 (DB0A)                  Copy Utility                 33 row(s) copied
Command ===> ____________________________________________________________________

From DB2 Object:
   Location  . . . . . ______________        Database . . ________ (optional)
   Owner . . . . . . . FMNDB2__              Table space  ________ (optional)
   Name  . . . . . . . EMP______________

   Copy count  . . . . ALL_____        Number of rows to copy

From Template:
   Data set name . . . FMN2.TEMPLATE_____________________________________
   Member  . . . . . . EMP_____

Processing Options:
   Template usage                      Enter "/", "A"lways to select option
   3  1. Above                         _  Edit template
      2. Previous
      3. Generate from table
      4. Generate/Replace
```

Figure 26-29 File Manager/DB2 confirmation message

This process of copying DB2 data using File Manager/DB2 on line is very similar to the process of copying data on line using File Manager.

26.4.2 DB2 data copy in a batch mode

DB2 data can be copied to another DB2 object using File Manager/DB2 also in a batch mode. The sample procedure is presented in Example 26-5. This job will copy all records from one table to the other according to the column definitions in the catalog with automatic mapping by field name.

Example 26-5 Sample procedure to copy DB2 data

```
//FMNDB2   EXEC PGM=FMNDB2,PARM=('SSID=DB0A,SQID=DELAHAY')
//STEPLIB  DD DSN=DB2CUST0.DB0A.SDSNEXIT,DISP=SHR
//         DD DSN=DSN710.SDSNLOAD,DISP=SHR
//         DD DSN=DSNDB0A.RUNLIB.LOAD,DISP=SHR
//SYSPRINT DD SYSOUT=*
//FMNTSPRT DD SYSOUT=*
//SYSTERM  DD SYSOUT=*
//SYSIN    DD *
$$FILEM DBC OBJIN="FMNDB2"."EMP",
$$FILEM     OBJOUT="DELAHAY"."EMP",
$$FILEM     ROWS=ALL
/*
```

26.5 Importing and exporting DB2 data using File Manager

The user can:

- Select rows
- Export to a data set in:
  - The default File Manager/DB2 format
  - A format defined by a copybook
  - A comma-delimited format
- Import from a file:
  - Generated by the Export function
  - Described by a copybook

26.5.1 DB2 data export on line

Perform the following steps to export data from DB2 to sequential or VSAM data sets:

1. Select option **7** on the Utility Functions panel of File Manager/DB2, as shown in Figure 26-30.

```
 Process   Options   Utilities   Help
FM/DB2 (DB0A)                Utility Functions
Command ===> 7

1  Print          Print DB2 table or view
2  Objects        Create and drop DB2 objects
3  Copy           Copy data within DB2
4  Object List    Display and process DB2 object lists
5  Privileges     Manage DB2 privileges
6  Import         Import sequential or VSAM data into DB2
7  Export         Export DB2 data to sequential or VSAM data set
8  Create         Create DB2 test data
9  Utilities      DB2 utility job generation
10 Audit trail    Print audit trail report
11 Print browse   Browse FM/DB2 print data set
```

Figure 26-30 File Manager/DB2 Utility Functions panel - Option 7

2. Fill in the fields about the source object and the template to be used, as shown in Figure 26-31.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------
FM/DB2 (DB0A)                    Export Utility        Object not found (DB0A)
Command ===> ____________________________________________________________

From DB2 Object:
   Location  . . . . . ________________        Database . . ________ (optional)
   Owner . . . . . . . DELAHAY                 Table space  ________ (optional)
   Name  . . . . . . . EMP_________________

   Export Count  . . . ALL_____       Number of rows to export

From Template:
   Data set name . . . FMN2.TEMPLATE_________________________________
   Member  . . . . . . EMP_______

Processing Options:
   Template usage                     Enter "/", "A"lways to select option
   3  1. Above                        _  Edit template
      2. Previous
      3. Generate from table
      4. Generate/Replace
```

Figure 26-31 Source DB2 object and template on File Manager/DB2 export utility panel

3. Do the same thing for the target object, as shown in Figure 26-32.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------
Export from DELAHAY.EMP
Command ===> ____________________________________________________________

To Partitioned, Sequential or VSAM Data Set:
   Data set name . . . FMN2.EXPORT___________________________________
   Member  . . . . . . EMP_____
   Volume  . . . . . . ______

To Copybook or Template:
   Data set name . . . FMN1.TEMPLATE_________________________________
   Member  . . . . . . EMPX_____

Processing Options:
   Template usage            Disposition           Enter "/", "A"lways
   1  1. Above                1 1. Old or Reuse    /  Edit template mapping
      2. Previous               2. Mod             _  Batch execution
      3. Generate from input                       _  Batch data set creation
      4. Generate and save
      5. None. (CSV output)
```

Figure 26-32 Target DB2 object and template definition for data export

4. If required, the user can edit the templates to specify selection criteria and edit the template mapping if there are differences between source and target formats, as shown in Figure 26-33.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
FM/DB2 (DB0A)                  Template Mapping                     Line 1 of 15
Command ===> ____________________________________________________________ Scroll CSR

To          SERGEI.FMN2.EXPORT(EMP)
From        DELAHAY.EMP

Cmd Lv To Field Name        Type  Len   CL# From Column name        Datatype  NI
___  2 EMPNO                 AN     6 |   1 EMPNO                    CH(6)     NN
___  2 FIRSTNME              AN    12 |   2 FIRSTNME                 VC(12)    NN
___  2 MIDINIT               AN     1 |   3 MIDINIT                  CH(1)     NN
S__  2 LASTNAME              AN    15 |   4 LASTNAME                 VC(15)    NN
___  2 WORKDEPT              AN     3 |   5 WORKDEPT                 CH(3)
___  2 PHONENO               AN     4 |   6 PHONENO                  CH(4)
___  2 HIREDATE              AN    10 |   7 HIREDATE                 DATE
___  2 JOB                   AN     8 |   8 JOB                      CH(8)
___  2 EDLEVEL               BI     2 |   9 EDLEVEL                  SMINT
___  2 SEX                   AN     1 |  10 SEX                      CH(1)
___  2 BIRTHDATE             AN    10 |  11 BIRTHDATE                DATE
___  2 SALARY                PD     7 |  12 SALARY                   DEC(9,2)
___  2 BONUS                 PD     7 |  13 BONUS                    DEC(9,2)
___  2 COMM                  PD     7 |  14 COMM                     DEC(9,2)
E__  2 FILLER                AN     8 |
```

Figure 26-33 Template mapping when exporting DB2 data

5. The user can change the correspondence between fields and edit new fields to specify default values for data creation.

   A specific relationship between fields is defined using the File Manager/DB2 Column mapping panel, shown in Figure 26-34.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------------
FM/DB2 (DB0A)                 From Column Mapping
Command ===> ____________________________________________________________ Scroll CSR
To template/copybook  : SERGEI.FMN2.EXPORT(EMP)
From DB2 object . . . : DELAHAY.EMP
To field  . . . . . . : #5 LASTNAME
From column . . . . . : #4 LASTNAME

Sel   Cl#  From Column            Data type(length)
           ****  Top of data  ****
___     D  Delete "From column"
___     1  EMPNO                  CHARACTER(6)
___     2  FIRSTNME               VARCHAR(12)
___     3  MIDINIT                CHARACTER(1)
S__     4  LASTNAME               VARCHAR(15)
___     5  WORKDEPT               CHARACTER(3)
___     6  PHONENO                CHARACTER(4)
___     7  HIREDATE               DATE
___     8  JOB                    CHARACTER(8)
___     9  EDLEVEL                SMALLINT
___    10  SEX                    CHARACTER(1)
___    11  BIRTHDATE              DATE
___    12  SALARY                 DECIMAL(9,2)
```

Figure 26-34 Column mapping panel

6. Creation attributes for columns can be specified on the Column Attributes panel, as shown in Figure 26-35.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------------
FM/DB2 (DB0A)                  Column Attributes
Command ===> ____________________________________________________________________

Fixed Attributes:
         Column name  FILLER
                Type  AN

Use Attributes:
              Heading  ____________________
         Output width  _____

Create Attributes:
               Filler  X'00'__
               Action  __
     Start character   _
              Pattern  ____________________________________________________
 Repeat user pattern   ___            YES or NO
```

Figure 26-35 Data creation attributes for an alphanumeric column

7. When data is exported, File Manager displays the confirmation message as shown in Figure 26-36.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------------
FM/DB2 (DB0A)                 Export Utility                 32 rows exported
Command ===> ____________________________________________________________________

From DB2 Object:
   Location  . . . . . ________________        Database . . ________ (optional)
   Owner . . . . . . . DELAHAY_            Table space  ________ (optional)
   Name  . . . . . . . EMP_________________

   Export Count  . . . ALL_____       Number of rows to export

From Template:
   Data set name . . . 'DELAHAY.FMN2.TEMPLATE'____________________________
   Member  . . . . . . EMP_______

Processing Options:
   Template usage                     Enter "/", "A"lways to select option
   3  1. Above                        _  Edit template
      2. Previous
      3. Generate from table
      4. Generate/Replace
```

Figure 26-36 DB2 data is exported successfully

8. The fragment of the source DB2 table is shown in Figure 26-37.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------------
FM/DB2 (DB0A)                     Table Browse          FM/DB2 template loaded
Command ===> ___________________________________________________________ Scroll CSR
TABLE DELAHAY.EMP                                                     Format TABL
EMPNO  FIRSTNME     MIDINIT LASTNAME        WORKDEPT PHONENO HIREDATE   JOB
#1     #2           #3      #4              #5       #6      #7         #8   +
CH(6)  VARCHAR(12)  CH(1)   VARCHAR(15)      CH(3)    CH(4)   DATE       CH(8)
PU--+> <---+----1-> -       <---+----1----> <-NF     <-->    <---+----> <---+-
****  Top of data  ****
000010 CHRISTINE<   I       HAAS<           A00      3978    1965-01-01 PRES
000110 VINCENZO<    G       LUCCHESI<       A00      3490    1958-05-16 SALESR
000120 SEAN<                O'CONNELL<      A00      2167    1963-12-05 CLERK
000020 MICHAEL<     L       THOMPSON<       B01      3476    1973-10-10 MANAGE
000030 SALLY<       A       KWAN<           C01      4738    1975-04-05 MANAGE
000130 DOLORES<     M       QUINTANA<       C01      4578    1971-07-28 ANALYS
000140 HEATHER<     A       NICHOLLS<       C01      1793    1976-12-15 ANALYS
000060 IRVING<      F       STERN<          D11      6423    1973-09-14 MANAGE
000150 BRUCE<               ADAMSON<        D11      4510    1972-02-12 DESIGN
000160 ELIZABETH<   R       PIANKA<         D11      3782    1977-10-11 DESIGN
000170 MASATOSHI<   J       YOSHIMURA<      D11      2890    1978-09-15 DESIGN
000180 MARILYN<     S       SCOUTTEN<       D11      1682    1973-07-07 DESIGN
000190 JAMES<       H       WALKER<         D11      2986    1974-07-26 DESIGN
000200 DAVID<               BROWN<          D11      4501    1966-03-03 DESIGN
```

Figure 26-37 Fragment of source DB2 table

9. The fragments of the resulting sequential data set is shown in Figure 26-38.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
File Manager Browse SERGEI.FMN2.EXPORT(EMP)
Command ===> ___________________________________________________________ Scroll CSR
                                                   Record 0_____ Format TABL
EMPNO  FIRSTNME     MIDINIT LASTNAME        WORKDEPT PHONENO HIREDATE   JOB
#2     #3           #4      #5              #6       #7      #8         #9   +
AN 1:6 AN 7:12      AN 19:1 AN 20:15        AN 35:3  AN 38:4 AN 42:10   AN 52:
<---+> <---+----1-> -       <---+----1----> <->      <-->    <---+----> <---+-
****  Top of data  ****
000010 CHRISTINE    I       HAAS            A00      3978    1965-01-01 PRES
000110 VINCENZO     G       LUCCHESI        A00      3490    1958-05-16 SALESR
000120 SEAN                 O'CONNELL       A00      2167    1963-12-05 CLERK
000020 MICHAEL      L       THOMPSON        B01      3476    1973-10-10 MANAGE
000030 SALLY        A       KWAN            C01      4738    1975-04-05 MANAGE
000130 DOLORES      M       QUINTANA        C01      4578    1971-07-28 ANALYS
000140 HEATHER      A       NICHOLLS        C01      1793    1976-12-15 ANALYS
000060 IRVING       F       STERN           D11      6423    1973-09-14 MANAGE
000150 BRUCE                ADAMSON         D11      4510    1972-02-12 DESIGN
000160 ELIZABETH    R       PIANKA          D11      3782    1977-10-11 DESIGN
000170 MASATOSHI    J       YOSHIMURA       D11      2890    1978-09-15 DESIGN
000180 MARILYN      S       SCOUTTEN        D11      1682    1973-07-07 DESIGN
000190 JAMES        H       WALKER          D11      2986    1974-07-26 DESIGN
000200 DAVID                BROWN           D11      4501    1966-03-03 DESIGN
```

Figure 26-38 Fragment of exported DB2 data (Part 1 of 2)

10. Press PF11 to scroll right. The field with default value X'00' is added, as shown in Figure 26-39.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 File Manager Browse SERGEI.FMN2.EXPORT(EMP)
 Command ===> _____________________________________________________ Scroll CSR
                                                   Record 0       Format TABL
 SEX      BIRTHDATE            SALARY            BONUS             COMM FILLER
 #11      #12                     #13              #14              #15 #16
 AN 62:1  AN 63:10            PD 73:7          PD 80:7          PD 87:7 AN 94:8
 -        <---+----> <---+----1----> <---+----1----> <---+----1----> <---+-->
 ****  Top of data  ****
 F        1933-08-14         52750.00         1000.00          4220.00 ........
 M        1929-11-05         46500.00          900.00          3720.00 ........
 M        1942-10-18         29250.00          600.00          2340.00 ........
 M        1948-02-02         41250.00          800.00          3300.00 ........
 F        1941-05-11         38250.00          800.00          3060.00 ........
 F        1925-09-15         23800.00          500.00          1904.00 ........
 F        1946-01-19         28420.00          600.00          2274.00 ........
 M        1945-07-07         32250.00          600.00          2580.00 ........
 M        1947-05-17         25280.00          500.00          2022.00 ........
 F        1955-04-12         22250.00          400.00          1780.00 ........
 M        1951-01-05         24680.00          500.00          1974.00 ........
 F        1949-02-21         21340.00          500.00          1707.00 ........
 M        1952-06-25         20450.00          400.00          1636.00 ........
 M        1941-05-29         27740.00          600.00          2217.00 ........
```

Figure 26-39 Fragment of exported DB2 data (Part 2 of 2)

26.5.2 DB2 data export in a batch mode

The DB2 data export procedure using File Manager/DB2 in batch mode is presented in Example 26-6. This job will export data from table EMP to an output PDS member according to an MVS template.

Example 26-6 Sample procedure to export DB2 data

```
//FMNDB2   EXEC PGM=FMNDB2,PARM=('SSID=DB0A,SQID=DELAHAY')
//STEPLIB  DD DSN=DB2CUST0.DB0A.SDSNEXIT,DISP=SHR
//         DD DSN=DSN710.SDSNLOAD,DISP=SHR
//         DD DSN=DSNDB0A.RUNLIB.LOAD,DISP=SHR
//SYSPRINT DD SYSOUT=*
//FMNTSPRT DD SYSOUT=*
//SYSTERM  DD SYSOUT=*
//FMNOUT   DD DISP=SHR,
//            DSN=DELAHAY.FMN2.EXPORT(EMP2)
//SYSIN    DD *
$$FILEM DBX OBJIN="DELAHAY"."EMP",
$$FILEM OUTPUT=FMNOUT,
$$FILEM TCOUT=DELAHAY.FMN.DATA(EMPX),
$$FILEM LANG=COBOL,
$$FILEM NULLIND=_,
$$FILEM CSV=NO,
$$FILEM ROWS=ALL
/*
```

26.5.3 DB2 data import on line

Perform the following steps to import data from a sequential or VSM data set into DB2:

1. Select option **6** on the Utility Functions panel of File Manager/DB2, as shown in Figure 26-40.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------
FM/DB2 (DB0A)                  Utility Functions
Command ===> 6 ___________________________________________________________

1  Print         Print DB2 table or view
2  Objects       Create and drop DB2 objects
3  Copy          Copy data within DB2
4  Object List   Display and process DB2 object lists
5  Privileges    Manage DB2 privileges
6  Import        Import sequential or VSAM data into DB2
7  Export        Export DB2 data to sequential or VSAM data set
8  Create        Create DB2 test data
9  Utilities     DB2 utility job generation
10 Audit trail   Print audit trail report
11 Print browse  Browse FM/DB2 print data set
```

Figure 26-40 File Manager/DB2 Utility Functions panel - Option 6

2. Fill in the fields about the source data set and the template to be used when importing data as shown in Figure 26-41.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------
FM/DB2 (DB0A)                  Import Utility
Command ===> ____________________________________________________________

From Partitioned, Sequential or VSAM Data Set:
   Data set name . . . 'DELAHAY.FMN2.EXPORT'_____________________________
   Member  . . . . . . EMP2____
   Volume  . . . . . . _______

   Import Count  . . . ALL_____        Number of rows to import

From Copybook or Template:
   Data set name . . . 'DELAHAY.FMN.DATA'________________________________
   Member  . . . . . . EMPX______

Processing Options:
   Template usage                      Enter "/", "A"lways
   1  1. Above                         _  Edit template
      2. Previous
```

Figure 26-41 Specifying source data set and template

3. Do the same thing for the target object, as shown in Figure 26-42.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------------
Import from DELAHAY.FMN2.EXPORT(EMP2)
Command ===> ____________________________________________________________________

To DB2 Object:
   Location  . . . . . ________________        Database . . ________ (optional)
   Owner . . . . . . . DELAHAY                 Table space  ________ (optional)
   Name  . . . . . . . EMP2

To Template:  From DELAHAY.FMN.DATA(EMPX)
   Data set name . . . 'DELAHAY.FMN2.TEMPLATE'
   Member  . . . . . . EMP

Processing Options:
   Template usage                      Enter "/", "A"lways to select option
   1  1. Above                         /  Edit template mapping
      2. Previous                      _  Batch execution
      3. Generate from table           _  Use REXX proc ________
      4. Generate/Replace              _  Update existing rows
```

Figure 26-42 Specifying target DB2 object for importing data

4. If required, the user can edit the templates to specify selection criteria and edit the template mapping if there are differences between source and target formats as shown in Figure 26-43.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
FM/DB2 (DB0A)                Template Mapping                     Line 1 of 14
Command ===> ____________________________________________________ Scroll CSR

To          DELAHAY.EMP2
From        DELAHAY.FMN2.EXPORT(EMP2)

Cmd CL# To Column name  Datatype  NI   Lv From Field Name             Type  Len
        ****  Top of data  ****
___   1 EMPNO           CH(6)     NN |  2 EMPNO                        AN     6
___   2 FIRSTNME        VC(12)    NN |  2 FIRSTNME                     AN    12
___   3 MIDINIT         CH(1)     NN |  2 MIDINIT                      AN     1
___   4 LASTNAME        VC(15)    NN |  2 LASTNAME                     AN    15
___   5 WORKDEPT        CH(3)        |  2 WORKDEPT                     AN     3
___   6 PHONENO         CH(4)        |  2 PHONENO                      AN     4
___   7 HIREDATE        DATE         |  2 HIREDATE                     AN    10
___   8 JOB             CH(8)        |  2 JOB                          AN     8
___   9 EDLEVEL         SMINT        |  2 EDLEVEL                      BI     2
___  10 SEX             CH(1)        |  2 SEX                          AN     1
___  11 BIRTHDATE       DATE         |  2 BIRTHDATE                    AN    10
___  12 SALARY          DEC(9,2)     |  2 SALARY                       PD     7
___  13 BONUS           DEC(9,2)     |  2 BONUS                        PD     7
___  14 COMM            DEC(9,2)     |  2 COMM                         PD     7
```

Figure 26-43 Template mapping when importing data to DB2

5. The user can also change the correspondence between fields and edit new fields to specify default values for data creation.

   When data import is finished, File Manager/DB2 displays the confirmation message, as shown in Figure 26-44.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------
FM/DB2 (DB0A)                    Import Utility                 32 rows imported
Command ===> ____________________________________________________________

From Partitioned, Sequential or VSAM Data Set:
   Data set name . . . 'DELAHAY.FMN2.EXPORT'
   Member  . . . . . . EMP2 ___
   Volume  . . . . . . ______

   Import Count  . . . ALL ____        Number of rows to import

From Copybook or Template:
   Data set name . . . 'DELAHAY.FMN.DATA'
   Member  . . . . . . EMPX ____

Processing Options:
   Template usage                     Enter "/", "A"lways
   1  1. Above                        _  Edit template
      2. Previous
```

Figure 26-44 Data successfully imported to DB2

26.5.4 Data import to DB2 in batch mode

Example 26-7 shows a sample procedure to import data into DB2 in batch mode. This job will load data from an input PDS to table EMP2 according to an MVS template.

Example 26-7 Sample procedure to import data to DB2

```
//FMNDB2   EXEC PGM=FMNDB2,PARM=('SSID=DB0A,SQID=DELAHAY')
//STEPLIB  DD DSN=DB2CUST0.DB0A.SDSNEXIT,DISP=SHR
//         DD DSN=DSN710.SDSNLOAD,DISP=SHR
//         DD DSN=DSNDB0A.RUNLIB.LOAD,DISP=SHR
//SYSPRINT DD SYSOUT=*
//FMNTSPRT DD SYSOUT=*
//SYSTERM  DD SYSOUT=*
//SYSIN    DD *
$$FILEM DBI DSNIN=DELAHAY.FMN2.EXPORT(EMP2),
$$FILEM TCIN=DELAHAY.FMN.DATA(EMPX),
$$FILEM LANG=COBOL,
$$FILEM OBJOUT="DELAHAY"."EMP2",
$$FILEM UPDATE=NO,
$$FILEM ROWS=ALL
/*
```

26.6 Extract and load IMS data using File Manager

The File Manager component for IMS, referred to here as File Manager/IMS, can be used to perform the following actions:

- Build the extract criteria
- Run the extract in a batch mode
- View the extract reports
- Run the load in a batch mode
- View the load reports

26.6.1 IMS data extract

Use the following steps to perform IMS data extract:

1. On the File Manager/IMS Primary Option panel, shown in Figure 26-45, select option **3** to extract data from the IMS database.

```
 Process   Options   Help
 ------------------------------------------------------------------------
FM/IMS                        Primary Option Menu
Command ===> 3

0  Settings      Set processing options                 User ID . : SLITVAK
1  Browse        Browse data                            System ID : STLABF6
2  Edit          Edit data                              Appl ID . : FMI5
3  Extract       Extract data from database             Version . : 5.1.0
4  Load          Load database                          Terminal. : 3278
5  Template      Create or update template for a DBD    Screen. . : 1
6  View          Create, edit, or update view           Date. . . : 2004/11/11
7  Criteria      Create, edit, update extract criteria  Time. . . : 09:59
8  Print         Print Audit Trail
9  Utilities     FM/IMS utilities
X  Exit          Terminate FM/IMS
```

Figure 26-45 Option 3 on File Manager/IMS primary option panel

2. On the Extract Entry Panel, specify the source IMS database as shown in Figure 26-46.

```
 Process   Options   Help
 ------------------------------------------------------------------------
FM/IMS                        Extract Entry Panel
Command ===>

DBD:
   Data set name . . . 'DELAHAY.FMN.DBDLIB'
   Member  . . . . . . DJ1E

IMS:
   Subsystem Name  . . ____  (If BMP)
   AGN Name  . . . . . ________

Key values:
   Data set name . . . ____________________________________
   Member  . . . . . . ________

Processing Options:
   Region Type          Enter "/" to select option
   1  1. DLI            _  Use key values
      2. BMP            /  Use criteria
                        _  Use dynalloc DB dsnames
```

Figure 26-46 Specifying source IMS database for data extraction

3. Depending on the region type, DLI or BMP, supply and confirm the data sets associated with the source IMS database, as shown in Figure 26-47.

```
 Process   Options   Help
FM/IMS            Extract : Database Data Set Specification
Command ===> _____________________________________________________ Scroll PAGE

Database DJ1E

DBD name  ddname    Data set name
DJ1E      DJ1E      'DELAHAY.FMN.DB.DJ1E'
DJ2E      DJ2E      'DELAHAY.FMN.DB.DJ2E'
DJ2F      DJ2F      'DELAHAY.FMN.DB.DJ2F'
DJ3E      DJ3E      'DELAHAY.FMN.DB.DJ3E'
DJ3F      DJ3F      'DELAHAY.FMN.DB.DJ3F'
DJ1F      DJ1F      'DELAHAY.FMN.DB.DJ1F'
**** End of data ****

Processing Options:
   Fetch dsnames from            Enter "/" to select option
   1  1. Profile                 /  Save dsnames in profile
      2. Dynamic Allocation data set

         Press ENTER to confirm usage of the specified data set(s)
```

Figure 26-47 IMS database data sets specified

4. On the next panel, specify the extract options, that is, the template and the optional criteria as shown in Figure 26-48.

Note: There is a specific option on the IMS primary options panel (7 Criteria) to work with selection criteria.

```
 Process   Options   Help
FM/IMS                        Extract Options
Command ===> ____________________________________________________________

Database DJ1E

Template:
   Data set name . . . 'DELAHAY.FMN3.TEMPLATE'

Criteria set:
   Data set name . . . 'DELAHAY.FMN.CRITERIA'
   Member  . . . . . . EXTDJ1E

Processing Options:
   Selection criteria                Enter "/" to select option
   2  1. Use new criteria            /  Edit existing criteria
      2. Use existing criteria
      3. Use no criteria
```

Figure 26-48 File Manager/IMS extract options panel

5. The Criteria Specification panel is displayed next. The user can edit the existing criteria or add new ones, if required, as shown in Figure 26-49.

```
 Process   Options   Help
FM/IMS                 Extract : Criteria Specification
Command ===> ____________________________________________________ Scroll PAGE

Criteria DELAHAY.FMN.CRITERIA(EXTDJ1E)
Template DELAHAY.FMN.TEMPLATE(DJ1E)
          Sub
Cmd  Sel  set  Criteria subset description
 _    *     R  Relationship criteria
 _    *     1  Extraction criteria
   **** End of data ****
```

Figure 26-49 Criteria for data extract in File Manager/IMS

6. There are relationships between segments. Select the connections you want and the number of levels you want to search for extraction, as shown in Figure 26-50.

```
 Process   Options   Help
FM/IMS                    Relationship Criteria
Command ===> ____________________________________________ Scroll PAGE

Criteria DELAHAY.FMN.CRITERIA(EXTDJ1E)
Template DELAHAY.FMN.TEMPLATE(DJ1E)

Default max relationship occurrences to pursue _____  (Blank for no limit)

          Source    Source    Target                        Max occurrences to
Cmd  Sel  DBD       segment   DBD       Relationship type   pursue per DB record
 _    *   DJ1E      LINKSUB   DJ2E      Bidirectional       ____2
 _    *   DJ2E      LINKSTR   DJ3E      Bidirectional       ____5
 _    *   DJ2E      INSHIRE   DJ1E      Bidirectional       ____3
 _    *   DJ3E      INSUB     DJ2E      Bidirectional       _____
   **** End of data ****
```

Figure 26-50 Relationship criteria panel

7. The user can also specify the segments by editing the specifications as shown in Figure 26-51.

```
 Process   Options   Help
FM/IMS                    Criteria : Segment Selection
Command ===> ____________________________________________________ Scroll PAGE

Criteria DELAHAY.FMN.CRITERIA(EXTDJ1E)
Template DELAHAY.FMN.TEMPLATE(DJ1E)
Subset          ____________________________________________________

Cmd  Sel  Level           Segment   Description
 e    *   1               SHIRE
 _    *    2              SHIRENP
 _    *    2              LINKSUB
   **** End of data ****
```

Figure 26-51 Segment selection for criteria panel

8. On the next panel you have the segment description with the available layouts as shown in Figure 26-52.

```
 Process   Options   Help
FM/IMS         Criteria : Segment Layout Selection                Line 1 of 3
Command ===> ____________________________________________________ Scroll PAGE

Criteria DELAHAY.FMN.CRITERIA(EXTDJ1E)
Template DELAHAY.FMN.TEMPLATE(DJ1E)
Subset          ____________________________________________________
Segment  SHIRE

Cmd SIE Segment layout name                            Prompt   Offset Length
        ****  Top of data  ****
___  SI  SHIRE                                                 0     53
___  SI  SHIRE-TOWN                                            0     53
___  SI  SHIRE-CITY                                            0     53
        ****  End of data  ****
```

Figure 26-52 Segment layout selection for criteria

9. Edit the layouts to specify the selection criteria as shown in Figure 26-53.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
FM/IMS                    Criteria : Segment Layout                Line 1 of 11
Command ===> ____________________________________________________ Scroll PAGE

Criteria DELAHAY.FMN.CRITERIA(EXTDJ1E)                          Subset
Template DELAHAY.FMN.TEMPLATE(DJ1E)                            Segment SHIRE
----------- Criteria - Enter 1 or 2 to specify expression by field -----------
1 Id : #4='0'                                                                +
2 Sel: #3>'D'                                                                +
Offset 0
Cmd Seq SHC Ref  Field Name                            Picture  Type Start Length
                 ****  Top of data  ****
___ ___       1  1 SHIRE                                         AN      1     53
___ ___       2K  2 SHIRE-KEY                                    AN      1     20
___ ___       3K   3 SHIRE-NAME                        X(20)     AN      1     20
___ ___       4   2 SHIRE-SELECT                                 AN     21      1
___ ___       5    3 SHIRE-TYPE                        X(1)      AN     21      1
___ ___       6   2 SHIRE-ID REDEFINES SHIRE-SELECT   X(1)      AN     21      1
___ ___       7   2 SHIRE-CODE1                        9(3)      ZD     22      3
___ ___       8   2 SHIRE-CODE2                        9(5)      ZD     25      5
___ ___       9   2 SHIRE-ROAD                         S9(15)    PD     30      8
___ ___      10   2 SHIRE-RIVER                        S9(15)    PD     38      8
___ ___      11   2 SHIRE-BRIDGE                       S9(15)    PD     46      8
```

Figure 26-53 Selection criteria added

10. Exit and save the criteria. File Manager/IMS displays the confirmation message, as shown in Figure 26-54.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
FM/IMS                        Extract Options              Criteria set saved
Command ===> ____________________________________________________________________

Database DJ1E

Template:
   Data set name . . . 'DELAHAY.FMN3.TEMPLATE'

Criteria set:
   Data set name . . . 'DELAHAY.FMN.CRITERIA'
   Member  . . . . . . EXTDJ1E

Processing Options:
   Selection criteria                 Enter "/" to select option
   2  1. Use new criteria             _  Edit existing criteria
      2. Use existing criteria
      3. Use no criteria
```

Figure 26-54 Criteria set saved confirmation message

11. On the next panel, provide the specifications for the target sequential data set, as shown in Figure 26-55.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
FM/IMS                Extract : To Data Set Specification
Command ===> _____________________________________________________________________

Extract to:
   Data set name . . . FMN3.EXTRACT.DJ1E_____________________________
   Disposition . . . . NEW       (NEW or OLD)
   Volume serial . . . ______    (If not cataloged)
   Device type . . . . SYSDA___  (Generic unit or device address)

Allocation parameters (if disposition is NEW):
   Management class  . ________  (Blank for default management class)
   Storage class . . . ________  (Blank for default storage class)
   Data class  . . . . ________  (Blank for default data class)
   Space units . . . . CYL       (BLK, TRK, CYL, KB or MB)
   Primary quantity  . 1_______  (in above units)
   Secondary quantity  1_______  (in above units)
```

Figure 26-55 Target data set specifications

File Manager/IMS will generate the extraction JCL. A sample is presented in Example 26-8.

Example 26-8 IMS data extract sample procedure

```
//FILEMAN  EXEC PGM=FMN1IMSB
//STEPLIB  DD DSN=ADT5.FMN.SFMNMOD1,DISP=SHR
//SYSPRINT DD SYSOUT=*
//FMNTSPRT DD SYSOUT=*
//ISPSLIB  DD DSN=ADT5.FMN.SFMNSLIB,DISP=SHR
//SYSTERM  DD SYSOUT=*
//FMIMSIN  DD *
$$FILEM IXB REGNTYPE=DLI,
$$FILEM     DBDDSN=DELAHAY.FMN.DBDLIB,
$$FILEM     DBDMEM=DJ1E,
$$FILEM     CRITERIA=Y,
$$FILEM     DSNCRIT=DELAHAY.FMN.CRITERIA,
$$FILEM     MEMCRIT=EXTDJ1E,
$$FILEM     RESLIB1='IMS8ADT.USERRESL',
$$FILEM     RESLIB2='IMS800A.SDFSRESL',
$$FILEM     DBRC=N,
$$FILEM     IMSOBA=6,
$$FILEM     IMSNBA=10,
$$FILEM     CHKPFREQ=100,
$$FILEM     LKUNIT=CYL,
$$FILEM     LKPQTY=10,
$$FILEM     LKSQTY=10,
$$FILEM     RKUNIT=CYL,
$$FILEM     RKPQTY=10,
$$FILEM     RKSQTY=10,
$$FILEM     DFSVSAMP=IMS8ADT.PROCLIB,
$$FILEM     VSMPMEM=DFSVSM00
/*
//REPORT   DD SYSOUT=*
//UNLOAD   DD DSN=DELAHAY.FMN3.EXTRACT.DJ1E,
//         DISP=(,CATLG,DELETE),UNIT=SYSALLDA,
//         SPACE=(CYL,(1,1,0),RLSE),
//         DCB=(RECFM=VB,LRECL=00150,BLKSIZE=0)
```

```
//* ---------------------------------------------------------------
//* -- IMS DLI DATASETS REQUIRED FOR THE EXTRACT                 --
//* ---------------------------------------------------------------
//DJ1E DD DSN=DELAHAY.FMN.DB.DJ1E,DISP=SHR
//DJ2E DD DSN=DELAHAY.FMN.DB.DJ2E,DISP=SHR
//DJ2F DD DSN=DELAHAY.FMN.DB.DJ2F,DISP=SHR
//DJ3E DD DSN=DELAHAY.FMN.DB.DJ3E,DISP=SHR
//DJ3F DD DSN=DELAHAY.FMN.DB.DJ3F,DISP=SHR
//DJ1F DD DSN=DELAHAY.FMN.DB.DJ1F,DISP=SHR
```

The generated data set can be used by File Manager/IMS for upload. Figure 26-56 presents a fragment of the extracted data.

```
 Process   Options   Help

File Manager Browse DELAHAY.FMN.EXTRACT.DJ1E
Command ===> _____________________________________________ Scroll CSR
                                 Record 2       Col 1       Format CHAR
----+----1----+----2----+----3----+----4----+----5----+----6----+----7----+---
........DJ1E _  ..SHIRE   ......KALGOORLIE
........DJ1E    ..SHIRENP ......KALGOORLIE
........DJ1E    ..LINKSUB ......KALGOORLIE          HANNANS
........DJ1E    ..LINKSUB ......KALGOORLIE          KALGOORLIE
........DJ1E    ..SHIRE   ......WYALKATCHEM
........DJ1E    ..SHIRENP ......WYALKATCHEM
........DJ1E    ..LINKSUB ......WYALKATCHEM         KORRELOCKING
........DJ1E    ..LINKSUB ......WYALKATCHEM         WYALKATCHEM
........DJ1E    ..LINKSUB ......WYALKATCHEM         WYALKATCHEM SHIRE
****  End of data  ****
```

Figure 26-56 Data extracted from IMS database

Example 26-9 contains a partial IMS data export report.

Example 26-9 IMS database extract report

```
********************************* Top of Data *********************************
1IBM File Manager for z/OS IMS Component
                                          Database Extract Summary

 Primary database  DJ1E
0Extract
    From           DELAHAY.FMN.DB.DJ1E
    To             DELAHAY.FMN3.EXTRACT.DJ1E
0DBD
    Data set name  DELAHAY.FMN.DBDLIB
    Member         DJ1E
0Region Type       DLI
0Processing Options:
    Use Key values N
    Use Criteria   Y
         Data set  DELAHAY.FMN.CRITERIA(EXTDJ1E)
0Databases Extracted:
0               DBD Name  ddname    Data set name
                --------  --------  ---------------------------------------------
                DJ1E      DJ1E      DELAHAY.FMN.DB.DJ1E
                DJ2E      DJ2E      DELAHAY.FMN.DB.DJ2E
```

```
                DJ2F      DJ2F      DELAHAY.FMN.DB.DJ2F
                DJ3E      DJ3E      DELAHAY.FMN.DB.DJ3E
                DJ3F      DJ3F      DELAHAY.FMN.DB.DJ3F
                DJ1F      DJ1F      DELAHAY.FMN.DB.DJ1F
1IBM File Manager for z/OS IMS Component
                                          Database Extract Statistics Report


                                   -Segm Length- ----- Extract Statistics ------  Logically
                Database Segment Level    Min   Max      Read   Rejected  Extracted   Related DB
                ---------------------------------------------------------------------------------
                DJ1E     SHIRE     01      53    53        11          0         11
                         SHIRENP   02      11    11        11          0         11
                         LINKSUB   02      18    18        87          0         87   DJ2E
    Subtotal    DJ1E                                     109          0        109

                DJ2E     SUBURB    01      62    62        57          0         57
                         LINKSTR   02      28    28      3603          0       3603   DJ3E
                         NSTREET   02      30    30        29          0         29
                         INSHIRE   02      20    20        73          0         73   DJ1E
                         LETRBOX   02      34    34        30          0         30
    Subtotal    DJ2E                                    3792          0       3792

                DJ3E     STREET    01      32    32       215          0        215
                         INSUB     02      18    18       332          0        332   DJ2E
    Subtotal    DJ3E                                     547          0        547

    Grand total                                         4448          0       4448

                                          *** End of IMS Extract Report ***

******************************** Bottom of Data ********************************
```

26.6.2 Load data extracted from IMS by File Manager

Perform the following steps to load data into an IMS database.

1. Select option **4** on the File Manager/IMS main panel, as shown in Figure 26-57.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 FM/IMS                          Primary Option Menu
 Command ===> 4
 ------------------------------------------------------------------------------

 0  Settings      Set processing options                 User ID . : SLITVAK
 1  Browse        Browse data                            System ID : STLABF6
 2  Edit          Edit data                              Appl ID . : FMI5
 3  Extract       Extract data from database             Version . : 5.1.0
 4  Load          Load database                          Terminal. : 3278
 5  Template      Create or update template for a DBD    Screen. . : 1
 6  View          Create, edit, or update view           Date. . . : 2004/11/11
 7  Criteria      Create, edit, update extract criteria  Time. . . : 09:59
 8  Print         Print Audit Trail
 9  Utilities     FM/IMS utilities
 X  Exit          Terminate FM/IMS
```

Figure 26-57 Option 4 on File Manager/IMS primary option panel selected

2. When the next panel appears, specify the target IMS database.and processing options, as shown in Figure 26-58.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
FM/IMS                          Load Entry Panel
Command ===> ____________________________________________________________________

DBD:
   Data set name . . . 'DELAHAY.FMN.DBDLIB'____________________________
   Member  . . . . . . DJ1E____

IMS:
   Subsystem Name  . . ____  (If BMP)
   AGN Name  . . . . . ________

Processing Options:
 Region Type      Load Options                    Enter "/" to select option
 1  1. DLI        2  1. Insert only               _  Use dynalloc DB dsnames
    2. BMP           2. Insert or Update
                     3. Insert-Load PSB (If DLI)
```

Figure 26-58 File Manager/IMS load entry panel

3. If required, confirm the data sets associated with the target IMS database as shown in Figure 26-59.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
FM/IMS              Load : Database Data Set Specification
Command ===> ____________________________________________________ Scroll PAGE

Database DJ1E

DBD name  ddname    Data set name
DJ1E      DJ1E      'DELAHAY.FMN.DB.DJ1E'
DJ2E      DJ2E      'DELAHAY.FMN.DB.DJ2E'
DJ2F      DJ2F      'DELAHAY.FMN.DB.DJ2F'
DJ3E      DJ3E      'DELAHAY.FMN.DB.DJ3E'
DJ3F      DJ3F      'DELAHAY.FMN.DB.DJ3F'
DJ1F      DJ1F      'DELAHAY.FMN.DB.DJ1F'
**** End of data ****

Processing Options:
   Fetch dsnames from            Enter "/" to select option
   1  1. Profile                 /  Save dsnames in profile
      2. Dynamic Allocation data set

         Press ENTER to confirm usage of the specified data set(s)
```

Figure 26-59 IMS data set specification panel

4. Next, specify the source data set and the IMS logging options, as shown in Figure 26-60.

```
 Process   Options   Help
 ------------------------------------------------------------------------
 FM/IMS                          Load Options
 Command ===> ____________________________________________________________

 Load from:
    Data set name . . . FMN3.EXTRACT.DJ1E_____________________________
    Volume serial . . . ______    (If not cataloged)
    Device type . . . . SYSDA___  (Generic unit or device address)

 Processing Options:
  IMS log (if DLI)
  1  1. Keep
     2. Delete
     3. None
```

Figure 26-60 Source data set and IMS logging options for data load

File Manager/IMS generates the load JCL. The sample procedure is shown in Example 26-10.

Example 26-10 Sample procedure to load data into IMS database

```
//SORTSTEP EXEC PGM=SORT
//STEPLIB  DD DSN=SYS1.SORTLIB,DISP=SHR
//SYSOUT   DD SYSOUT=*
//SORTIN   DD DSN=DELAHAY.FMN3.EXTRACT.DJ1E,
//         DISP=SHR
//SORTOUT  DD DSN=&&FMN1XTSD,
//       LIKE=DELAHAY.FMN3.EXTRACT.DJ1E,
//       UNIT=SYSALLDA,
//       DISP=(NEW,PASS)
//SORTWK01 DD UNIT=SYSALLDA,
//       SPACE=(CYL,(10,10)),
//       DISP=(NEW,DELETE,DELETE)
//SORTWK02 DD UNIT=SYSALLDA,
//       SPACE=(CYL,(10,10)),
//       DISP=(NEW,DELETE,DELETE)
//SORTWK03 DD UNIT=SYSALLDA,
//       SPACE=(CYL,(10,10)),
//       DISP=(NEW,DELETE,DELETE)
//SYSIN    DD *
 RECORD TYPE=V
 SORT FIELDS=(05,2,BI,A,07,2,BI,A,09,4,BI,A)
/*
//FILEMAN  EXEC PGM=FMN1IMSB
//STEPLIB  DD DSN=ADT4.FMN.SFMNMOD1,DISP=SHR
//SYSPRINT DD SYSOUT=*
//FMNTSPRT DD SYSOUT=*
//ISPSLIB  DD DSN=ADT4.FMN.SFMNSLIB,DISP=SHR
//SYSTERM  DD SYSOUT=*
//FMIMSIN  DD *
$$FILEM ILB REGNTYPE=DLI,
$$FILEM     LOADTYPE=2,
$$FILEM     LDDSN=DELAHAY.FMN3.EXTRACT.DJ1E,
```

```
$$FILEM     DBDDSN=DELAHAY.FMN.DBDLIB,
$$FILEM     DBDMEM=DJ1E,
$$FILEM     RESLIB1='IMS8ADT.USERRESL',
$$FILEM     RESLIB2='IMS800A.SDFSRESL',
$$FILEM     DBRC=N,
$$FILEM     IMSOBA=6,
$$FILEM     IMSNBA=10,
$$FILEM     CHKPFREQ=100,
$$FILEM     IMSBKO=Y,
$$FILEM     DFSVSAMP=IMS8ADT.PROCLIB,
$$FILEM     VSMPMEM=DFSVSM00
/*
//LOAD     DD DSN=&&FMN1XTSD,DISP=SHR
//REPORT   DD SYSOUT=*
//IEFRDER  DD DSN=DELAHAY.IMSLOG.D041028.T162138.DJ1E,
//       UNIT=SYSALLDA,
//       SPACE=(TRK,(20,20),RLSE),
//       DCB=(RECFM=VB,LRECL=6140,BLKSIZE=6144),
//       DISP=(NEW,DELETE,CATLG)
//* ---------------------------------------------------------------
//* -- IMS DLI DATASETS REQUIRED FOR THE LOAD                    --
//* ---------------------------------------------------------------
//DJ1E DD DSN=DELAHAY.FMN.DB.DJ1E,DISP=OLD
//DJ2E DD DSN=DELAHAY.FMN.DB.DJ2E,DISP=OLD
//DJ2F DD DSN=DELAHAY.FMN.DB.DJ2F,DISP=OLD
//DJ3E DD DSN=DELAHAY.FMN.DB.DJ3E,DISP=OLD
//DJ3F DD DSN=DELAHAY.FMN.DB.DJ3F,DISP=OLD
//DJ1F DD DSN=DELAHAY.FMN.DB.DJ1F,DISP=OLD
```

26.7 File Export

File Manager and its DB2 and IMS components are adequate to create rather simple subsets of specific data for limited testing, for instance, unit testing.

For complex test data generation when including related data from different sources, for example sequential and VSAM files and IMS and DB2 databases, the new tool called File Export for z/OS might be more appropriate. The main panel of the File Export ISPF interface is presented in Figure 26-61.

```
ABX$MAIN  V1R1 -------  IBM File Export for z/OS  ------- 2004/12/02  04:21:24
Option  ===> ____________________________________________________________________

 1  Repository Management
 2  Session Management

 S  Settings
 A  About
 X  Exit

 IBM* ROCKET**
 Licensed Materials - Property of IBM
 5697-I12
 (C) Copyright IBM Corp. 2003,2004 All Rights Reserved.
 (C) Copyright Rocket Software, Inc. 2003-2004 All Rights Reserved.
 *Trademark of International Business Machines
 **Trademark of Rocket Software, Inc.
```

Figure 26-61 File Export ISPF interface main panel

File Export guides the user through panels to define the data extraction criteria and then generate the JCL. The batch job will extract or load the requested data. The user works with each data type in a session, as shown in Figure 26-62.

```
ABX$SMNG  V1R1 ----------  Session Management  ---------- 2004/12/02  04:30:42
Option  ===> ____________________________________________________________________

Session Name  SEQ003__    User  SERGEI
--------------------------------------------------------------------------------
New Session
 1  DB2
 2  IMS
 3  Data Set

Restore Session
 4  Open Session
 5  Session List
```

Figure 26-62 File Export session management panel

The DB2 session panel is shown in Figure 26-63.

```
ABX$DB2W  V1R1 ---------------  DB2 Menu  --------------- 2004/12/02  04:31:26
Option  ===> _____________________________________________________________

Session Name  SEQ001

Database specification
 1  Source and Target

Data specification
 2  Relationship Specification
 3  Query Specification and Manipulation

JCL specification
 4  Generate JCL
```

Figure 26-63 File Export DB2 session panel

The IMS session panel is presented in Figure 26-64.

```
ABX$IMST  V1R1 ---------- IMS Source and Target Type Specification  ---------
Option  ===> ____________________________________________________________

Session Name  SEQ002

Select the Source IMS type and the Target IMS type by typing a slash '/'
next to the IMS type.

Select the Source IMS Type:           Select the Target IMS Type:

  IMS   _                               IMS       _
  DEDB  _                               DEDB      _
  DLI   _                               DLI       _
  DBB   _                               DBB       _
  BMP   _                               BMP       _
                                        FEUNLOAD  _
```

Figure 26-64 File Export IMS session panel

When the user creates a new IMS session, File Export asks for the source type and the target type as shown in Figure 26-65.

```
ABX$IMST  V1R1 ---------- IMS Source and Target Type Specification  ---------

Session Name  IMS

Select the Source IMS type and the Target IMS type by typing a slash '/'
next to the IMS type.

Select the Source IMS Type:            Select the Target IMS Type:

  IMS   _                                IMS       _
  DEDB  _                                DEDB      _
  DLI   _                                DLI       _
  DBB   _                                DBB       _
  BMP   _                                BMP       _
                                         FEUNLOAD  _



Option  ===> _
```

Figure 26-65 File Export new IMS session panel

The MVS data (sequential and VSAM data sets) session panel is shown in Figure 26-66.

```
ABX$SEQW  V1R1 --------- Sequential & VSAM Menu  -------- 2004/12/02  04:32:51
Option  ===> ____________________________________________________________________

Session Name  SEQ003

Settings Specification
 0  Session Settings

Data set Specification
 1  Source & Target Data Set Specification

Data specification
 2  Variable and Literal Specification
 3  Relationship Specification
 4  Field Specification
 5  Type and Select statements Specification and Manipulation

JCL specification
 6  Generate JCL
```

Figure 26-66 File Export MVS data session panel

The language used by File Export is similar to SQL. There are a source data description paragraph and a data selection paragraph. The following sections present some samples of the procedures using File Export to extract data.

26.7.1 Copybooks: File Manager versus File Export

Table 26-4 identifies the correspondence between COBOL copybook used by File Manager and the field definitions used by File Export.

Table 26-4 Field descriptions

File Manager for z/OS	File Export for z/OS
01 REC-TYPE01. 05 REC-TYPE PIC XX. 05 NAME PIC X(20). 03 EMPLOYEE-NO PIC 9(4) BINARY. 03 AGE PIC 9(4) BINARY. 03 SALARY PIC9(7) PACKED-DECIMAL. 03 MONTH PIC 9(8) BINARY OCCURS 12 TIMES. 03 FILLER PIC XX. 01 REC-TYPE02. 05 REC-TYPE PIC XX. 05 NAME PIC X(20). 03 JOB-TITLE PIC X(14). 03 ADDR1 PIC X(20). 03 ADDR2 PIC X(20). 03 POSTCODE PIC X(4).	:: TYPE-01 FIELD(REC-TYPE) = (1,2,CHAR); FIELD(NAME) = (3,20,CHAR); FIELD(EMPLOYEE-NO) = (23,4,HEX); FIELD(AGE) = (27,4,HEX); FIELD(SALARY) = (31,7,PACKED); FIELD(MONTH) = (NEXT,8,HEX) OCCURS (12) TGLEN (48); FIELD(FILLER) = (79,2,CHAR); :: TYPE-02 FIELD(REC-TYPE) = (1,2,CHAR); FIELD(NAME) = (3,20,CHAR); FIELD(JOB-TITLE) = (23,14,CHAR); FIELD(ADDR1) = (37,20,CHAR); FIELD(ADDR2) = (57,20,CHAR); FIELD(POSTCODE) = (77,4,CHAR);
	:: SELECT-01 TYPE (TYPE01) WHEN REC-TYPE='01'; :: SELECT-02 TYPE (TYPE02) WHEN REC-TYPE='02';

Table 26-5 shows the correspondence between COBOL copybooks and SQL definitions.

Table 26-5 Field descriptions for DB2 data

DDL for DB2 table	COBOL copybook
FMNDB2.EMP EMPNO CHAR(6) FIRSTNME VARCHAR(12) MIDINIT CHAR(1) LASTNAME VARCHAR(15) WORKDEPT CHAR(3) PHONENO CHAR(4) HIREDATE DATE JOB CHAR(8) EDLEVEL SMALLINT SEX CHAR(1) BIRTHDATE DATE SALARY DECIMAL(9, 2) BONUS DECIMAL(9, 2) COMM DECIMAL(9, 2)	01 TABLE-EMP. 03 EMPNO PIC X(6). 03 FIRSTNME PIC X(12). 03 MIDINIT PIC X(1). 03 LASTNAME PIC X(15). 03 WORKDEPT PIC X(3). 03 PHONENO PIC X(4). 03 HIREDATE PIC X(10). 03 JOB PIC X(8). 03 EDLEVEL PIC 9(4) BINARY. 03 SEX PIC X. 03 BIRTHDATE PIC X(10). 03 SALARY PIC S9(11)V9(2) USAGE COMP-3. 03 BONUS PIC S9(11)V9(2) USAGE COMP-3. 03 COMM PIC S9(11)V9(2) USAGE COMP-3. 03 FILLER PIC X(51).
FMNDB2.DEPT DEPTNO CHAR(3) DEPTNAME VARCHAR(36) MGRNO CHAR(6) ADMRDEPT CHAR(3) LOCATION CHAR(16)	01 TABLE-DEPT. 03 DEPTNO PIC X(3). 03 DEPTNAME PIC X(36). 03 MGRNO PIC X(6). 03 ADMRDEPT PIC X(3). 03 LOCATION PIC X(16).

26.7.2 A few samples

Some of the capabilities of File Export are presented in the following examples.

One MVS to one MVS

The first sample shows a one-to-one extraction of MVS data, from one MVS (partitioned) data set to another MVS (sequential) data set:

1. Create a new MVS session.
2. Select option **1** on the MVS session panel to specify the input and the output data sets as shown in Figure 26-67.

```
ABX$VSDB  V1R1 ---------  Source and Target Data Set Specification  ---------

Session Name  SEQ00

Select Source Type and Enter Data Set Information:
  Source Type    S  (VSAM / Sequential)
  Source File    DELAHAY.FMN.DATA(FMNCDATA)
  Source Member  ________  (blank for sequential,
                            "*" for member selection list)

Select Target Type and Enter Data Set Information:
  Target Type    S  (VSAM / Sequential)
  Target File    DELAHAY.ABX.EXPORT.SEQ
  Target Member  ________  (blank for sequential,
                            "*" for member selection list)

Valid Option Commands:  S - Specify Source DD  T - Specify Target DD




Option  ===>
```

Figure 26-67 Data sets specification for MVS session

3. Select option **4** on the MVS session panel to specify the fields in the input data set as shown in Figure 26-68.

```
ABX$FSEQ  V1R1 ----------------  Fields  ---------------- 2004/12/03  18:01:36

Session Name  SEQ00
File          DELAHAY.FMN.DATA(FMNCDATA)
Max Size      80
                                                                Row 1 of 12
Cmd  Field                          Start   Size    Type      Scramble
_    REC-TYPE                       1       2       CHAR      N
_    NAME                           3       20      CHAR      N
_    EMPLOYEE-NO                    23      4       HEX       N
_    AGE                            27      4       HEX       N
_    SALARY                         31      7       PACKED    N
_    MONTH                          38      4       HEX       N
_    FILLER                         79      2       CHAR      N
_    JOB-TITLE                      23      14      CHAR      N
_    ADDR1                          37      20      CHAR      N
_    ADDR2                          57      20      CHAR      N

Valid Line Commands  :  E - Edit Field  D - Delete Field
Valid Option Commands:  A - Add Field



Option  ===>                                                Scroll ===> PAGE
```

Figure 26-68 Fields definitions for an MVS session

When you add or edit a field, the appropriate values for field specification are filled as shown in Figure 26-69.

```
ABX$NFLD  V1R1 ----------  Field Specification  --------- 2004/12/03  18:02:41
                                                                More:     +
Enter Field information below:
  Field Name       MONTH                          
  Field Position   38      
  Field Length     4   
  Field Type       HEX     

  Occurs           12
  Depending On     
  Precision        __
  Scale            __
  Signed           N  (Y/N)
  Scramble         N  (Y/N)
  Group Length     48  
  Maximum Length   
  Position Algorithm                                                  
  Group Length Algorithm                                              

Enter "/" to select option
_  Specify additional Position Algorithm
_  Specify additional Group Length Algorithm

Option  ===>                                                          
```

Figure 26-69 Field specification for MVS data

4. Select option **5** on the MVS session panel to specify the extraction criteria. Start by defining the data types available in the source as shown in Figure 26-70.

```
ABX$SQTL  V1R1 -----------  Record Type List  ----------- 2004/12/03  18:05:05

Session Name  SEQ00

                                                            Row 1 of 3
Cmd  Name                          Excl  Skip        Limit       Then
_    TYPE-01                       N     __________  __________  __________
_    TYPE-02                       Y     __________  __________  __________
_    DEFAULT                       Y     __________  __________  __________




Valid Line Commands :  E - Edit  D - Delete  F - Field List
                       S - SELECT Statement List  Q - SQL Output
Valid Option Command:  A - Add New Record Type  L - Load from Repository



Option  ===>                                          Scroll ===> PAGE
```

Figure 26-70 Type specification for MVS data

5. For each type, specify whether it will be selected or not by typing `Y` or `N` in the Exclusion field.

   When you create or edit a type, you must specify the identification criteria for this type as shown in Figure 26-71.

```
ABX$SQLT  V1R1 --------------  Record Type  ------------- 2004/12/03  18:06:00

Type Name  TYPE-01

                                                                    Row 1 of 1
Cmd  Value
_    REC-TYPE='01'

Valid Line Commands :  D - Delete Record Type Selection Criteria
Valid Option Command:  A - Add Record Type Selection Criteria

Option  ===>                                                  Scroll ===> PAGE
```

Figure 26-71 Type identification for MVS data

If you do not specify fields, all the fields will be selected.

6. You can add a selection criteria to limit the extraction to specific records as shown in Figure 26-72.

```
ABX$CLST  V1R1 ----------  Selection Criteria  ---------- 2004/12/03  18:49:31

Session Name      MVS01
Record Type Name  TYPE-01
                                                                    Row 1 of 1
Cmd  Value
_    NAME>'F'

Valid Line Commands  :  D - Delete Line
Valid Option Commands:  A - Add Line

Option  ===> ____________________________________________ Scroll ===> PAGE
```

Figure 26-72 Record selection for MVS data

7. Select option **6** on the MVS session panel to start the extraction JCL generation as shown in Figure 26-73.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       DELAHAY.ABX.JCLLIB(MVS01) - 01.00              Columns 00001 00072
****** **************************** Top of Data ****************************
000001 //FILEXPOR JOB DELAHAY,CLASS=A,NOTIFY=&SYSUID,MSGCLASS=X
000002 //*
000003 //STEP1    EXEC PGM=ABXUTIL
000004 //STEPLIB  DD   DISP=SHR,DSN=ADT4.ABX.SABXLOAD
000005 //*
000006 //ABXIN    DD   DISP=SHR,DSN=DELAHAY.ABX.DATA(FMNCDATA)
000007 //ABXOUT   DD   DISP=SHR,DSN=DELAHAY.ABX.EXPORT.SEQ
000008 //ABXPRINT DD   SYSOUT=*
000009 //ABXCTL   DD   *
000010 OPTIONS((SEQ, SEQ), DECIMAL, PERCENT, NOQUIET, FIELDS);
000011 WUNIT(SYSDA);
000012 FIELD(REC-TYPE) = (1,2,CHAR);
000013 FIELD(NAME) = (3,20,CHAR);
000014 FIELD(EMPLOYEE-NO) = (23,4,HEX);
000015 FIELD(AGE) = (27,4,HEX);
000016 FIELD(SALARY) = (31,7,PACKED);
000017 FIELD(MONTH) = (NEXT,8,HEX)
000018  OCCURS(12)
000019  TGLEN(48)
Command ===> ____________________________________________ Scroll ===> CSR
```

Figure 26-73 Generated JCL for MVS data

The resulting JCL is shown in Example 26-11.

Example 26-11 Data extraction from one data set to another data set

```
//*-----------------------------------------------------------------
//* Two record types in the input
//*     identified by REC-TYPE value 01 or 02
//* Select records of type 01 whose name is further than 'F' in
//*     alphabetical order
//*-----------------------------------------------------------------
//STEP1    EXEC PGM=ABXUTIL
//STEPLIB  DD   DISP=SHR,DSN=ADT4.ABX.SABXLOAD
//*
//ABXIN    DD   DISP=SHR,DSN=DELAHAY.ABX.DATA(FMNCDATA)
//ABXOUT   DD   DISP=SHR,DSN=DELAHAY.ABX.EXPORT.SEQ
//ABXPRINT DD   SYSOUT=*
//ABXCTL   DD   *
OPTIONS((SEQ, SEQ), DECIMAL, PERCENT, NOQUIET, FIELDS);
WUNIT(SYSDA);
FIELD(REC-TYPE) = (1,2,CHAR);
FIELD(NAME) = (3,20,CHAR);
FIELD(EMPLOYEE-NO) = (23,4,HEX);
FIELD(AGE) = (27,4,HEX);
FIELD(SALARY) = (31,7,PACKED);
FIELD(MONTH) = (NEXT,8,HEX)
OCCURS(12)
 TGLEN(48)
 MAXLEN(-1);
FIELD(FILLER) = (79,2,CHAR);
FIELD(JOB-TITLE) = (23,14,CHAR);
FIELD(ADDR1) = (37,20,CHAR);
FIELD(ADDR2) = (57,20,CHAR);
FIELD(POSTCODE) = (77,4,CHAR);
  TYPE (TYPE-01) WHEN
  REC-TYPE='01';
  SELECT *
      FROM RECORD WHERE
```

```
    NAME>'F';
  TYPE (TYPE-02) WHEN
  REC-TYPE='02';
  EXCLUDE;
  TYPE (DEFAULT);
  EXCLUDE;
/*
```

You can check the result in the output data set shown in Figure 26-74.

```
 Process   Options   Help
File Manager Browse DELAHAY.ABX.EXPORT.SEQ
                                                    Record 0        Format TABL
REC-TYPE NAME                 EMPLOYEE-NO     AGE   SALARY     MONTH(1)
£2       £4                            £5      £6       £7           £8
AN 1:2   AN 3:20                  BI 23:2 BI 25:2  PD 27:4      BI 31:4
<>       <---+----1----+---->      <---+>  <---+> <---+--> <---+----1>
****  Top of data  ****
01       Grant Smithy                1234      35    75000            6
01       Graham Prestcott            5583      28    48000            7
01       Ted Dexter                  3327      52    60250           14
01       Roddy Armstrong             5683      34    77000           28
01       James Browne                1117      46   125000           47
01       Silvia Carrot               2308      29    61400            1
01       John Laws                   3422      21    46750           14
01       Liz Childs                  3439      55    66000           44
01       Keith Sampson               2329      40    71550           26
01       John Neptune                3486      40    69000           30
01       Michael Bevan               5455      46    83250           10
01       Mary Sands                  7790      36    58000           18
****  End of data  ****

Command ===>                                                     Scroll CSR
```

Figure 26-74 Output data set

One MVS to two MVS

The second sample shows a one-to-many extraction of MVS data, from one MVS data set to several MVS data sets:

1. Create a new MVS session.
2. Select option **1** on the MVS session panel to specify the input and the output data sets. Select option **T** to define the Target data sets as shown in Figure 26-75.

```
ABX$DDE2  V1R1 --------  DD Statement Definition  ------- 2004/12/03  18:32:39

DSN  DELAHAY.ABX.DATA(EXPORT1)

 ENTER TARGET DD PARAMETERS:

Cmd  Text
     DSN=DELAHAY.ABX.DATA(EXPORT1),
_    DISP=SHR
_    DSN=DELAHAY.ABX.EXPORT.SEQ,DISP=SHR





Valid Line Commands  :  D - Delete line
Valid Option Commands:  A - Add line  T - Generate DD statement from template
                        W - Generate DD statement with wizard



Option  ===>                                                  Scroll ===> PAGE
```

Figure 26-75 Defining several output data sets

3. This will define several DD statements for the same ABXOUT DD card; you will have to modify the draft JCL as shown in Figure 26-76 to create several DD cards.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       DELAHAY.ABX.JCLLIB(MVS02) - 01.01              Columns 00001 00072
****** ***************************** Top of Data ******************************
000001 //FILEXPOR JOB DELAHAY,CLASS=A,NOTIFY=&SYSUID,MSGCLASS=X
000002 //*
000003 //STEP1    EXEC PGM=ABXUTIL
000004 //STEPLIB  DD   DISP=SHR,DSN=ADT4.ABX.SABXLOAD
000005 //*
000006 //ABXIN    DD   DISP=SHR,DSN=DELAHAY.ABX.DATA(FMNCDATA)
000007 //ABXOUT   DD   DSN=DELAHAY.ABX.DATA(EXPORT1),
000008 //              DISP=SHR
000009 //         DD   DSN=DELAHAY.ABX.EXPORT.SEQ,DISP=SHR
000010 //ABXPRINT DD   SYSOUT=*
000011 //ABXCTL   DD   *
000012 OPTIONS((SEQ, SEQ), DECIMAL, PERCENT, NOQUIET, FIELDS);
000013 WUNIT(SYSDA);
000014 FIELD(REC-TYPE) = (1,2,CHAR);
000015 FIELD(NAME) = (3,20,CHAR);
000016 FIELD(EMPLOYEE-NO) = (23,4,HEX);
000017 FIELD(AGE) = (27,4,HEX);
000018 FIELD(SALARY) = (31,7,PACKED);
000019 FIELD(MONTH) = (NEXT,8,HEX)
Command ===>                                              Scroll ===> CSR
```

Figure 26-76 Draft JCL generated by File Export

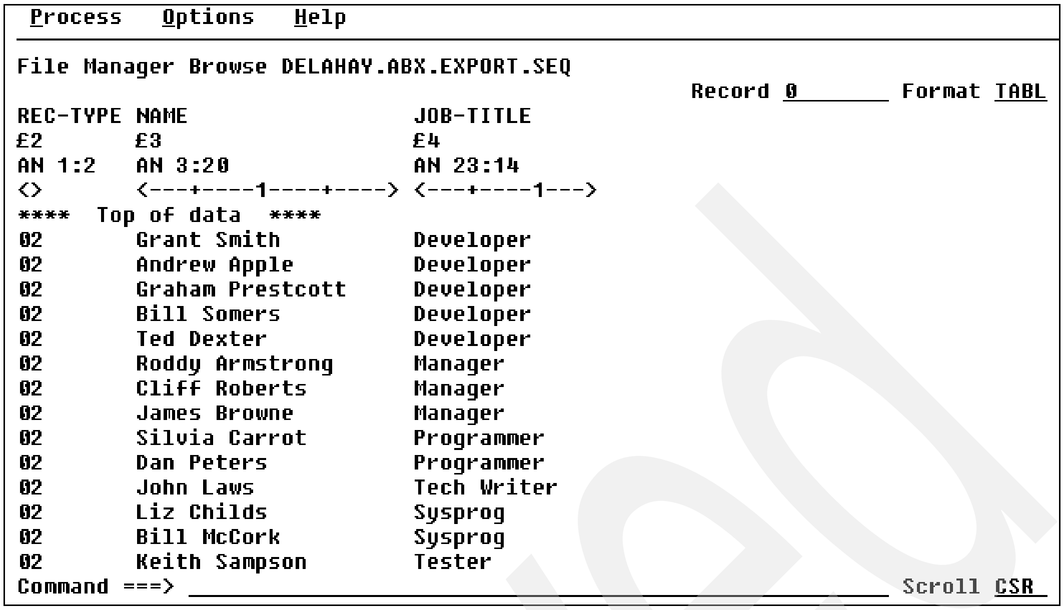
4. You will also have adapt the SELECT statements as shown in the Example 26-12.

Example 26-12 Extracting data to two different data sets

```
//*-----------------------------------------------------------------
//* Two record types in the input
//*     identified by REC-TYPE value 01 or 02
//* Select some fields of records of type 01
//* Select all records of type 02 and write them in another data set
//*-----------------------------------------------------------------
```

```
//STEP1    EXEC PGM=ABXUTIL
//STEPLIB  DD   DISP=SHR,DSN=ADT4.ABX.SABXLOAD
//*
//ABXIN    DD   DISP=SHR,DSN=DELAHAY.ABX.DATA(FMNCDATA)
//ABXOUT   DD   DISP=SHR,DSN=DELAHAY.ABX.DATA(EXPORT1)
//ABXOUT2  DD   DISP=SHR,DSN=DELAHAY.ABX.EXPORT.SEQ
//ABXPRINT DD   SYSOUT=*
//ABXCTL   DD   *
OPTIONS((SEQ, SEQ), DECIMAL, PERCENT, NOQUIET, FIELDS);
WUNIT(SYSDA);
FIELD(REC-TYPE) = (1,2,CHAR);
FIELD(NAME) = (3,20,CHAR);
FIELD(EMPLOYEE-NO) = (23,4,HEX);
FIELD(AGE) = (27,4,HEX);
FIELD(SALARY) = (31,7,PACKED);
FIELD(MONTH) = (NEXT,8,HEX)
 OCCURS(12)
 TGLEN(48)
 MAXLEN(-1);
FIELD(FILLER) = (79,2,CHAR);
FIELD(JOB-TITLE) = (23,14,CHAR);
FIELD(ADDR1) = (37,20,CHAR);
FIELD(ADDR2) = (57,20,CHAR);
FIELD(POSTCODE) = (77,4,CHAR);
  TYPE (TYPE-01) WHEN
  REC-TYPE='01';
  SELECT REC-TYPE,
         NAME,
         EMPLOYEE-NO,
         SALARY FROM RECORD;
  TYPE (TYPE-02) WHEN
  REC-TYPE='02';
  SELECT NOTHING FROM RECORD ;
  THEN INSERT INTO DD:ABXOUT2
  VALUES (REC-TYPE,NAME,JOB-TITLE);
:: nothing from the input will go to the default output
:: some columns will be copied in another output
TYPE (DEFAULT);
  EXCLUDE;
/*
```

5. The second output data set contains the records of type 02 extracted from the input as shown in Figure 26-77.



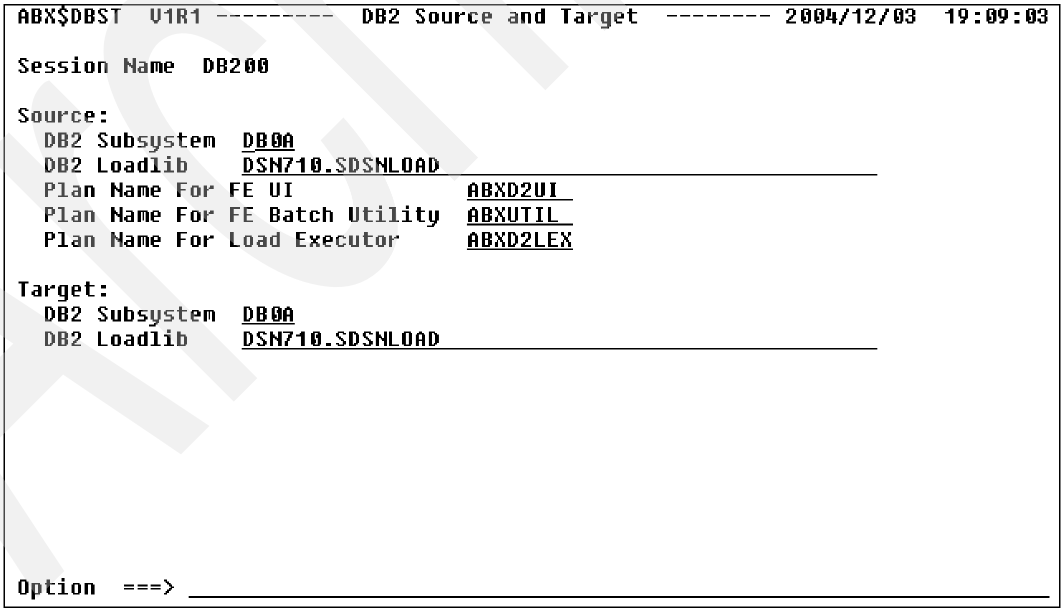
```
 Process   Options   Help
──────────────────────────────────────────────────────────────────────────────
File Manager Browse DELAHAY.ABX.EXPORT.SEQ
                                                    Record 0         Format TABL
REC-TYPE NAME                 JOB-TITLE
£2       £3                   £4
AN 1:2   AN 3:20              AN 23:14
<>       <---+----1----+----> <---+----1--->
****  Top of data  ****
02       Grant Smith          Developer
02       Andrew Apple         Developer
02       Graham Prestcott     Developer
02       Bill Somers          Developer
02       Ted Dexter           Developer
02       Roddy Armstrong      Manager
02       Cliff Roberts        Manager
02       James Browne         Manager
02       Silvia Carrot        Programmer
02       Dan Peters           Programmer
02       John Laws            Tech Writer
02       Liz Childs           Sysprog
02       Bill McCork          Sysprog
02       Keith Sampson        Tester
Command ===>                                                     Scroll CSR
```

Figure 26-77 Extracted MVS data

One DB2 to one MVS

The third sample shows a one-to-one extraction of DB2 data, from one DB2 table to one MVS data set:

1. Create a new DB2 session.
2. Select option **1** on the DB2 session panel to specify the input DB2 sub-system ID as shown in Figure 26-78.



```
ABX$DBST  V1R1 ---------  DB2 Source and Target  -------- 2004/12/03  19:09:03

Session Name  DB200

Source:
  DB2 Subsystem  DB0A
  DB2 Loadlib    DSN710.SDSNLOAD
  Plan Name For FE UI             ABXD2UI
  Plan Name For FE Batch Utility  ABXUTIL
  Plan Name For Load Executor     ABXD2LEX

Target:
  DB2 Subsystem  DB0A
  DB2 Loadlib    DSN710.SDSNLOAD







Option  ===>
```

Figure 26-78 DB2 session parameters

3. Select option **1** on the DB2 session panel to specify the query parameters. The Edit Query panel is displayed as shown in Figure 26-79.

```
ABX$DBQE  V1R1 --------------  Edit Query  -------------- 2004/12/03  19:10:33

Session Name  DB200
Query Name    QUERY01

Data specification
  1  Target Specification
  2  Edit Source Tables
  3  Edit Source Columns
  4  Edit Selection Criteria
  5  Target Columns Modification Functions
  6  Edit Intermediate Dataset Specification

Option  ===> _
```

Figure 26-79 DB2 query edit panel

4. Select option **1**, Target specifications on the Edit query panel to specify the output as shown in Figure 26-80.

```
ABX$DBQT  V1R1 ---------  Target Specification  --------- 2004/12/03  19:11:39

Session Name  DB200
Query Name    QUERY01

Target Table Owner   XXX
Target Table Name    YYY

Limit Target Rows    7        (Specify Limit Value)
Select After Limit   _        (Select One Row Every 'n' Rows)
Skip First 'n' Rows  _

Enter '/' to Select From Options Below
  /  Specify RESUME on LOAD
  _  Specify REPLACE on LOAD
  _  Specify REUSE on LOAD

  _  Specify NOTHING. All records will be suppressed.

Option  ===> _
```

Figure 26-80 Dummy output table

You can limit the number of retrieved rows by typing a value in the Limit target rows field. Press PF3 to return to the Edit query panel.

5. Select option 2, Edit source table to specify the input table as shown in Figure 26-81.

```
ABX$DBTL  V1R1 -------------  Source Tables  ------------ 2004/12/03  19:16:01
  ┌──────────────── Table Specification ────────────────┐
S │                                                     │
Q │    Table Owner  DELAHAY                             │
  │                                                     │                Row 1 of 1
C │    Table Name   EMP                                 │
e │                                                     │
  │                                                     │
  │                                                     │
  │                                                     │
  └─────────────────────────────────────────────────────┘
Valid Line Commands  :  E - Edit  D - Delete
Valid Option Commands:  A - Add

Option  ===>                                                Scroll ===> PAGE
```

Figure 26-81 Input table specification

Press PF3 to return to the Edit query panel.

6. You can then specify the columns you want to extract as shown in Figure 26-82. Select option **3,** Edit source columns in the Edit query panel.

```
ABX$DBCL  V1R1 -------------  Query Columns  ------------ 2004/12/03  19:17:48

Session Name  DB200
Query Name    QUERY01
                                                                  Row 1 of 5
Cmd  Table                          Column              Type
_    DELAHAY.EMP                    EMPNO               CHAR
_    DELAHAY.EMP                    FIRSTNME            VARCHAR
_    DELAHAY.EMP                    LASTNAME            VARCHAR
_    DELAHAY.EMP                    WORKDEPT            CHAR
_    DELAHAY.EMP                    SALARY              DECIMAL

Valid Line Commands  :  D - Delete
Valid Option Commands:  A - Add Column  L - Add Columns from list

Option  ===> ____________________________________________________ Scroll ===> PAGE
```

Figure 26-82 Input table selected columns

In this example, do not select any column so you can get them all. Press PF3 to return to the Edit query panel.

7. Select option **6**, Edit Intermediate Dataset Specification in the Edit query panel to specify the output data set as shown in Figure 26-83.

```
ABX$DDE1  V1R1 --------  DD Statement Definition  ------- 2004/12/03  19:20:03

DSN  DELAHAY.ABX.EXPORT.SEQ

 ENTER DD PARAMETERS FOR DATASET:

Cmd  Text
     DSN=DELAHAY.ABX.EXPORT.SEQ,
_    DISP=SHR

Valid Line Commands  :  D - Delete line
Valid Option Commands:  A - Add line  T - Generate DD statement from template
                        W - Generate DD statement with wizard

Option  ===> _                                          Scroll ===> PAGE
```

Figure 26-83 Output data set

Press PF3 to return to the session main panel.

8. Select option **4**, Generate JCL as shown in Figure 26-84.

```
ABX$DBJC  V1R1 ------------  JCL Generation  ------------ 2004/12/03  19:21:19

Session Name  DB200

Enter data set and member name where JCL will be saved:
  Data set name  DELAHAY.ABX.JCLLIB
  Member name    DB200X   (Blank for member list)

Enter "/" to select option
  /  Generate export only

Enter AL command to add more STEPLIBs

Option  ===> _
```

Figure 26-84 Generate JCL panel

Do not forget to limit the JCL to the export step.

9. File Export displays the generated JCL as shown in Example 26-13.

Example 26-13 Generated JCL

```
//STEP1    EXEC PGM=ABXUTIL
//STEPLIB  DD   DISP=SHR,DSN=ADT4.ABX.SABXLOAD
//         DD   DISP=SHR,DSN=DSN710.SDSNLOAD
//*
```

```
//*
//ABXCTL   DD   *
OPTIONS(DB2, PLAN(ABXUTIL));
SOURCEDB2(DB0A);
TARGETDB2(DB0A);
TABLE(QUERY01)
SOURCE(DELAHAY.EMP)
TARGET(XXX.YYY) DD:TABL0001 LOAD(RESUME);
SELECT *
FROM DELAHAY.EMP A;
LIMIT (7);
//ABXCNTLI DD   DISP=OLD,DSN=DELAHAY.ABXCNTL.D041202.T182123.CNTLIN
//ABXCNTLO DD   DSN=DELAHAY.ABXCNTL.D041202.T182123.CNTLOUT,
//             DISP=(NEW,CATLG),
//             UNIT=SYSDA,
//             SPACE=(TRK,(1,1),RLSE)
//ABXPRINT DD   SYSOUT=*
//SYSABEND DD   SYSOUT=*
//TABL0001 DD   DSN=DELAHAY.ABX.EXPORT.SEQ,
//             DISP=SHR
/*
```

Note: For export to MVS only, you can replace the ABXCNTLI data set generated here by File Export by * and the ABXCNTLO data set by SYSOUT=*.

10. Instead of writing to the external “table,” you can specify an output data set as shown in Example 26-14.

Example 26-14 Additional data set output

```
...
TABLE(QUERY01)
SOURCE(DELAHAY.EMP)
TARGET(XXX.YYY) DD:TABL0001 LOAD(RESUME);
SELECT *
FROM DELAHAY.EMP A;
LIMIT (7);
THEN INSERT INTO DD:MYOUT
     VALUES (EMPNO, ' ',
             FIRSTNME, ' ',
             LASTNAME, ' ',
             WORKDEPT);
...
```

11. You can check the extracted records as shown in Figure 26-85.

```
 Process   Options   Help
File Manager Browse DELAHAY.ABX.EXPORT.SEQ
                                   Record 0        Col 1          Format CHAR
----+----1----+----2----+----3----+----4----+----5----+----6----+----7----+---
****  Top of data  ****
          000010..CHRISTINE   I...HAAS           A0039781965-01-01PRES    ..F19
          000110..VINCENZO    G...LUCCHESI       A0034901958-05-16SALESREP..M19
          000120..SEAN         ...O'CONNELL      A0021671963-12-05CLERK   ..M19
          000020..MICHAEL     L...THOMPSON       B0134761973-10-10MANAGER ..M19
          000030..SALLY       A...KWAN           C0147381975-04-05MANAGER ..F19
          000130..DOLORES     M...QUINTANA       C0145781971-07-28ANALYST ..F19
          000140..HEATHER     A...NICHOLLS       C0117931976-12-15ANALYST ..F19
****  End of data  ****

Command ===> _____________________________________________________ Scroll CSR
```

Figure 26-85 Extracted rows in a sequential file

12. Compare it to the original table, shown in Figure 26-86.

```
 Process   Options   Utilities   Help
FM/DB2 (DB0A)                    Table Browse                  33 rows fetched
TABLE FMNDB2.EMP                                                   Format TABL
EMPNO  FIRSTNME     MIDINIT LASTNAME        WORKDEPT PHONENO HIREDATE   JOB
£1     £2           £3      £4              £5       £6      £7         £8   +
CH(6)  VARCHAR(12)  CH(1)   VARCHAR(15)     CH(3)    CH(4)   DATE       CH(8)
PU--+> <---+----1-> -       <---+----1----> <-NF     <-->    <---+----> <---+-
****  Top of data  ****
000010 CHRISTINE<   I       HAAS<           A00      3978    1965-01-01 PRES
000110 VINCENZO<    G       LUCCHESI<       A00      3490    1958-05-16 SALESR
000120 SEAN<                O'CONNELL<      A00      2167    1963-12-05 CLERK
000020 MICHAEL<     L       THOMPSON<       B01      3476    1973-10-10 MANAGE
000030 SALLY<       A       KWAN<           C01      4738    1975-04-05 MANAGE
000130 DOLORES<     M       QUINTANA<       C01      4578    1971-07-28 ANALYS
000140 HEATHER<     A       NICHOLLS<       C01      1793    1976-12-15 ANALYS
000060 IRVING<      F       STERN<          D11      6423    1973-09-14 MANAGE
000150 BRUCE<               ADAMSON<        D11      4510    1972-02-12 DESIGN
000160 ELIZABETH<   R       PIANKA<         D11      3782    1977-10-11 DESIGN
000170 MASATOSHI<   J       YOSHIMURA<      D11      2890    1978-09-15 DESIGN
000180 MARILYN<     S       SCOUTTEN<       D11      1682    1973-07-07 DESIGN
000190 JAMES<       H       WALKER<         D11      2986    1974-07-26 DESIGN
000200 DAVID<               BROWN<          D11      4501    1966-03-03 DESIGN
Command ===> _____________________________________________________ Scroll CSR
```

Figure 26-86 Fragment of original DB2 table

13. The exported data is displayed by File Manager using the copybook shown in Example 26-15 as the “table” export.

Example 26-15 COBOL copybook for exported EMP table

```
01 TABLE-EMP.
   03 EMPNO        PIC X(6).
   03 LEN-FIRST    PIC X(2).
   03 FIRSTNME     PIC X(12).
   03 MIDINIT      PIC X(1).
   03 LEN-LAST     PIC X(3).
```

```
03 LASTNAME     PIC X(15).
03 WORKDEPT     PIC X(3).
03 PHONENO      PIC X(4).
03 HIREDATE     PIC X(10).
03 JOB          PIC X(8).
03 EDLEVEL      PIC 9(4) BINARY.
03 SEX          PIC X.
03 BIRTHDATE    PIC X(10).
03 SALARY       PIC S9(7)V9(2) USAGE COMP-3.
03 BONUS        PIC S9(7)V9(2) USAGE COMP-3.
03 COMM         PIC S9(7)V9(2) USAGE COMP-3.
```

Note that File Export added a length field before the variable length fields FIRSTNME and LASTNAME and that the numeric fields are shorter than the expected definition. This is because it is an intermediary data set for the DB2 loader.

You would not need this for the data set output as shown in Example 26-16.

Example 26-16 File Export output in MYOUT

```
********************************** TOP OF DATA **********************************
000010 CHRISTINE    HAAS           A00
000110 VINCENZO     LUCCHESI       A00
000120 SEAN         O'CONNELL      A00
000020 MICHAEL      THOMPSON       B01
000030 SALLY        KWAN           C01
000130 DOLORES      QUINTANA       C01
000140 HEATHER      NICHOLLS       C01
******************************** BOTTOM OF DATA ********************************
```

26.7.3 Relationship specification: Filter

The main advantage of File Export is the possibility to define relationships between data to make a comprehensive extraction from several sources:

1. Start by creating an external list of items, such as a list of departments' names beginning by a “D,” extracted from the FMNDB2.DEPT table. Example 26-17 is a working File Export JCL step to do this extraction.

   Example 26-17 Extract D departments with File Export*

```
//DEPTEXTR EXEC PGM=ABXUTIL
//STEPLIB  DD   DISP=SHR,DSN=ADT4.ABX.SABXLOAD
//         DD   DISP=SHR,DSN=DSN710.SDSNLOAD
//*
//ABXCTL   DD   *
OPTIONS(DB2, PLAN(ABXUTIL));
SOURCEDB2(DB0A);
TARGETDB2(DB0A);
TABLE(QUERY01)
SOURCE(FMNDB2.DEPT)
TARGET(A.B) DD:TABL0001;
SELECT A.DEPTNO
FROM FMNDB2.DEPT A
WHERE DEPTNO > 'D' AND DEPTNO < 'E';
//ABXCNTLI DD   *
//*
//ABXCNTLO DD   SYSOUT=*
//ABXPRINT DD   SYSOUT=*
```

```
//SYSABEND DD   SYSOUT=*
//*TABL0001 DD   DISP=SHR,DSN=DELAHAY.ABX.EXPORT.DEPTNO
//TABL0001  DD   DISP=(NEW,PASS),SPACE=(TRK,(1,1)),UNIT=SYSDA,
//              DSN=&&DEPTLST
/*
```

2. Three items are extracted as shown in Figure 26-87.

```
  Menu  Utilities  Compilers  Help
 ---------------------------------------------------------------------------
 BROWSE    DELAHAY.ABX.EXPORT.DEPTNO                 Line 00000000 Col 001 003
********************************* Top of Data ********************************
D01
D11
D21
****************************** Bottom of Data ******************************



 Command ===> _____________________________________________ Scroll ===> CSR
```

Figure 26-87 List of D departments

3. Next, create another external list of items such as a list of managers extracted from the FMNDB2.EMP table. Example 26-18 is a working File Export JCL step to do this extraction.

Example 26-18 Extract managers with File Export

```
//MGREXTR  EXEC PGM=ABXUTIL
//STEPLIB  DD   DISP=SHR,DSN=ADT4.ABX.SABXLOAD
//         DD   DISP=SHR,DSN=DSN710.SDSNLOAD
//*
//ABXCTL   DD   *
OPTIONS(DB2, PLAN(ABXUTIL));
SOURCEDB2(DB0A);
TARGETDB2(DB0A);
TABLE(QUERY01)
SOURCE(FMNDB2.EMP)
TARGET(A.B) DD:TABL0001;
SELECT A.LASTNAME
FROM FMNDB2.EMP A
WHERE JOB = 'MANAGER';
//ABXCNTLI DD   *
//*
//ABXCNTLO DD   SYSOUT=*
//ABXPRINT DD   SYSOUT=*
//SYSABEND DD   SYSOUT=*
//*TABL0001 DD   DISP=SHR,DSN=DELAHAY.ABX.EXPORT.MGRLST
//TABL0001  DD   DISP=(NEW,PASS),SPACE=(TRK,(1,1)),UNIT=SYSDA,
//              DSN=&&MGRLST
/*
```

4. Seven items are extracted as shown in Figure 26-88.

```
  Menu  Utilities  Compilers  Help
 ------------------------------------------------------------------------------
 BROWSE    DELAHAY.ABX.EXPORT.MGRLST                          Data Set - Browsed
******************************** Top of Data ********************************
..THOMPSON
..KWAN
..STERN
..PULASKI
..GEYER
..HENDERSON
..SPENSER
******************************* Bottom of Data *******************************

 Command ===> ____________________________________________________ Scroll ===> CSR
```

Figure 26-88 List of managers

5. Specify the relationships as shown in Figure 26-89.

```
ABX$RELS  V1R1 ------  Relationship Specification  ------ 2004/12/04  10:06:53

Session Name  SEQ001

                                                                   Row 1 of 2
Cmd  Label     Type  Dataset
_    DEPTNAME  F     DELAHAY.ABX.EXPORT.DEPTNO
_    MGRNAME   F     DELAHAY.ABX.EXPORT.MGRLST

Valid Line Commands  :  E - Edit Entry  D  - Delete Entry
Valid Option Commands:  AF - Add Filter  AC - Add Content

Option  ===> ____________________________________________________ Scroll ===> PAGE
```

Figure 26-89 List of relationships

6. Create a “Filter” relationship for each external reference file as shown in Figure 26-90.

```
ABX$RELF  V1R1 ---------  Filter Specification  --------- 2004/12/04  10:08:23

Source Name  MGRNAME
DSN          DELAHAY.ABX.EXPORT.MGRLST
Position     3
Length       15
Data Type    CHAR

Press ENTER or F3 to process, F12 to cancel



Option  ===>
```

Figure 26-90 Filter relationship

7. Finally, the sample procedure to extract from the DB2 EMP and DEPT tables employees with DEPT-NOs belonging to DEPTLST that are not in the MGRLST is presented in Example 26-19.

Example 26-19 Extract employees with File Export

```
//EMPEXTR  EXEC PGM=ABXUTIL
//STEPLIB  DD   DISP=SHR,DSN=ADT4.ABX.SABXLOAD
//         DD   DISP=SHR,DSN=DSN710.SDSNLOAD
//*
//*DEPTLST  DD   DISP=SHR,DSN=DELAHAY.ABX.EXPORT.DEPTNO
//*MGRLST   DD   DISP=SHR,DSN=DELAHAY.ABX.EXPORT.MGRLST
//DEPTLST  DD   DISP=(OLD,DELETE),DSN=&&MGRLST
//MGRLST   DD   DISP=(OLD,DELETE),DSN=&&DEPTLST
//*
//ABXCTL   DD   *
OPTIONS(DB2, PLAN(ABXUTIL));
 RELATIONSHIP DD:DEPTLST
              DEPTNAME = (1,3,CHAR);
 RELATIONSHIP DD:MGRLST
              MGRNAME = (3,15,CHAR);
SOURCEDB2(DBOA);
TARGETDB2(DBOA);
::
TABLE(QUERY01)
SOURCE(FMNDB2.DEPT)
TARGET(A.B) DD:TABL0001
ONLY WHEN DEPTNO IN (DEPTNAME)
;
SELECT A.DEPTNO,
       A.DEPTNAME,
       A.LOCATION
FROM FMNDB2.DEPT A;
THEN INSERT INTO DD:OUTL01
     VALUES ('NUMBER = ', DEPTNO,
             ' LABEL = ', DEPTNAME);
::
```

```
TABLE(QUERY02)
SOURCE(FMNDB2.EMP)
TARGET(X.Y) DD:TABL0002
CHILD OF QUERY01
WHERE QUERY02.WORKDEPT = QUERY01.DEPTNO
ONLY WHEN LASTNAME NOT IN (MGRNAME)
;
SELECT A.EMPNO,
       A.FIRSTNME,
       A.MIDINIT,
       A.LASTNAME,
       A.WORKDEPT
FROM FMNDB2.EMP A;
THEN INSERT INTO DD:OUTL02
     VALUES ('ID = ', EMPNO,
             ' NAME = ', LASTNAME,
             ' WORK IN = ', WORKDEPT);
//ABXCNTLI DD  *
//*
//ABXCNTLO DD  SYSOUT=*
//ABXPRINT DD   SYSOUT=*
//SYSABEND DD   SYSOUT=*
//OUTL01   DD   SYSOUT=*
//OUTL02   DD   SYSOUT=*
//TABL0001 DD   SYSOUT=*
//TABL0002 DD   SYSOUT=*
/*
```

The list of extracted departments is shown in Example 26-20.

Example 26-20 DB2 list of departments

```
NUMBER = D01 LABEL = DEVELOPMENT CENTER
NUMBER = D11 LABEL = MANUFACTURING SYSTEMS
NUMBER = D21 LABEL = ADMINISTRATION SYSTEMS
```

The list of employees who work in these departments and who are not managers is shown in Example 26-21.

Example 26-21 DB2 list of employees

```
ID = 000150 NAME = ADAMSON         WORK IN = D11
ID = 000160 NAME = PIANKA          WORK IN = D11
ID = 000170 NAME = YOSHIMURA       WORK IN = D11
ID = 000180 NAME = SCOUTTEN        WORK IN = D11
ID = 000190 NAME = WALKER          WORK IN = D11
ID = 000200 NAME = BROWN           WORK IN = D11
ID = 000210 NAME = JONES           WORK IN = D11
ID = 000220 NAME = LUTZ            WORK IN = D11
ID = 000230 NAME = JEFFERSON       WORK IN = D21
ID = 000240 NAME = MARINO          WORK IN = D21
ID = 000250 NAME = SMITH           WORK IN = D21
ID = 000260 NAME = JOHNSON         WORK IN = D21
ID = 000270 NAME = PEREZ           WORK IN = D21
```

26.7.4 Relationship specification: External reference

When creating a new column in the target, the user can specify an external source for the new data. As an example, there is a column for a status already and the user wants to add a

new column with a bonus value according to this status. This information can be kept in a data set as shown in Example 26-22.

Example 26-22 External content data set

```
0 CUSTOMER          -
1 PREFERRED MEMBER  -
2 GOLD MEMBER       -
3 PLATINUM MEMBER   -
```

An existing column has one of the key values (2) and the new generated column will have the corresponding value (GOLD MEMBER).

The user describes this data set by specifying the field to be extracted and the reference key for the merge as shown in Figure 26-91.

```
ABX$RELC  V1R1 -----------  Relation Content  ----------- 2004/12/03  18:00:46

Content Name  LABEL2
DSN           DELAHAY.ABX.DATA(LABEL2)

Key For Content:
  Position      1
  Length        1
  Data Type     CHAR

Data For Content:
  Position      3
  Length        18
  Data Type     CHAR

Press ENTER or F3 to process, F12 to cancel




Option  ===>
```

Figure 26-91 External content specification

It will be converted in the JCL as shown in Example 26-23.

Example 26-23 External content specification in JCL

```
RELATIONSHIP DD:ABX00002
 CONTENT LABEL2 KEY = (1,1,CHAR)
                DATA = (3,18,CHAR);
```

This new field is defined in the SELECT statement using the syntax Content_name(Reference_field) as shown in Example 26-24.

Example 26-24 New field with external content

```
SELECT REC-TYPE,NAME,AAA,LABEL2(STATUS)
       FROM RECORD WHERE NAME > 'D'
```

For DB2, a new column (BONUS) with an external source (LABEL2) with cross-reference to an existing column (STATUS) is defined, as shown in Example 26-25.

Example 26-25 New column with external content

```
COLUMN(BONUS)
  SOURCE = LABEL2(STATUS) NEW;
```

26.7.5 File Export versus File Manager in batch

File Manager processes one table at a time.

The default is to process the table on line by editing a table template. The DB2 data extraction in batch mode is not documented. The user has to generate a sample batch and adapt it to the requirements.

Without using a File Manager/DB2 template with selection criteria, it is a two-step process:

- Extract the columns from the table
- Select the records

There is no reference list for field comparison as with File Export. The user must write the explicit values or use a procedure to generate the appropriate instructions as shown in Example 26-26.

Example 26-26 Sample procedure for related DB2 data extraction using File Manager

```
//*
//* generate the File Manager SYSIN according to reference list
//* here manually generated in card DEPTLST
//*
//REXX01   EXEC PGM=IKJEFT01,DYNAMNBR=30,REGION=4096K
//LISTOUT  DD UNIT=SYSDA,SPACE=(TRK,(1,1)),DISP=(NEW,PASS),
//            DSN=&&DSCIN
//LISTIN   DD *
D11
D21
/*
//SYSEXEC  DD DISP=SHR,DSN=DELAHAY.FMN.JCL
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSTERM  DD SYSOUT=*
//SYSTSIN  DD *
 %GENEXP1
/*
//*-----------------------------------------------------------------------
//*
//* DB2 table extraction
//*
//FMNDB2   EXEC PGM=FMNDB2,PARM=('SSID=DB0A,SQID=DELAHAY')
//STEPLIB  DD DSN=DB2CUST0.DB0A.SDSNEXIT,DISP=SHR
//         DD DSN=DSN710.SDSNLOAD,DISP=SHR
//         DD DSN=DSNDB0A.RUNLIB.LOAD,DISP=SHR
//SYSPRINT DD SYSOUT=*
//FMNTSPRT DD SYSOUT=*
//SYSTERM  DD SYSOUT=*
//FMNOUT   DD UNIT=SYSDA,SPACE=(TRK,(1,1)),DISP=(NEW,PASS),
//            DSN=&&DEPTLST
//SYSIN    DD *
$$FILEM DBX OBJIN="FMNDB2"."DEPT",
$$FILEM TCIN=DELAHAY.FMN.JCL(DEPTX),
$$FILEM OUTPUT=FMNOUT,
$$FILEM TCOUT=DELAHAY.FMN.JCL(DEPTX2),
```

```
$$FILEM LANG=COBOL,
$$FILEM NULLIND=_,
$$FILEM ROWS=ALL
//*-----------------------------------------------------------------------
//*
//* File Manager processing of the DB2 extract
//*
//FM01     EXEC  PGM=FILEMGR
//DD01     DD DISP=(OLD,DELETE),DSN=&&DEPTLST
//DD010    DD SYSOUT=*
//DDUMMY   DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSIN    DD DSN=&&DSCIN,DISP=(OLD,PASS)
/*
```

Example 26-27 contains such a procedure, GENEXP1, written in REXX.

Example 26-27 Sample REXX procedure GENEXP1

```
/* REXX */
 TRACE OFF
 /* RESET                                                    */
 DEPTCNT = 1
 DEPT.DEPTCNT = ""
 /* READ INPUT DATASET LISTIN                                */
 "EXECIO * DISKR LISTIN (FINIS STEM FILEM."
 /* THE COMMON INSTRUCTIONS FOR THE FM STEP                  */
 QUEUE "$$FILEM DSC  INPUT=DD01,OUTPUT=DDUMMY, "
 QUEUE "$$FILEM TCIN=DELAHAY.FMN.JCL(DEPTX2), "
 QUEUE "$$FILEM LANG=COBOL, "
 QUEUE "$$FILEM PROC=* "
 /* BUILD AN INSTRUCTION FOR EVERY INPUT ITEM                */
 DO I = 1 TO FILEM.0
  DEPT.I = SUBSTR(FILEM.I,1,3)
  IF I = 1 THEN DO
     IF I < FILEM.0 THEN DO
        QUEUE "   IF SUBSTR(INREC,1,3)='"DEPT.I"' ! ,  "
        END
        ELSE DO
        QUEUE "   IF SUBSTR(INREC,1,3)='"DEPT.I"' "
        END
     END
     ELSE DO
        IF I < FILEM.0 THEN DO
        QUEUE "      SUBSTR(INREC,1,3)='"DEPT.I"' ! ,  "
        END
        ELSE DO
        QUEUE "      SUBSTR(INREC,1,3)='"DEPT.I"' "
        END
     END
 /* END LOOP                                                 */
 END
 /* CLOSE THE STACK                                          */
 QUEUE "   THEN DO "
 QUEUE "      OUTREC = 'NUMBER =' SUBSTR(INREC,1,3) ,"
 QUEUE "               'WHERE =' SUBSTR(INREC,5,16) ,"
 QUEUE "               'LABEL =' SUBSTR(INREC,22,36) "
 QUEUE "      WRITE(DD010)"
 QUEUE "   END"
 QUEUE ""
```

```
/* WRITE THE DSC INSTRUCTIONS TO THE SYSIN DATASET          */
"EXECIO * DISKW LISTOUT (FINIS"
EXIT RC
```

Part 7

Workload Simulator for z/OS and OS/390

In this part we discuss major steps to install, customize, and use Workload Simulator.

27

Workload Simulator and Workload Simulator Test Manager settings and customization

In this chapter we discuss the main steps to install and customize Workload Simulator (WSim) and Workload Simulator Test Manager (WTM).

Some steps in this process are usually performed by systems support personnel and some are performed by WSim and WTM users.

27.1 Installing WSim

Step-by-step instructions for WSim installation from the product tape can be found in the program directory shipped with the tape.

To use the WSim/ISPF interface, which provides user-friendly access to most of the functions and utilities of WSim, perform the following steps:

1. Concatenate:
   - **.SITPPNL data set to ISPPLIB DD
   - **.SITPMSG data set to ISPMLIB DD
   - **.SITPEXEC data set to SYSEXEC DD
   - **.SITPTBL data set to ISPTLIB DD
   - **.SITPSKEL data set to ISPSLIB DD
2. Connect the WSim/ISPF interface to the ISPF system by including an option to invoke the WSim/ISPF interface from one of the ISPF panels.

   An example is shown in Figure 27-1.

```
  LPAR F6 ---Application & Integrated Middleware Selection Panel--F6 F6--------
OPTION  ===>
                                                                       SLITVAK
   AM  AM 2.1       - Application Monitor Version 2.1
   DT  DT 5.1       - Debug Tool for z/OS Version 5.1
   FA  FAULT AZ 5.1 - Fault Analyzer Version 5.1
   WM  WSIM 1.1     - Workload Simulator Version 1.1
   WT  WSIM/TM 1.1  - WSIM Test Manager
   F5  FILE MGR 5.1 - File Manager for z/OS and OS/390 Version 5.1
   D5  FM/DB2 5.1   - File Manager/DB2 Version 5.1
   I5  FM/IMS 5.1   - File Manager/IMS Version 5.1

Press END to return to ISPF/PDF Primary Option Menu.









PF 1=HELP      2=SPLIT     3=END       4=RETURN    5=RFIND     6=RCHANGE
PF 7=UP        8=DOWN      9=SWAP     10=LEFT     11=RIGHT    12=RETRIEVE
```

Figure 27-1 ISPF panel with WSim invocation

3. Allocate and catalog data sets which will be used by WSim.

   The REXX procedure ITP0INST, which allocates all the data sets required to run the WSim/ISPF interface, is provided in **.SITPEXEC data set. This procedure also sets up the default for an installation qualifier. If this procedure is not run or is not customized, each WSim user will have to repeatedly set some default values on the WSim/ISPF interface panels.
4. Update the load library name and models qualifier on the SETUP panel of the WSim/ISPF interface. If default name “WSIM” is used or they are defined in the REXX procedure ITP0INST in the previous step, there is no need to do this.
5. Authorize WSim to run under TSO if WSim simulations will be run interactively.

Note: The models qualifier field on the SETUP panel of the WSim/ISPF interface will accept a maximum of 17 characters and its value cannot contain quotes. This must be taken into consideration when the local models qualifier value is defined.

Figure 27-2 provides an example of local customizations for some qualifiers.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------
VIEW       SLITVAK.SUPERC.LIST                              Columns 00009 00080
Command ===> ____________________________________________ Scroll ===> CSR
000009 load_library   = "'ADTOOLS.WORKLOAD.SIM11.SITPLOAD'"
000010 load_library                = "'WSIM.SITPLOAD'"
000011
000012 models_qualifier            = 'ADTOOLS.WORKLOAD.SIM11'
000013 models_qualifier            = 'WSIM'
000014
000015 dslist.1.data_set           = "'ADTOOLS.WSIM.TESTFILE'"
000016 dslist.1.data_set           = "'WSIM.TESTFILE'"
000017
000018 dslist.2.data_set           = "'ADTOOLS.WSIM.MSGFILE'"
000019 dslist.2.data_set           = "'WSIM.MSGFILE'"
000020
000021 dslist.3.data_set           = "'ADTOOLS.WSIM.STLIN'"
000022 dslist.3.data_set           = "'WSIM.STLIN'"
000023
000024 dslist.4.data_set           = "'ADTOOLS.WSIM.NETWORK'"
000025 dslist.4.data_set           = "'WSIM.NETWORK'"
000026
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 27-2 WSim installation: Sample of local qualifiers

The sample script INSTALL1 supplied with the product can be used for the verification of WSim installation.

27.2 Installing WTM

Two major tasks have to be performed after WTM data sets from the product tape are transferred:

1. Create a Virtual Telecommunications Access Method (VTAM) application major node.
2. Provide access to WTM.

These tasks are usually performed by systems support personnel.

Additional activities are required for each user of WTM. These activities may be performed by systems support personnel or, in some cases, by users themselves.

27.2.1 Creating a VTAM application major node

A VTAM application major node must be defined in SYS1.VTAMLST or its local equivalent. A local sample is shown in Example 27-1.

Example 27-1 VTAM application major node definition sample for WSim and WTM

```
* ------------------------------------------------------------------ *
*           WSIM TEST MANAGER MAJOR NODE
* ------------------------------------------------------------------ *
        VBUILD TYPE=APPL
*WTM DISPLAY MONITOR
WTMDM00  APPL  EAS=1,PARSESS=YES
WTMDM01  APPL  EAS=1,PARSESS=YES
WTMDM02  APPL  EAS=1,PARSESS=YES
   .
WTMDM09  APPL  EAS=1,PARSESS=YES
*INTERACTIVE DATA CAPTURE
ITPIDC0  APPL  EAS=1,SESSLIM=YES
ITPIDC1  APPL  EAS=1,SESSLIM=YES
ITPIDC2  APPL  EAS=1,SESSLIM=YES
   .
ITPIDC9  APPL  EAS=1,SESSLIM=YES
*WTM APPLICATIONS FOR LU SESSION SIMULATION
VAPPL00  APPL  EAS=1
VAPPL01  APPL  EAS=1
VAPPL02  APPL  EAS=1
   .
VAPPL09  APPL  EAS=1
*WTM APPLICATIONS FOR CPI-C SIMULATION
APPL0    APPL  APPC=YES
APPL1    APPL  APPC=YES
APPL2    APPL  APPC=YES
   .
APPL9    APPL  APPC=YES
```

The VTAM application major node must be activated.

Each WSim and WTM user will have to reference a unique name from the list of APPL statements for the display monitor (WTMDM0#) and interactive data capture (ITPIDC#).

27.2.2 Providing access to WTM

There are several ways to provide users with access to WTM. An entry point can be added to one of the ISPF menu panels to allow general access on the system, or individual access can be provided for the users defined to WTM.

A copy of the REXX procedure WSIMTM, which can be found in **. SITPEXEC, is used in both cases:

- For global access, concatenate a data set with a copy of the REXX procedure WSIMTM to SYSPROC DD.
- For individual access, make a copy of the REXX procedure WSIMTM into a CLIST with your own high-level qualifier.

The copy of WSIMTM must be updated with correct local WSim data set names, otherwise WTM will not work.

The WSIMTM concatenates the following data sets:

- **.SITPPNL to ISPPLIB DD

- **.SITPMSG to ISPMLIB DD
- **. SITPEXEC to SYSEXEC DD
- **. SITPTBL to ISPTLIB DD
- userid.WTMUSER.SKELS to ISPSLIB DD

27.2.3 Setting up a new WTM user

Perform the following steps for every new user of WTM:

1. Make a copy of the member ITMNUSER from the supplied **.SITPSAMP data set and modify it as follows:
   - Add a job card.
   - Change userid to match the user being set up.
   - Change High Level Index to the correct local qualifier.
2. Submit the job which will allocate the following 4 files:
   - userid.WTMUSER.IDCDFLTS - The WSim Interactive Data Capture (IDC) defaults file
   - userid.WTMUSER.IDCLOG - The WSim IDC log partitioned data set
   - userid.WTMUSER.SKELS - A copy of the WTM SKELS data set
   - userid.WTMUSER.TABLES - A data set which will contains the tables used or created by WTM
3. Log on as the user being set up and go to the WTM main panel to finalize WTM settings. The WTM main panel is presented as shown in Figure 27-3.

```
                              WSim Test Manager

Select one of the following.  Then press Enter.

      Command   Action
_  1. CASE      Create and Process Testcases
   2. GROUP     Create and Process Testgroups
   3. CYCLE     Create and Process Testcycles
   4. RUN       Create WSim Networks and Schedule WSim Simulation Runs
   D. DOC       Create Test Documentation
   P. PROJECT   Add/Change Project or Alternate HLI
   U. UTIL      Run WSim Test Manager Utilities
   W. WII       Invoke WSim/ISPF Interface

Project:                                Alternate HLI:

Command ===>
PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 27-3 WTM main panel

   - Move the cursor to the command line and enter the command **vars** or first select option **U** to get to the WTM Utilities panel, and there select option **1**. On the presented panel a number of default values are shown. Several missing fields must be populated to finish the setup.

     The WTM Variables and Options panel is shown in Figure 27-4.

– The values in the fields IDC VTAM APPL name and Display Monitor VTAM APPL name must be chosen from the corresponding values defined in the VTAM application major node definition. The value in the field VTAM Name Model must refer to the pool of values defined in VTAM application major node definition.

– Another very important field on this panel is Automatic REFRESH. If set to `Y`, the REFRESH command will run every time you log on. This helps to ensure that shared projects are kept up-to-date.

```
                  Specify WSim Test Manager Variables and Options

Update the fields, then press Enter to save the values.   Press PF3 to end.

WSim/ISPF Interface Access: HIDE     (hide, display or interact)
Panel Message Delay  : 2  seconds    (1-10)
Log Display Monitor Chars: {}`       Automatic REFRESH?       : N  (Y/N)
Display Panel ID?    : N  (Y/N)      Display Function Keys?   : Y  (Y/N)

WSim Load Library    : ADTOOLS.WORKLOAD.SIM11.SITPLOAD
IDC VTAM APPL name   : ITPIDC0       Display Monitor VTAM APPL name: WTMDM0
Fully Validate WSim Data Set Names? : Y  (Y/N)

Work Data Sets HLI   : SLITVA2
Low Level Names:  STL: STL           MSGTXTs: MSGTXTS     NTWRKS: NTWRKS

VTAM Name Model      : VAPPL##
WSim Name Model      : WSIMLU##
Numeric substitution start value: 00


PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 27-4 WTM Variables and Options panel

4. Press ENTER to save the values. The basic WTM install is finished.

Some additional values on the WTM Variables and Options panel will have to be supplied later.

27.2.4 Latest enhancements

To use the latest enhancements for the Workload Simulator, apply the PTF which fixes APAR PQ94132.

To use the 3270 password masking with WSim utilities and reports, run the job with the ITPGNKYZ utility, which generates a SMP/E USERMODE that sets a site-unique encryption key and initialization vector value.

To use the WSim Adapters for Rational TestManager, which allow WTM existing schedules and custom JCL scripts to be launched from the Rational TestManager running on a remote workstation, complete the following steps:

1. On the host system:
   a. Run the ITMSEQ CLIST member to apply changes to all schedules to be used by the Rational TestManager, for example, TSO EXEC ‘HLI.SITPEXEC(ITMSEQ)’
   b. From the WTM main panel enter the command **vars**, and press PF3 when the Variables and Options panel is displayed. A file VARSINFO will be created.
   c. From the WTM main panel enter the command **project**, and press PF3 when the Process Projects panel is displayed. A file PROJLIST is created.

2. On every remote workstation to be used:
   a. Run the install program WSimAdaptersRTMinstall.exe, which comes with the PTF.
   b. Register the WSim Adapters as new test script types with the Rational TestManager.
   c. Create new local test script sources for WTM schedules and JCL scripts.

28

Workload Simulator Test Manager

The Workload Simulator Test Manager (WSim Test Manager or WTM) is a usability enhancement that provides guidance through the test process. WTM offers selectable modes of operation, test management services, automatic script generation, and task automation.

In this chapter we describe how WTM can be used. We step through the process of preparing simulations using Interactive Data Capture (IDC), running simulations and viewing some reports that are provided by WTM.

We also discuss ways to organize test cases into test groups, and how to employ user tables to modify user inputs.

28.1 Concepts

The primary concept of the WTM testing structure is a *project*, which is a set of libraries containing schedules and test scenarios. Projects can be archived and reused. A project must be created before any schedules or test scenarios can be created using WTM.

Test scenarios are organized into three levels: a test case, test group and test cycle. A test group is an ordered list of test cases. Test cases can be reused within multiple test groups. A test cycle is an ordered group of test groups and test cases.

WTM offers various ways to automate the development of test cases, which are WSim scripts written in Structured Translator Language (STL).

Generated test cases are paired with network resource definitions as part of developing WTM schedules. A WTM schedule is a WSim network definition and the associated test scenario definition. The WTM schedule is used by WTM to define and control the WSim simulation run (test). WTM schedules can be archived and reused.

Figure 28-1 shows how WTM helps to manage and organize tests.

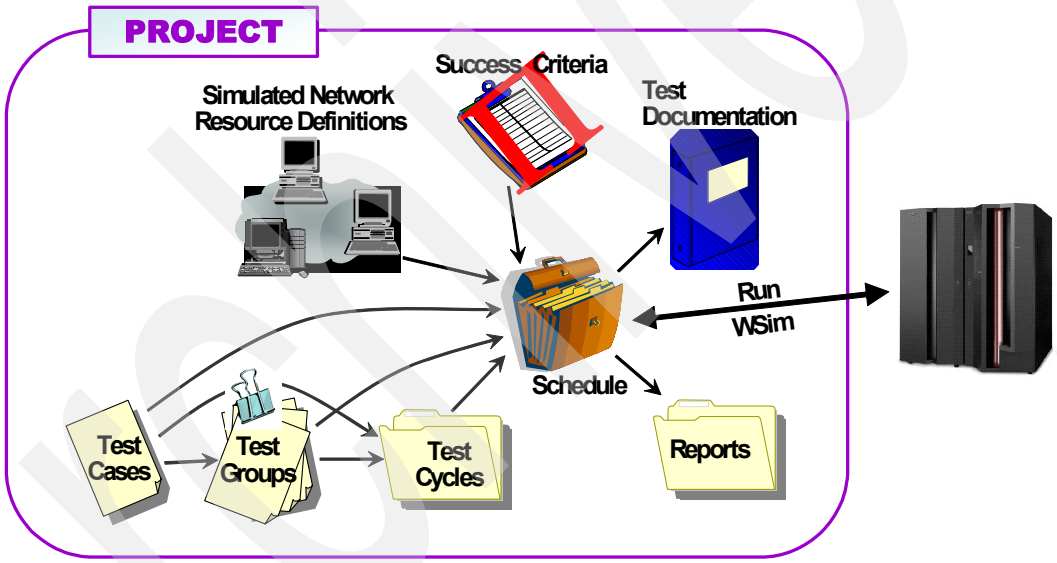


Figure 28-1 WTM helps to manage and organize tests

28.2 WTM operating modes

WTM has three modes of operation that control the amount of interaction you can have with WSim:

- **Hide mode** (the default): Most of the WTM and WSim interactions are hidden.
- **Display mode**: You can observe the entire WSim and WTM process but you have no direct control over them.
- **Interact mode**: You have control over WSim and WTM and must press the Enter key to start and execute various WSim utilities, and press the PF3 key to exit.

The operating mode can be entered on the command line (**hide**, **display** or **interact**) of any WTM panel or by selecting the VARS option from the WTM Utilities panel shown in Figure 28-2.

```
                              Access Utilities

Select one of the following.  Then press Enter.

      Command   Action
_  1. VARS      Specify WSim Test Manager Variables and Options
   2. PUTBL     Process User Tables
   3. SPACE     Specify data set attributes

   4. SKELS     Edit WSim Test Manager skeletons
   5. MSGTXT    Edit MSGTXTs data set
   6. NTWRK     Edit NTWRK data set
   7. WTMVARS   Edit WSim Test Manager variable @INCLUDE member

   8. TRANSLATE Translate all STL programs
   9. REFRESH   Refresh all WTM tables



Command ===> _____________________________________________________________

PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap      10=Left     11=Right    12=Retrieve
```

Figure 28-2 WTM Utilities panel

WTM also has a number of utilities for accessing and updating control information, skeletons, and data set attributes. STL translate and table refresh functions are also accessible. Select option **U** from the WTM main panel, or enter UTILS on the command line to access this panel.

There are two additional fields on the WTM Variables and Options panel which are important:

- **Panel Message Delay**: Specify the minimum time in seconds a panel message will be displayed.
- **Automatic REFRESH**: If set to **Y**, the REFRESH command will run every time you log on.

This helps to ensure that shared projects are kept up-to-date.

28.3 Creating a project

Perform the following steps to create a project:

1. To work with project definitions, select option **P** on the WTM main panel as shown in Figure 28-3.

```
                              WSim Test Manager

Select one of the following.  Then press Enter.

      Command   Action
p _1. CASE      Create and Process Testcases
   2. GROUP     Create and Process Testgroups
   3. CYCLE     Create and Process Testcycles
   4. RUN       Create WSim Networks and Schedule WSim Simulation Runs
   D. DOC       Create Test Documentation
   P. PROJECT   Add/Change Project or Alternate HLI
   U. UTIL      Run WSim Test Manager Utilities
   W. WII       Invoke WSim/ISPF Interface

Project:                                Alternate HLI:





Command ===> ____________________________________________________________
PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-3 Option P to work with project definitions

2. To create a new project enter the command **add** on the next panel as shown in Figure 28-4.

```
                                Process Projects
Command==> add_                                          Press PF3 to end.

Change the primary and alternate high level index fields as required
(for a list of projects, enter ? in the Project field).

To create a new project, enter the command add
To delete an entire project, enter the command delete


Project      : __________

Alternate HLI: ___________________________
   Alternate HLI Userid: _______





PF 1=Help      2=Split     3=End       4=          5=Add       6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

*Figure 28-4 Command **add** new project*

3. The next panel provides an opportunity to define the name for the project and High Level Index (HLI) and add the description, as shown in Figure 28-5.

```
                                Process Projects
                              Add New Project

 Type Project Name, Description and High Level Index.  Then press Enter.

 Project Name    : resproj1
 Description     : First project for the residency - testing application
 High Level Index: slitvak.wtmtest.resproj1

 PF 1=Help      2=Split     3=End       4=          5=          6=
 PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve

PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-5 New project name, description, and HLI

4. Make sure that there are no existing data sets that have this High Level Index, otherwise WTM may write over the existing data sets. WTM creates project data sets and displays the confirmation as shown in Figure 28-6.

```
                                Process Projects                Project Created
Command==> _                                                   Press PF3 to end.

Change the primary and alternate high level index fields as required
(for a list of projects, enter ? in the Project field).

To create a new project, enter the command add
To delete an entire project, enter the command delete


Project      : RESPROJ1   First project for the residency - testing applicatio

Alternate HLI: ____________________________
   Alternate HLI Userid: _______

PF 1=Help      2=Split     3=End       4=          5=Add       6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-6 Project created

5. The alternate HLI (high level index) fields enable a user to use test items (test cases, MSGTXTS, and schedules) created under another high level index. These test assets can belong to the same user or somebody else. If an alternate high level index is specified, all test assets for that index are flagged with ALT. The user cannot modify any of the ALT test assets since the data sets containing the ALT test assets are available in BROWSE mode only.

   Both the Alternate HLI and the Alternate HLI Userid fields must be completed to use this feature. Enter the high level index of the other project in the Alternate HLI field, and enter the owning user ID of the alternate high level index in the Alternate HLI Userid field.

6. Using ISPF option **3.4**, the user can see that WTM created 16 data sets (9 partitioned and 7 sequential) for the project RSPROJ1 with the names starting with the supplied HLI, as shown in Figure 28-7.

```
  Menu  Options  View  Utilities  Compilers  Help
 ------------------------------------------------------------------------------
 DSLIST - Data Sets Matching SLITVAK.WTMTEST.**                     Row 1 of 16
 Command ===> ____________________________________________ Scroll ===> CSR

 Command - Enter "/" to select action                 Message          Volume
 ------------------------------------------------------------------------------
          SLITVAK.WTMTEST.RESPROJ1.CONTROL                                STF611
          SLITVAK.WTMTEST.RESPROJ1.CREP                                   STF608
          SLITVAK.WTMTEST.RESPROJ1.DOC                                    STF608
          SLITVAK.WTMTEST.RESPROJ1.MODELS                                 STF611
          SLITVAK.WTMTEST.RESPROJ1.MSGTXTS                                STF608
          SLITVAK.WTMTEST.RESPROJ1.NTWRKS                                 STF608
          SLITVAK.WTMTEST.RESPROJ1.SCRIPT                                 STF611
          SLITVAK.WTMTEST.RESPROJ1.SGENSTL                                STF608
          SLITVAK.WTMTEST.RESPROJ1.STL                                    STF611
          SLITVAK.WTMTEST.RESPROJ1.VTAMLST                                STF608
          SLITVAK.WTMTEST.RESPROJ1.WTMTAB.CASE                            STF611
          SLITVAK.WTMTEST.RESPROJ1.WTMTAB.CREP                            STF611
          SLITVAK.WTMTEST.RESPROJ1.WTMTAB.CYCLE                           STF608
          SLITVAK.WTMTEST.RESPROJ1.WTMTAB.GROUP                           STF608
          SLITVAK.WTMTEST.RESPROJ1.WTMTAB.SCHED                           STF608
          SLITVAK.WTMTEST.RESPROJ1.WTMTAB.UTBL                            STF611
 ****************************** End of Data Set list ******************************
```

Figure 28-7 List of project data sets

WTM requests a table refresh. A normal table refresh is sufficient in most cases; the user must select an extended table refresh only if changes were made to the generated STL scripts or network definition statement.

28.4 Test cases

Test cases are STL programs with supporting documentation and User Data Tables (UTBLs). Select option **1**, or enter the command `CASE` from the WTM main panel to see the list of test cases for the current project. This list consists of the following columns:

- **Command code**: A one-character command field (valid values are D, N, O, T, U, and V).
- **Name**: The name of the test case and the member name of the corresponding STL program.
- **Type**: The type of the test case. Filled by WTM. The values are V for VTAMAPPL, C for CPI-C, T for TCP/IP and *-* for undefined, for example, a skeleton or an STL program created outside of WTM.
- **UTBLs**: The number of UTBLs for this test case.
- **Notes**: An asterisk indicates that there are notes for this test case.
- **Description**: An optional free-form description of the test case.

For each test case, the following commands can be entered in the command field:

D **Delete**: Delete the test case and the associated members in WTM libraries.

N **Notes**: Edit the Notes data set for this test case.

O **Open**: Edit the STL source for this test case. WTM automatically translates the changed STL program into a message generation deck (MSGTXT).

T **Translate**: Translate the STL source code into MSGTXT.

U **UTBLs**: Display the list of UTBLs for this test case. The STL source for UTBLs in the list can be edited or browsed.

V **View**: If the original WSim or IDC log has not been overwritten, this option shows some or all of the screen images for this test case.

To create a new test case, enter the command **add**, or press PF5 as shown in Figure 28-8.

```
                              Process Testcases
Command==> add
Enter a line command or add to create a new Testcase.        Press PF3 to end.
Line commands: D Delete, N Notes, O Open, T Translate, U UTBLs, V View.

  Name    Type UTBLs Notes Description
****************************** Bottom of data ******************************



PF 1=Help      2=Split     3=End       4=          5=Add       6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-8 Adding a new test case

There are six ways to create a new test case, as shown in Figure 28-9:

1. Use the Interactive Data Capture (IDC) utility to actually log on to the application and capture your session.
2. Use a System Network Architecture (SNA) trace file that was previously saved.
3. Use a log file that was generated during one of the previous runs of the IDC utility.
4. Use one of the WSim model scripts.
5. Use one of the STL skeleton scripts.
6. Use an SNA trace file that was previously saved into a Common Programming Interface for Communications (CPI-C) script.

```
                                Process Testcases
                                Add New Testcase

 Type Testcase Name, Description and Source.  Then press Enter.

 Testcase Name: tsttrad
 Description  : Logon,trader,logon,getquote,buystock,getquote,logoff,logoff_
 Source       : 1  1. Add a 3270 testcase using IDC
                   2. Add a 3270 testcase using an SNA trace
                   3. Add a 3270 testcase using a WSim or IDC log
                   4. Add a testcase using the WSim STL models
                   5. Add a testcase using an STL skeleton
                   6. Add a CPI-C testcase using an LU 6.2 SNA trace

 PF 1=Help      2=Split     3=End       4=          5=          6=
 PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve

 PF 1=Help      2=Split     3=End       4=          5=          6=
 PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-9 Add New Test case panel

In this section we describe how to create a new test case TSTTRAD using the IDC utility. The test case will do the following:

1. Connect to CICS region CICSC22F. This is the corresponding VTAM name. Refer to the parameter APPLID of the CICS system.
2. Go to CICS native mode.
3. Start transaction **test.**
4. Invoke option **7.**
5. Log on to the trader application.
6. Select a company.
7. Get a quote.
8. Buy some stock.
9. Get a quote.
10. Log off from the trader application.
11. Log off from CICS using **cesf** transaction.

During the capture process WTM will display some “Milestones” panels to show steps and status of WSim tasks. The selection of the WTM operating mode affects what WTM will display:

- Hide: Only “Milestones” panels.
- Display: WSim panels also.
- Interactive: Interaction with the WSim panels is possible.

The capture described in this section is performed in the Hide mode.

Use the following steps to create a test case using the IDC utility, as shown in Figure 28-10.

```
                                Process Testcases
Comman ┌──────────────────────────── "Milestones" ────────────────────────┐
Enter  │_                                                                  │ F3 to end.
Line c │ These are the steps to create a testcase using IDC:               │ w.
       │                                                                   │
  Name │Step 1 - ITPIDC   - Invoke IDC.                                    │
****** │ Step 2 - ITPLSGEN - Create STL program from the IDC Log.          │ **********
       │ Step 3 - ITMDCAS  - Add WSim/TM upgrades to STL program.          │
       │ Step 4 - ITPSTL   - Translate STL program into a MSGTXT.          │
       │                                                                   │
       └───────────────────────────────────────────────────────────────────┘
```

Figure 28-10 Test case creation steps when using the IDC utility

1. Start a session with a host application to capture data as shown in Figure 28-11.

```
IDCMAIN          WSim Interactive Data Capture (IDC) Utility

Select one of the following, then press Enter.

1 1. Start a session with a host application and capture data

  2. Generate an STL program from captured data
  3. Generate a message generation deck from captured data

  4. End the IDC utility program




WSim Version 1 Release 1.0.0  Program Number 5655-I39

Licensed Materials - Property of IBM
5655-I39 (C) Copyright IBM Corporation 1976, 2002.  All Rights Reserved.
US Government Users Restricted Rights - Use, duplication or disclosure
restricted by GSA ADP Schedule Contract with IBM Corporation.
F1=Help  F3=Exit  F12=Cancel
```

Figure 28-11 Starting the first step to capture data

2. Press Enter and the WSim IDC start panel will be displayed as shown in Figure 28-12.

```
IDCSSP          WSim IDC:  Start Session with Host Application

Type information, then press Enter.

Session Data
  Host application name . . . . . . . cicsc22f
  Logon mode name . . . . . . . . . . LSX32702 (Optional)
  Logon user data . . . . . . . . . . ________________________ (Optional)


IDC log data set name . . . . . . . . SLITVAK.WTMUSER.IDCLOG(TSTTRAD)
  If data set already exists, specify R        (R=Replace or A=Append)

Start capturing data immediately? . . Y        (Y=Yes or N=No)

IDC Escape key  . . . . . . . . . . . PA1__    (PAn, PFnn, CLEAR, or ATTN)





F1=Help  F3=Exit  F5=Refresh  F11=Save  F12=Cancel
```

Figure 28-12 WSim IDC start panel

3. You connect to the application CICSC22F, the IDC log is captured in the member TSTTRAD (which will be replaced if it existed earlier), data capture starts immediately, and an IDC escape key is defined as PA1. Press Enter. The CICS sign-on panel is displayed as shown in Figure 28-13.

```
BM'S  INTERNAL  SYSTEMS  MUST  ONLY  BE  USED  FOR  CONDUCTING
BM'S BUSINESS OR FOR PURPOSES AUTHORIZED BY IBM MANAGEMENT



ype your userid and password, then press ENTER:

       Userid . . . . slitvak    Groupid . . . ________
       Password . . .
       Language . . . __

   New Password . . .



FHCE3520 Please type your userid.
3=Exit
   a
```

Figure 28-13 CICS sign on panel

4. The user ID and the password are keyed in and sign on is completed, as shown in Figure 28-14.

```
DFHCE3549 Sign-on is complete (Language ENU).
```

Figure 28-14 CICS sign on is complete

5. Start the **test** transaction, which will provide you with several options as shown in Figure 28-15.

```
test
```

Figure 28-15 Test transaction entered

6. Choose option **7** to start the trader application as shown in Figure 28-16.

```
                    Sample Program - CPU / VSAM / DB2
                          for Application Monitor


                  Test Selection:

                        1. CPU  Loop
                        2. VSAM Loop
                        3. DB2  Loop
                        4. IMS  Loop

                        5. Cobol Multi-Csects program
                        6. Cobol and PL/I Multi-Csects program

                        7. Named Set and Transaction Routing
                           start CICS/DB2 remote application
                        8. Named Set and Dynamic Program Link
                        9. Named Set and Function Shipping

                 Select a test (value between 1 and 9):  7


--------------------------------------------------------------------------------
PF3=Return                                                            PF12=Exit
```

Figure 28-16 Test transaction option 7 selected

7. The trader application is started. Sign on, supplying a user name and password, as shown in Figure 28-17.

```
                 Share Trading Demonstration                    TRADER.T001

                    Share Trading Manager: Logon




              Enter your User Name:  zz


              Enter your Password:





--------------------------------------------------------------------------------
PF3=Exit                                                              PF12=Exit
```

Figure 28-17 Trader application logon panel

8. The company selection panel of the trader application is displayed as shown in Figure 28-18. Company number **1** is selected.

```
                    Share Trading Demonstration                    TRADER.T002

                 Share Trading Manager: Company Selection

                       1. Casey_Import_Export

                       2. Casey_Import_Export

                       3. Casey_Import_Export

                       4. Casey_Import_Export



               Please select a company (1,2,3 or 4) :  1



 ------------------------------------------------------------------------------
 PF3=Return                                                          PF12=Exit
```

Figure 28-18 Trader application company selection panel

9. Option **1** for a real-time quote is chosen as shown in Figure 28-19.

```
                    Share Trading Demonstration                    TRADER.T003

                 Share Trading Manager: Options

                       1. New Real-Time Quote

                       2. Buy Shares

                       3. Sell Shares



               Please select an option (1,2 or 3):  1



 ------------------------------------------------------------------------------
 PF3=Return                                                          PF12=Exit
```

Figure 28-19 trader application options panel

10. The panel with the results of the query is presented as shown in Figure 28-20.

```
                    Share Trading Demonstration                    TRADER.T004

                  Share Trading Manager: Real-Time Quote

      User Name:     ZZ

      Company Name:  Casey_Import_Export

      Share Values:                    Commission Cost:
         NOW:           70.00            for Selling:        010
         1 week ago:    72.00            for Buying:         005
         6 days ago:    71.00
         5 days ago:    70.00
         4 days ago:    69.00
         3 days ago:    68.00
         2 days ago:    68.00         Number of Shares Held: 0010
         1 day ago:     69.00         Value of Shares Held:         700.0



Request Completed OK
--------------------------------------------------------------------------------
PF3=Return                                                            PF12=Exit
```

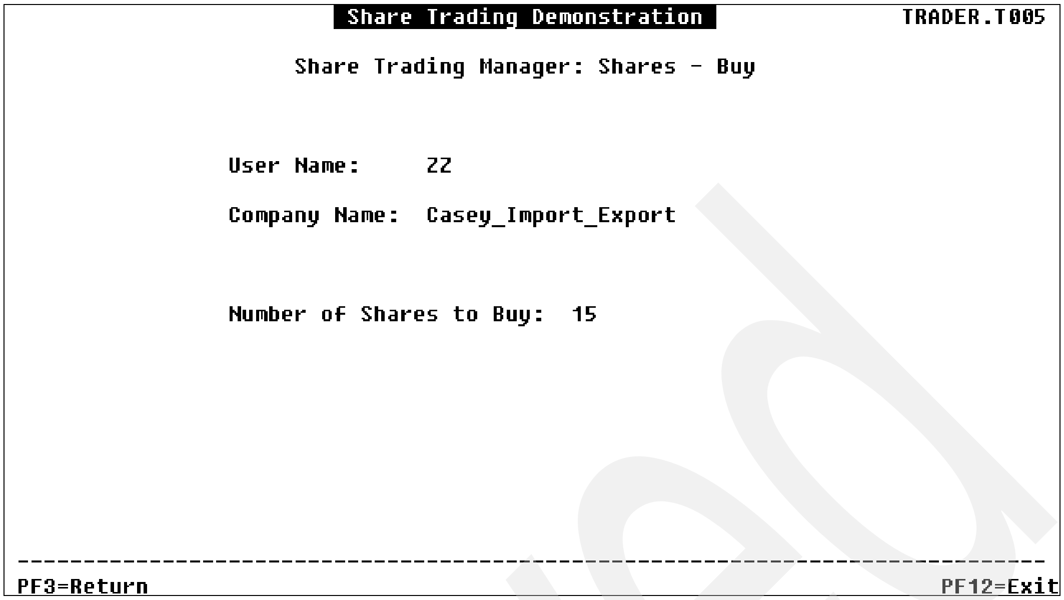
Figure 28-20 Trader application quote results

11. After pressing PF3 to return to the Options panel of the trader application, select the option to buy some shares as shown in Figure 28-21.

```
                    Share Trading Demonstration                    TRADER.T003

                  Share Trading Manager: Options


                   1. New Real-Time Quote

                   2. Buy Shares

                   3. Sell Shares




              Please select an option (1,2 or 3):  2



--------------------------------------------------------------------------------
PF3=Return                                                            PF12=Exit
```

Figure 28-21 Trader application option 2 selected

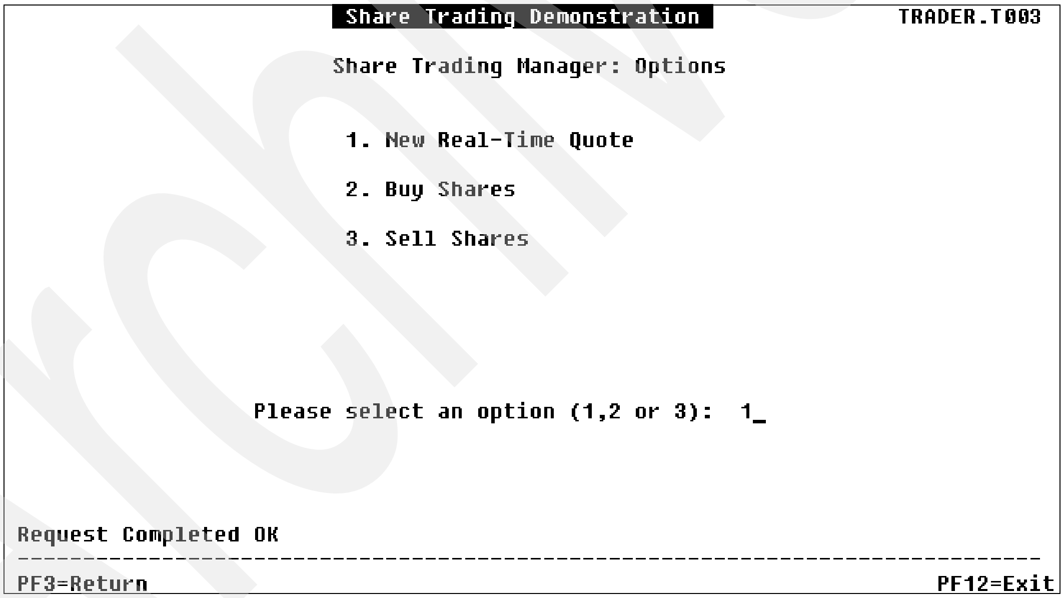
12. For this example, buy 15 shares, as shown in Figure 28-22.



```
                    Share Trading Demonstration                TRADER.T005

                 Share Trading Manager: Shares - Buy


             User Name:     ZZ

             Company Name:  Casey_Import_Export


             Number of Shares to Buy:  15








------------------------------------------------------------------------------
PF3=Return                                                           PF12=Exit
```

Figure 28-22 Trader application buying shares

13. Issue a second request for a real-time quote as shown in Figure 28-23.



```
                    Share Trading Demonstration                TRADER.T003
                    Share Trading Manager: Options

                      1. New Real-Time Quote

                      2. Buy Shares

                      3. Sell Shares




               Please select an option (1,2 or 3):  1_



Request Completed OK
------------------------------------------------------------------------------
PF3=Return                                                           PF12=Exit
```

Figure 28-23 Trader application quote request

14. The results confirm the buy of 15 additional shares and the number of shares held is now 25, as shown in Figure 28-24.

```
                         Share Trading Demonstration                TRADER.T004

                    Share Trading Manager: Real-Time Quote

      User Name:     ZZ

      Company Name:  Casey_Import_Export

      Share Values:                     Commission Cost:
         NOW:           70.00             for Selling:        010
         1 week ago:    72.00             for Buying:         005
         6 days ago:    71.00
         5 days ago:    70.00
         4 days ago:    69.00
         3 days ago:    68.00
         2 days ago:    68.00          Number of Shares Held: 0025
         1 day ago:     69.00          Value of Shares Held:        1750.0



Request Completed OK
--------------------------------------------------------------------------------
PF3=Return                                                              PF12=Exit
```

Figure 28-24 Trader application new quote results

15. To finish working with the trader application press PF12, the session is over as shown in Figure 28-25.

```
Trader: Session Over     Share Trading Demonstration                TRADER.T004

                    Share Trading Manager: Real-Time Quote

      User Name:     ZZ

      Company Name:  Casey_Import_Export

      Share Values:                     Commission Cost:
         NOW:           70.00             for Selling:        010
         1 week ago:    72.00             for Buying:         005
         6 days ago:    71.00
         5 days ago:    70.00
         4 days ago:    69.00
         3 days ago:    68.00
         2 days ago:    68.00          Number of Shares Held: 0025
         1 day ago:     69.00          Value of Shares Held:        1750.0



Request Completed OK
--------------------------------------------------------------------------------
PF3=Return                                                              PF12=Exit
```

Figure 28-25 Trader application session over

16. Transaction **cesf** is entered to exit CICS, as shown in Figure 28-26.

```
cesf
```

Figure 28-26 CICS transaction cesf entered

17. The CICS session is finished with the message shown in Figure 28-27.

```
DFHCE3590 Sign-off is complete.
```

Figure 28-27 CICS session ended

18. By pressing PA1, which was defined as the IDC control key, you are returned to the WSim IDC utility Escape Actions panel; the choice is made to finish the session with the host application as shown in Figure 28-28.

```
IDCESCA              WSim IDC:  Escape Actions

Select one of the following, then press Enter.
Note:  Options 4-9 do not change the current data capture status.

3 1. Start capturing data
  2. Stop capturing data
  3. End the session with the host application

  4. Add STL statements directly to the IDC log
  5. Add WSim scripting language statements directly to the IDC log

  6. Change IDC log data sets
  7. Reset logging to the beginning of the data set or appended data

  8. Pass the escape key to the host application
  9. Change the IDC escape key

Data capture status . . : ON
Current IDC log data set: SLITVAK.WTMUSER.IDCLOG(TSTTRAD)
Current escape key  . . : PA1


F1=Help  F3=Exit  F12=Cancel
```

Figure 28-28 Escape actions panel option 3 selected

19. The WSim IDC utility confirms the end of the session with CICSC22F as shown in Figure 28-29.

```
IDCSSP          WSim IDC:  Start Session with Host Application

Type information, then press Enter.

Session Data
  Host application name . . . . . . . CICSC22F
  Logon mode name . . . . . . . . . . LSX32702 (Optional)
  Logon user data . . . . . . . . . . ________________________ (Optional)


IDC log data set name . . . . . . . . SLITVAK.WTMUSER.IDCLOG(TSTTRAD)
  If data set already exists, specify R        (R=Replace or A=Append)

Start capturing data immediately? . . Y        (Y=Yes or N=No)

IDC Escape key  . . . . . . . . . . . PA1__    (PAn, PFnn, CLEAR, or ATTN)




ITP1508I SESSION ENDED WITH APPLICATION CICSC22F
F1=Help  F3=Exit  F5=Refresh  F11=Save  F12=Cancel
```

Figure 28-29 Session ended message

20. WTM completes the remaining steps as shown on the milestones panel (Figure 28-10 on page 683) and the new test case TSTTRAD is created. The line command **v** is entered to view the test case, as shown in Figure 28-30.

```
                              Process Testcases                      Row 1 of 1
Command==>
Enter a line command or add to create a new Testcase.         Press PF3 to end.
Line commands: D Delete, N Notes, O Open, T Translate, U UTBLs, V View.

  Name    Type UTBLs Notes Description
v TSTTRAD   V              Logon,trader,logon,getquote,buystock,getqu Added
****************************** Bottom of data ******************************

PF 1=Help       2=Split     3=End       4=          5=Add       6=
PF 7=Up         8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-30 View test case command

21. The test case TSTTRAD contains 33 display panels. Figure 28-30 shows the WTM panel that presents the choice to view the test case as a slide show with a logged or fixed time for each panel to be displayed, or to view just the index of the panels. The index has been selected in this example.

```
                              Process Testcases                      Row 1 of 1
                            Panel Display Options

 Press ENTER to continue or PF3 to cancel.

 Index or View  (I/V): i

 For View only:
   Logged or Fixed Delay (L/F): F
   Fixed Delay Value in Seconds: 1
   Display Panels    :    1 to 33


 PF 1=Help       2=Split     3=End       4=          5=          6=
 PF 7=Up         8=Down      9=Swap     10=Left     11=Right    12=Retrieve

PF 1=Help       2=Split     3=End       4=          5=          6=
PF 7=Up         8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-31 Index of display panels requested

22. The index of panels in the test case TSTTRAD is presented in two parts, shown in Figure 28-32 and Figure 28-33.

```
                          Resource Display Index                 Row 1 of 33
Command==>
Line commands: V to view panel, S to edit STL, D to delete.   Press PF3 to end.

  Timestamp AID    Testcase Panel
_ 17042428         TSTTRAD                                Signon to CIC
_ 17042428  ENTER  TSTTRAD                                Signon to CIC
_ 17043410         TSTTRAD                                Signon to CIC
_ 17043410  ENTER  TSTTRAD                                Signon to CIC
_ 17054622         TSTTRAD
_ 17054622  CLEAR  TSTTRAD
_ 17080947         TSTTRAD
_ 17080947  ENTER  TSTTRAD  test
_ 17130102         TSTTRAD                           Sample Program - CP
_ 17130102  ENTER  TSTTRAD                           Sample Program - CP
_ 17143619         TSTTRAD                               Share Trading D
_ 17143619  ENTER  TSTTRAD                               Share Trading D
_ 17185569         TSTTRAD                               Share Trading D
_ 17185569  ENTER  TSTTRAD                               Share Trading D
_ 17205391         TSTTRAD                               Share Trading D
_ 17205391  ENTER  TSTTRAD                               Share Trading D
_ 17221004         TSTTRAD                               Share Trading D
PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-32 Index of panels for test case (Part 1 of 2)

```
                          Resource Display Index                Row 18 of 33
Command==> _
Line commands: V to view panel, S to edit STL, D to delete.   Press PF3 to end.

  Timestamp AID    Testcase Panel
_ 17221004  PF3    TSTTRAD                                Share Trading D
_ 17235925         TSTTRAD                                Share Trading D
_ 17235925  ENTER  TSTTRAD                                Share Trading D
_ 17511250         TSTTRAD                                Share Trading D
_ 17511250  ENTER  TSTTRAD                                Share Trading D
_ 17522571         TSTTRAD                                Share Trading D
_ 17522571  ENTER  TSTTRAD                                Share Trading D
_ 17533785         TSTTRAD                                Share Trading D
_ 17533785  PF12   TSTTRAD                                Share Trading D
_ 17545699         TSTTRAD  Trader: Session Over          Share Trading D
_ 17545699  CLEAR  TSTTRAD
_ 17562011         TSTTRAD
_ 17562011  ENTER  TSTTRAD  cesf
_ 17580128         TSTTRAD
_ 17580128  CLEAR  TSTTRAD
_ 18065529         TSTTRAD
****************************** Bottom of data ******************************
PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-33 Index of panels for test case (Part 2 of 2)

23. The user can view the panels or the generated STL program using line commands. The beginning of the STL program is shown in Figure 28-34.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help

EDIT       SLITVAK.WTMTEST.RESPROJ1.STL(TSTTRAD) - 01.01   Columns 00001 00072
Command ===> ____________________________________________________ Scroll ===> PAGE
****** **************************** Top of Data ******************************
000001 @program=TSTTRADT
000002 @include wtmvars
000003 TSTTRAD: msgtxt
000004 /*-------------------------------------------------------------------*/
000005 /* ITPIDC: DISPLAY=TSOCON   APPLICATION=CICSC22F 17:01:08.59 10/25/04*/
000006 /*   ----------- DISPLAY CHARACTERISTICS AND FEATURES ------------   */
000007 /*   ALTCSET=APL                APLCSID=(963,310)                    */
000008 /*   BASECSID=(697,37)          CCSIZE=(8,15)     COLOR=MULTI        */
000009 /*   DBCS=NO                                                         */
000010 /*   DISPLAY=(24,80,24,80)      DLOGMOD=LSX32702  EXTFUN=YES         */
000011 /*   FLDOUTLN=NO                FLDVALID=NO       HIGHLITE=YES       */
000012 /*   MAXNOPTN=0                 PS=NONE           UOM=INCH           */
000013 /*-------------------------------------------------------------------*/
000014 /* ITPLSGEN: SCRIPT GENERATION PARAMETERS        18:08:14.97 10/25/04*/
000015 /*   INPUT    SLITVAK.WTMUSER.IDCLOG(TSTTRAD)                        */
000016 /*   OUTPUT   SLITVAK.WTMTEST.RESPROJ1.STL                           */
000017 /*   MSGTXT   TSTTRAD                                                */
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 28-34 Fragment of STL program

24. The STL program generated for the TSTTRAD test case is too long to present screen-by-screen, so a portion is presented in Example 28-1.

Example 28-1 The STL program generated for TSTTRAD (password fields are edited)

```
@program=TSTTRADT
@include wtmvars
TSTTRAD: msgtxt
/*-------------------------------------------------------------------*/
/* ITPIDC: DISPLAY=TSOCON   APPLICATION=CICSC22F 17:01:08.59 10/25/04*/
/*   ----------- DISPLAY CHARACTERISTICS AND FEATURES ------------   */
/*   ALTCSET=APL                APLCSID=(963,310)                    */
/*   BASECSID=(697,37)          CCSIZE=(8,15)     COLOR=MULTI        */
/*   DBCS=NO                                                         */
/*   DISPLAY=(24,80,24,80)      DLOGMOD=LSX32702  EXTFUN=YES         */
/*   FLDOUTLN=NO                FLDVALID=NO       HIGHLITE=YES       */
/*   MAXNOPTN=0                 PS=NONE           UOM=INCH           */
/*-------------------------------------------------------------------*/
/* ITPLSGEN: SCRIPT GENERATION PARAMETERS        18:08:14.97 10/25/04*/
/*   INPUT    SLITVAK.WTMUSER.IDCLOG(TSTTRAD)                        */
/*   OUTPUT   SLITVAK.WTMTEST.RESPROJ1.STL                           */
/*   MSGTXT   TSTTRAD                                                */
/*   NODELAY                                                         */
/*   GENERATE CHANGED                                                */
/*   LU       IDCSLU-1                                               */
/*   STL      TRACE=TSTTRADT                                         */
/*   NOVERIFY                                                        */
/*-------------------------------------------------------------------*/

/*---------------------------------------------------- 17010859 00001 */
onin0001: onin substr(ru,1,1) = 'F5'x,
              then found = on
found = off
initself('CICSC22F','LSX32702')
do while found = off             /* wait for onin0001 data received */
 wait until onin
end
```

```
deact onin0001

/* 17:01:08.63 ITP1507I SESSION STARTED WITH APPLICATION CICSC22F */

/*---------------------------------------------------- 17042428 00001 */
WTM_panel_ID = 'PNL00001'
log 'WTM_panel_ID' WTM_panel_ID
cursor(10,26)
ereof
charset 'field'
type 'slitvak'
cursor(11,26)
type '********'
cursor(12,26)
transmit using enter

/*---------------------------------------------------- 17043410 00003 */
WTM_panel_ID = 'PNL00002'
log 'WTM_panel_ID' WTM_panel_ID
cursor(11,26)
ereof
charset 'field'
type '*******'
transmit using enter

/*---------------------------------------------------- 17054622 00005 */
WTM_panel_ID = 'PNL00003'
log 'WTM_panel_ID' WTM_panel_ID
transmit using clear

/*---------------------------------------------------- 17080947 00007 */
WTM_panel_ID = 'PNL00004'
log 'WTM_panel_ID' WTM_panel_ID
cursor(1,1)
charset 'field'
type 'test'
transmit using enter

/*---------------------------------------------------- 17130102 00009 */
WTM_panel_ID = 'PNL00005'
log 'WTM_panel_ID' WTM_panel_ID
cursor(20,60)
charset 'field'
type '7'
cursor(20,60)
transmit using enter

/*---------------------------------------------------- 17143619 00011 */
WTM_panel_ID = 'PNL00006'
log 'WTM_panel_ID' WTM_panel_ID
cursor(9,42)
ereof
charset 'field'
type 'zz'
cursor(13,42)
ereof
type 'zz'
transmit using enter

/*---------------------------------------------------- 17185569 00013 */
```

```
WTM_panel_ID = 'PNL00007'
log 'WTM_panel_ID' WTM_panel_ID
cursor(17,60)
charset 'field'
type '1'
cursor(17,60)
transmit using enter

/*---------------------------------------------------- 17205391 00015 */
WTM_panel_ID = 'PNL00008'
log 'WTM_panel_ID' WTM_panel_ID
cursor(17,57)
ereof
charset 'field'
type '1'
transmit using enter

/*---------------------------------------------------- 17221004 00017 */
WTM_panel_ID = 'PNL00009'
log 'WTM_panel_ID' WTM_panel_ID
transmit using pf3

/*---------------------------------------------------- 17235925 00019 */
WTM_panel_ID = 'PNL00010'
log 'WTM_panel_ID' WTM_panel_ID
cursor(17,57)
ereof
charset 'field'
type '2'
transmit using enter

/*---------------------------------------------------- 17511250 00021 */
WTM_panel_ID = 'PNL00011'
log 'WTM_panel_ID' WTM_panel_ID
cursor(13,44)
ereof
charset 'field'
type '15'
transmit using enter

/*---------------------------------------------------- 17522571 00023 */
WTM_panel_ID = 'PNL00012'
log 'WTM_panel_ID' WTM_panel_ID
cursor(17,57)
ereof
charset 'field'
type '1'
transmit using enter

/*---------------------------------------------------- 17533785 00025 */
WTM_panel_ID = 'PNL00013'
log 'WTM_panel_ID' WTM_panel_ID
transmit using pf12

/*---------------------------------------------------- 17545699 00027 */
WTM_panel_ID = 'PNL00014'
log 'WTM_panel_ID' WTM_panel_ID
transmit using clear

/*---------------------------------------------------- 17562011 00029 */
```

```
WTM_panel_ID = 'PNL00015'
log 'WTM_panel_ID' WTM_panel_ID
cursor(1,1)
charset 'field'
type 'cesf'
transmit using enter

/*---------------------------------------------------- 17580128 00031 */
WTM_panel_ID = 'PNL00016'
log 'WTM_panel_ID' WTM_panel_ID
transmit using clear

/*---------------------------------------------------- 18065529 00080 */
WTM_panel_ID = 'PNL00017'
log 'WTM_panel_ID' WTM_panel_ID
onin0002: onin substr(rh,1,5) = '6B80003201'x,
               then found = on
found = off
termself()
do while found = off             /* wait for onin0002 data received */
 wait until onin
end
deact onin0002

/* 18:06:55.29 ITP1508I SESSION ENDED WITH APPLICATION CICSC22F */

say 'WTM:' msgtxtid() 'Finished'
endtxt
```

28.5 Scheduling and running a simulation

In this part we walk through a process to schedule and run simulations using WTM.

1. Select option **4** (Run) on the WTM main panel. The WTM Process WSim Schedules panel is displayed showing all test schedules defined for the current project. A schedule is a WSim network definition with supporting documentation and reporting facilities. There are no existing schedules in this case, so use the **add** command to create a new schedule, as shown in Figure 28-35.

```
                              Process WSim Schedules
Command==> add_
Enter a line command or add to create a new Schedule.        Press PF3 to end.
Line commands: D Delete, N Notes, O Output, P Preprocess, R Resources,
               S Testcases, U UTBLs, X Execute, W NTWRK.

  Name     Type Notes Description                UTBLs   Last Run
******************************** Bottom of data ********************************



PF 1=Help      2=Split     3=End       4=          5=Add       6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-35 Adding new schedule

2. Provide a name, description, and type for this new schedule on the next panel, as shown in Figure 28-36.

```
                              Process WSim Schedules
                           Add New Test Schedule

 Type Schedule Name, Description and Schedule Type.  Then press Enter.

 Schedule Name: tsttrad1
 Description  : test tsttrad testcase
 Schedule Type: v _   (V=VTAMAPPL, T=TCP/IP, C=CPI-C)



 PF 1=Help      2=Split     3=End       4=          5=          6=
 PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-36 New schedule name, description, and type

3. WTM displays the “Milestones” panel with the steps to be performed as shown in Figure 28-37.

```
                                Process WSim Schedules
Comman ┌──────────────────────────── "Milestones" ────────────────────────────┐
Enter  │                                                                      │ to end.
Line c │ These are the steps to create a new WSim Schedule:                   │
       │                                                                      │
       │ Step 1 - Define the simulated resources and scripts.                 │
  Name │ Step 2 - Generate a WSim network (NTWRK).                            │
****** │ Step 3 - Define completion report thresholds.                        │ ********
       │                                                                      │
       │                                                                      │
       │                                                                      │
       └──────────────────────────────────────────────────────────────────────┘
```

Figure 28-37 “Milestones” panel with steps to create schedule

4. Selection of the network resources is done on the next panel shown in Figure 28-38.

```
                        Resource List for Schedule TSTTRAD1          Row 1 of 1
Command==>
Enter command sort to sort by VTAMname.                       Press PF3 to end.
Line commands: S Select, I Insert, R Repeat, D Delete.

  VTAMname WSIMname  Test...  Name       Description
s VAPPL00  WSIMLU00
****************************** Bottom of data ********************************

PF 1=Help      2=Split     3=End       4=          5=Sort      6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-38 Selecting resource for simulation

5. The list of the test cases (also test groups and test cycles, if applicable) available to be used in this schedule is presented; you can define the order in which they will be executed. Each test case can be used more than once. In this case multiple order positions should be specified, for example, 1 2 3, as shown in Figure 28-39.

```
Specify Order for Resource WSIMLU00 under VAPPL00
                                                                Row 1 of 1
Command==>
Enter order or summary.          Press PF3 to end.

Order                  Name       Test...  Type  Description
1_                     TSTTRAD    Case       V   Logon,trader,logon,getquot
***************************** Bottom of data ******************************

PF 1=Help      2=Split     3=End       4=          5=          6=Summary
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-39 Specifying order for resource

6. After performing the remaining steps that were identified on the “milestones” panel, WTM finishes working with the network resources as shown in Figure 28-40.

```
                    Resource List for Schedule TSTTRAD1          Row 1 of 1
Command==>
Enter command sort to sort by VTAMname.                   Press PF3 to end.
Line commands: S Select, I Insert, R Repeat, D Delete.

  VTAMname WSIMname  Test...  Name       Description
_ VAPPL00  WSIMLU00  CASE     TSTTRAD    LOGON,TRADER,LOGON,GETQUOTE,BUYSTOCK,G
******************************** Bottom of data ********************************

PF 1=Help      2=Split     3=End       4=          5=Sort      6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-40 Test case assigned to resource

7. Press PF3. WTM will present (the first time only) the Completion Report Response Time Thresholds panel where you can enter response time mean, medium, mode, and high values in seconds and percentiles of 95%, 90%, 80% and 70% in seconds for success criteria to be used in the Response Time Report.

   These values can be set up later using option **4** RUN from the main panel, line command **o** for the corresponding schedule, PF8 to get to the lower part of the next panel, and option **18** THRESH.

8. Now you are ready to execute the newly created schedule TSTTRAD1. The line command **x** is used to start a simulation as shown in Figure 28-41.

```
                          Process WSim Schedules                    Row 1 of 1
Command==>
Enter a line command or add to create a new Schedule.        Press PF3 to end.
Line commands: D Delete, N Notes, O Output, P Preprocess, R Resources,
               S Testcases, U UTBLs, X Execute, W NTWRK.

  Name     Type Notes Description                 UTBLs   Last Run
x ISTTRAD1   V        test tsttrad testcase
****************************** Bottom of data ********************************

PF 1=Help      2=Split     3=End       4=          5=Add       6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-41 Starting TSTTRAD1

9. WTM provides an opportunity to specify the data set names of one or more WSim logs that will be written during the test and provides the generated ones which may be changed if desired, as shown in Figure 28-42.

```
                          Process WSim Schedules                    Row 1 of 1
                           Specify WSim Log Names

 Change the lognames and press Enter to continue or PF3 to end.

 ITPENTER: 'SLITVAK.WTMTEST.RESPROJ1.MLOG.TSTTRAD1'     Mandatory
 Baseline: ____________________________________________ Optional

 Completion Report (Y/N): y _

 PF 1=Help      2=Split     3=End       4=          5=          6=
 PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve

PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-42 Specifying log data sets names

10. When performing a test in batch mode, WSim directs generated console messages to the MVS log; when performing a test from TSO, WSim sends the console messages to the TSO terminal, as shown in Figure 28-43.

```
ITP016I Workload Simulator (WSim) Version 1, Release 1.0, May 23, 2002, 11:05

ITP003I WSim INITIALIZATION COMPLETE
ITP200I DISPLAY MONITOR FACILITY ACTIVE USING APPL WTMDM00
ITP029I INITIALIZATION COMPLETE FOR NETWORK TSTTRAD1
ITP006I NETWORK TSTTRAD1 STARTED
ITP137I TSTTRAD1 WSIMLU00-00001 - WTM: TSTTRAD Finished
ITP137I TSTTRAD1 WSIMLU00-00001 - WTMEND: 1 out of 1 finished.
ITP137I TSTTRAD1 WSIMLU00-00001 - WTMEND: All scripts run.  Issuing ZEND.
ITP002I TSTTRAD1 WSIMLU00-00001 - ZEND
ITP201I DISPLAY MONITOR FACILITY IS CLOSED DOWN
ITP079I WSim IS CLOSED DOWN
*** _
```

Figure 28-43 WSim simulation messages

11. The generated completion report from the simulation is presented in Figure 28-44.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       SLITVAK.WTMTEST.RESPROJ1.CREP(R5) - 01.00       Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> PAGE
****** ***************************** Top of Data ******************************
==MSG> -Warning- The UNDO command is not available until you change
==MSG>           your edit profile using the command RECOVERY ON.
==MSG> -CAUTION- Profile is set to STATS ON. Statistics did not exist for
==MSG>           this member, but will be generated if data is saved.
000001 Report for TSTTRAD1 - test tsttrad testcase.
000002 Run completed at 14:50:16 on 26 Oct 2004.
000003 No Response Time Thresholds selected.
000004 Loglist Extract:
000005 RUN  TIME 14.49.37,   OCTOBER 26, 2004  VERSION  1  RELEASE  1.0.0
000006 14501537 ITP137I TSTTRAD1 WSIMLU00-00001 - WTM: TSTTRAD Finished
000007 Comments:
000008 _
000009
****** **************************** Bottom of Data ****************************

 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 28-44 Completion report

The test is finished.

28.6 Simulation reports

Several output reports are accessible from the list of schedules. Enter the line command **o** to view outputs for the corresponding schedule, as shown in Figure 28-45.

```
                          Process WSim Schedules                    Row 1 of 1
Command==>
Enter a line command or add to create a new Schedule.        Press PF3 to end.
Line commands: D Delete, N Notes, O Output, P Preprocess, R Resources,
               S Testcases, U UTBLs, X Execute, W NTWRK.

  Name     Type Notes Description                 UTBLs   Last Run
o TSTTRAD1   V        test tsttrad testcase            10/26/04 14:49
******************************** Bottom of data ********************************

PF 1=Help      2=Split     3=End       4=          5=Add       6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-45 Output of TSTTRAD1 simulation to be viewed

The Report panel is presented as shown in Figure 28-46.

```
                          Reports for schedule TSTTRAD1

Select one of the following.  Then press Enter.

      Command   Action                                                More:  +
1  1. TLOG      View loglist report from the latest WSim run
   2. TRESP     View response time report from the latest WSim run
   3. TDM       Log display monitor for the latest WSim run
   4. TSP       View SYSPRINT from the latest WSim run

   5. MLOG      View baseline loglist report
   6. MRESP     View baseline response time report
   7. MDM       Log display monitor for the baseline log

   8. COMP      View screen compare report
   9. CDM       Log display comparator
  10. RTCOMP    Edit response time compare report

  11. COMPREP   Edit completion reports


Command===>
PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=          8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-46 WTM Reports panel, part 1

Press PF8 to see the second part of the panel, as shown in Figure 28-47.

```
                        Reports for schedule TSTTRAD1

Select one of the following.  Then press Enter.

       Command   Action                                                More:  -
__  12. LOGLIST   Edit Loglist report control cards
    13. RESPONSE  Edit Response report control cards
    14. COMPARE   Edit Compare report control cards

    15. NTWRK     Edit WSim network
    16. VTAMLST   Edit Application Major Node

    17. LLMASK    Edit completion report Loglist message masks

    18. THRESH    Set completion report response time thresholds




Command===> _____________________________________________________________
PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=          9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-47 WTM Reports panel, part 2

The choices on this panel have the following meanings:

TLOG WTM invokes ITPLL on the WSim/ISPF Interface for the WSim log from the latest simulation run. The loglist report is browsed. If the baseline log exists, this log is known as the test WSim log.

TRESP WTM invokes ITPRESP on the WSim/ISPF Interface for the WSim log from the latest simulation run. The response time report is browsed.

TDM The screen images for the WSim log from the latest simulation run are displayed.

TSP The SYSPRINT file from the latest simulation run is browsed.

MLOG WTM invokes ITPLL on the WSim/ ISPF Interface for the master WSim log. The loglist report is browsed.

MRESP WTM invokes ITPRESP on the WSim/ISPF Interface for the master WSim log. The response time report is browsed.

MDM The screen images for the master WSim log are displayed.

COMP If both the master and test WSim logs exist, WTM invokes ITPCOMP on the WSim/ISPF Interface. The output is browsed.

CDM If both the master and test WSim logs exist, WTM invokes the Log Display Monitor to compare screen images.

RTCOMP If both the master and test WSim logs exist, WTM creates a report that compares the response times on both the logs. You are placed in an edit session on this report.

COMPREP This displays a list of the completion reports that can be selected for edit.

LOGLIST Edit the control cards used when the WSim loglist utility ITPLL is run.

RESPONSE Edit the control cards used when the WSim response time utility ITPRESP is run.

COMPARE Edit the control cards used when the WSim screen compare utility ITPCOMP is run.

NTWRK Edit the WSim network definition for this schedule.

VTAMLST Edit the VTAM application major node generated by WTM for this schedule.

LLMASK When the completion report is created, certain messages are not shown on the report panel and some messages are not written to the completion report itself. These masks can be edited using this option.

THRESH Use this option to change the response time thresholds. The response time thresholds are useful in determining the rate of traffic – what messages were sent and received within a specified criteria. The percentile values are extracted from the response time skeleton. The top four values (out of 10 maximum) of the PERCENT command are used for the percentiles.

For our example, the only report of interest would be the loglist report. Select option **1** TLOG. Figure 28-48 appears, giving the ability to specify some control information for the report.

```
                    WSim: Control Analysis of Logged Data

Type information.  Then press Enter.
                                                                  More:  +
  Message decks/STL procedures _________ , _________ , _________ , _________
  Networks . . . . . . . . . . _________ , _________ , _________ , _________
  Group name and group type    _________ , U (V=VTAMAPPL, T=TCP/IP,
                                             A=APPCLU)
    Term, LU, or TP name . . . _______________ , _______________ , _______________
                               _______________ , _______________ , _______________
    Include terms, LUs, TPs    Y            (Y=Yes or N=No)
  Time Limits  . . . . . . . . ______ - ______

  Select one or more record types by typing a '/' or 'S'.
    / Console                / Informational          / Message generation trace
    / Display                / Data                   / STL trace
      _ Extended attributes  _ Header-only records    _ Verification logs
      _ Non-displayables     _ CPI-C trace data       / Log
      _ Separate partitions
      _ Scripting records


 F1=Help              F2=Split   F5=Refresh  F7=Bkwd  F8=Fwd  F9=Swap
F10=Additional input  F11=Save   F12=Cancel
```

Figure 28-48 Controls for loglist report, part 1

Press PF8 to see the lower part of this panel as shown in Figure 28-49.

```
                    WSim: Control Analysis of Logged Data

Type information.  Then press Enter.
                                                                    More: -
  Control Options
    SNA record formatting . . . Y  (Y=Yes or N=No)
    Uppercase translation . . . N  (Y=Yes or N=No)
    User exit member name . . . ________
    Header line . . . . . . . . ____________________________

  Data set to save controls . . ______________________________________________

 F1=Help               F2=Split   F5=Refresh  F7=Bkwd  F8=Fwd  F9=Swap
F10=Additional input  F11=Save   F12=Cancel
```

Figure 28-49 Controls for loglist report, part 2

Additional parameters can be added on the second panel, which is accessible by pressing PF10, as shown in Figure 28-50.

```
                    WSim: Specify Additional Loglist Groups

Type information.  Then press Enter.

  Group name and group type   ________ , U (V=VTAMAPPL, T=TCP/IP,
                                            A=APPCLU)
    Term, LU, or TP name. . . _______________ , _______________ , _______________
                              _______________ , _______________ , _______________
    Include terms, LUs, TPs   Y            (Y=Yes or N=No)
  Group name and group type   ________ , U (V=VTAMAPPL, T=TCP/IP,
                                            A=APPCLU)
    Term, LU, or TP name. . . _______________ , _______________ , _______________
                              _______________ , _______________ , _______________
    Include terms, LUs, TPs   Y            (Y=Yes or N=No)
  Group name and group type   ________ , U (V=VTAMAPPL, T=TCP/IP,
                                            A=APPCLU)
    Term, LU, or TP name. . . _______________ , _______________ , _______________
                              _______________ , _______________ , _______________
    Include terms, LUs, TPs   Y            (Y=Yes or N=No)



F1=Help  F2=Split  F5=Refresh  F9=Swap  F11=Save  F12=Cancel
```

Figure 28-50 Additional parameters fro loglist report

When WTM finishes the analysis, the informational message appears, and you can browse the report shown in Figure 28-51.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------
VIEW       SLITVAK.WTM.SYSPRINT                          Columns 00001 00072
Command ===> ____________________________________________ Scroll ===> PAGE
****** ***************************** Top of Data ******************************
000001 1WSim LOGLIST OUTPUT
000002 0CNSL
000003  DSPLY
000004  LOG
000005  INFO
000006  DATA
000007  CDLOG
000008  NOHDR
000009  NORR
000010  MTRC
000011  STRC
000012  NOVRPAC
000013  FMTSNA
000014  NOCTRC
000015  RUN
000016 1WSi
000017  RUN  | Analysis of log data set completed successfully. | SE  1.0.0
 F1=Help                                                          F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 28-51 Loglist report, part 1

This report can be very long! In this case it contains almost 2200 lines, as shown in Figure 28-52.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------
VIEW       SLITVAK.WTM.SYSPRINT                          Columns 00001 00072
Command ===> ____________________________________________ Scroll ===> PAGE
002176   NAME            NAME            NAME            TIME     TIME     TI
002177 0                                                14501537 0104300  1100
002178 0  ITP137I TSTTRAD1 WSIMLU00-00001 - WTM: TSTTRAD Finished
002179 0-----------------------------------------------------------------------
002180 0                                                14501537 0104300  1100
002181 0  ITP137I TSTTRAD1 WSIMLU00-00001 - WTMEND: 1 out of 1 finished.
002182 0-----------------------------------------------------------------------
002183 0                                                14501537 0104300  1100
002184 0  ITP137I TSTTRAD1 WSIMLU00-00001 - WTMEND: All scripts run.  Issuing Z
002185 0-----------------------------------------------------------------------
002186 0                                                14501537 0104300  1100
002187 0  ITP002I TSTTRAD1 WSIMLU00-00001 - ZEND
002188 0-----------------------------------------------------------------------
002189 0                                                14501537 0104300  1100
002190 0  ITP201I DISPLAY MONITOR FACILITY IS CLOSED DOWN
002191 1WSim LOGLIST OUTPUT
002192 0END
****** ***************************** Bottom of Data ***************************
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 28-52 Loglist report, part 2

28.7 Organizing test cases

There are several options available in WTM to organize test cases and make their reuse possible.

A *test group* is an ordered list of test items (test cases and MSGTXTs). The purpose of using test groups is to encourage modularity of test cases, such as a special logon test case,

developed once and reused in multiple test groups. As a result, if the logon process changes, only the logon test case needs to be changed, even it is used in several different test groups.

A *test cycle* is also an ordered list of test items (test cases, MSGTXTs, and test groups). The test cycles represent another level of organization of test items and allow the creation of highly complex structures. The WTM test cycle management panels are very similar to the ones for the test groups.

There are several methods to create modular scripts:

- Capture a single test case starting from the beginning of an application session.
- Capture a single test case by starting and stopping capture interactively.
- Capture multiple test cases from a single application session by switching to different IDC log files.

28.7.1 Creating modular scripts

Use the following steps to start an IDC session and capture test cases.

1. Select option **1** CASE or enter the command **case** from the WTM main panel. Use the **add** command to start with a new test case, as shown in Figure 28-53.

```
                                Process Testcases                       Row 1 of 1
Command==> add_
Enter a line command or add to create a new Testcase.          Press PF3 to end.
Line commands: D Delete, N Notes, O Open, T Translate, U UTBLs, V View.

  Name     Type UTBLs Notes Description
_ TSTTRAD   V             Logon,trader,logon,getquote,buystock,getqu
******************************** Bottom of data ********************************



PF 1=Help       2=Split      3=End        4=           5=Add        6=
PF 7=Up         8=Down       9=Swap      10=Left      11=Right     12=Retrieve
```

Figure 28-53 Adding another test case

2. On the next panel, shown in Figure 28-53, provide the name and description for the new test case and select option **1** to create the test case using the IDC utility.

```
                              Process Testcases                    Row 1 of 1
                                Add New Testcase

 Type Testcase Name, Description and Source.  Then press Enter.

 Testcase Name: logon
 Description  : logon to cicsc22f
 Source       : 1  1. Add a 3270 testcase using IDC
                   2. Add a 3270 testcase using an SNA trace
                   3. Add a 3270 testcase using a WSim or IDC log
                   4. Add a testcase using the WSim STL models
                   5. Add a testcase using an STL skeleton
                   6. Add a CPI-C testcase using an LU 6.2 SNA trace

 PF 1=Help      2=Split     3=End       4=          5=          6=
 PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve

 PF 1=Help      2=Split     3=End       4=          5=          6=
 PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-54 Adding test case logon

3. Press Enter. When the IDC utility panel appears, select option **1** to start a session with a host application. In our case this was the CICS system, as shown in Figure 28-55.

```
IDCMAIN          WSim Interactive Data Capture (IDC) Utility

Select one of the following, then press Enter.

1 1. Start a session with a host application and capture data

  2. Generate an STL program from captured data
  3. Generate a message generation deck from captured data

  4. End the IDC utility program

WSim Version 1 Release 1.0.0  Program Number 5655-I39

Licensed Materials - Property of IBM
5655-I39 (C) Copyright IBM Corporation 1976, 2002.  All Rights Reserved.
US Government Users Restricted Rights - Use, duplication or disclosure
restricted by GSA ADP Schedule Contract with IBM Corporation.
F1=Help  F3=Exit  F12=Cancel
```

Figure 28-55 WSim IDC panel, start session

4. Press Enter. On the next panel supply the host application name (the corresponding VTAM APPLID), the IDC utility log data set name, the option to replace or append the log data set, the IDC escape key, and if data capture should start immediately, as shown in Figure 28-56.

```
IDCSSP          WSim IDC:  Start Session with Host Application

Type information, then press Enter.

Session Data
  Host application name . . . . . . . cicsc22f
  Logon mode name . . . . . . . . . . LSX32702 (Optional)
  Logon user data . . . . . . . . . . ________________________ (Optional)


IDC log data set name . . . . . . . . SLITVAK.WTMUSER.IDCLOG(LOGON)
  If data set already exists, specify R        (R=Replace or A=Append)

Start capturing data immediately? . . Y        (Y=Yes or N=No)

IDC Escape key  . . . . . . . . . . . PA1__    (PAn, PFnn, CLEAR, or ATTN)





F1=Help  F3=Exit  F5=Refresh  F11=Save  F12=Cancel
```

Figure 28-56 IDC controls

5. Press Enter. WTM connects to the host application, in our case CICSC22F, as shown in Figure 28-57.

```
                          Signon to CICS                        APPLID CICSC22F

IBM'S  INTERNAL  SYSTEMS  MUST  ONLY  BE  USED  FOR  CONDUCTING
IBM'S BUSINESS OR FOR PURPOSES AUTHORIZED BY IBM MANAGEMENT



Type your userid and password, then press ENTER:

        Userid . . . . slitvak    Groupid . . . ________
        Password . . .
        Language . . . ___

     New Password . . .





DFHCE3520 Please type your userid.
F3=Exit
```

Figure 28-57 CICS signon panel

6. Enter the userid and password and press Enter. Logon to CICS is completed, as shown in Figure 28-58.

```
DFHCE3549 Sign-on is complete (Language ENU).
```

Figure 28-58 CICS signon complete

7. Since the test case under creation is just a logon to CICS, press the IDC escape key (PA1), and WTM brings up the Escape Actions panel, where you select option **2** to stop the capture as shown in Figure 28-59.

```
IDCESCA              WSim IDC:  Escape Actions

Select one of the following, then press Enter.
Note:  Options 4-9 do not change the current data capture status.

2 1. Start capturing data
  2. Stop capturing data
  3. End the session with the host application

  4. Add STL statements directly to the IDC log
  5. Add WSim scripting language statements directly to the IDC log

  6. Change IDC log data sets
  7. Reset logging to the beginning of the data set or appended data

  8. Pass the escape key to the host application
  9. Change the IDC escape key

Data capture status . . : ON
Current IDC log data set: SLITVAK.WTMUSER.IDCLOG(LOGON)
Current escape key  . . : PA1


F1=Help  F3=Exit  F12=Cancel
```

Figure 28-59 Stopping data capture

8. Press Enter and make sure that the data capture status indicator is OFF before proceeding further.

> **Important:** If option **3** to end the session with the host application is selected and Enter is pressed when data capture is ON, this event becomes a part of the captured data.

9. Press PF3 to return to the host session and issue the transaction **cesf logoff**. The IDC utility ends the session with CICSC22F and displays the confirmation message, as shown in Figure 28-60. The IDC log data set for the test case LOGON is created.

```
IDCSSP          WSim IDC:  Start Session with Host Application

Type information, then press Enter.

Session Data
  Host application name . . . . . . . CICSC22F
  Logon mode name . . . . . . . . . . LSX32702 (Optional)
  Logon user data . . . . . . . . . . ________________________ (Optional)


IDC log data set name . . . . . . . . SLITVAK.WTMUSER.IDCLOG(LOGON)
  If data set already exists, specify R        (R=Replace or A=Append)

Start capturing data immediately? . . Y        (Y=Yes or N=No)

IDC Escape key  . . . . . . . . . . . PA1__    (PAn, PFnn, CLEAR, or ATTN)





ITP1508I SESSION ENDED WITH APPLICATION CICSC22F
F1=Help  F3=Exit  F5=Refresh  F11=Save  F12=Cancel
```

Figure 28-60 Session ended message

10. Press PF3. WTM performs the steps to create the test case from the IDC log and returns to the list of test cases with a confirmation message as shown in Figure 28-61.

```
                              Process Testcases                      Row 1 of 2
Command==> add_
Enter a line command or add to create a new Testcase.         Press PF3 to end.
Line commands: D Delete, N Notes, O Open, T Translate, U UTBLs, V View.

  Name    Type UTBLs Notes Description
_ LOGON     V              logon to cicsc22f                          Added
_ TSTTRAD   V              Logon,trader,logon,getquote,buystock,getqu
****************************** Bottom of data ********************************









PF 1=Help      2=Split     3=End       4=          5=Add       6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-61 Test case LOGON added

11. Enter the command **add** and press Enter to continue with creation of the next test case. This test case will involve the start of the **test** transaction and the choice of option **7** to start the trader application as shown in Figure 28-62.

```
                              Process Testcases                      Row 1 of 2
                                Add New Testcase

 Type Testcase Name, Description and Source.  Then press Enter.

 Testcase Name: initst7
 Description  : start test trans, opt 7
 Source       : 1 _1. Add a 3270 testcase using IDC
                   2. Add a 3270 testcase using an SNA trace
                   3. Add a 3270 testcase using a WSim or IDC log
                   4. Add a testcase using the WSim STL models
                   5. Add a testcase using an STL skeleton
                   6. Add a CPI-C testcase using an LU 6.2 SNA trace

 PF 1=Help      2=Split     3=End       4=          5=          6=
 PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve

PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-62 Adding test case INITST7

12. The panel to enter data required to start a host application is displayed. Since the LOGON test case is already created, we do not want to start capturing data immediately as shown in Figure 28-63.

```
IDCSSP          WSim IDC:  Start Session with Host Application

Type information, then press Enter.

Session Data
  Host application name . . . . . . . cicsc22f
  Logon mode name . . . . . . . . . . LSX32702 (Optional)
  Logon user data . . . . . . . . . . ________________________ (Optional)


IDC log data set name . . . . . . . . SLITVAK.WTMUSER.IDCLOG(INITST7)
  If data set already exists, specify R        (R=Replace or A=Append)

Start capturing data immediately? . . n        (Y=Yes or N=No)

IDC Escape key  . . . . . . . . . . . PA1__    (PAn, PFnn, CLEAR, or ATTN)





F1=Help  F3=Exit  F5=Refresh  F11=Save  F12=Cancel
```

Figure 28-63 Data capture delayed

13. After you complete the steps to log on to application CICSC22F, at the screen shown in Figure 28-58 on page 712, press the IDC escape key (PA1). This will bring up the already familiar IDC utility Escape Actions panel. Note, that the data capture status is OFF, as shown in Figure 28-64.

```
IDCESCA              WSim IDC:  Escape Actions

Select one of the following, then press Enter.
Note:  Options 4-9 do not change the current data capture status.

1 1. Start capturing data
  2. Stop capturing data
  3. End the session with the host application

  4. Add STL statements directly to the IDC log
  5. Add WSim scripting language statements directly to the IDC log

  6. Change IDC log data sets
  7. Reset logging to the beginning of the data set or appended data

  8. Pass the escape key to the host application
  9. Change the IDC escape key

Data capture status . . : OFF
Current IDC log data set: SLITVAK.WTMUSER.IDCLOG(INITST7)
Current escape key  . . : PA1


F1=Help  F3=Exit  F12=Cancel
```

Figure 28-64 Starting delayed data capture

14. Select option **1** and press Enter. The data capture status is now ON, as shown in Figure 28-65.

```
IDCESCA              WSim IDC:  Escape Actions

Select one of the following, then press Enter.
Note:  Options 4-9 do not change the current data capture status.

_ 1. Start capturing data
  2. Stop capturing data
  3. End the session with the host application

  4. Add STL statements directly to the IDC log
  5. Add WSim scripting language statements directly to the IDC log

  6. Change IDC log data sets
  7. Reset logging to the beginning of the data set or appended data

  8. Pass the escape key to the host application
  9. Change the IDC escape key

Data capture status . . : ON
Current IDC log data set: SLITVAK.WTMUSER.IDCLOG(INITST7)
Current escape key  . . : PA1


F1=Help  F3=Exit  F12=Cancel
```

Figure 28-65 Data capture is now ON

15. Press PF3 and WTM will bring you back to the application screen. Start the transaction `test` as shown Figure 28-66.

```
test

DFHCE3549 Sign-on is complete (Language ENU).
```

Figure 28-66 Transaction test entered

16. On the screen brought up by the **test** transaction select option **7** and press Enter to start the trader application, as shown in Figure 28-67.

```
                    Sample Program - CPU / VSAM / DB2
                          for Application Monitor


                  Test Selection:

                      1. CPU  Loop
                      2. VSAM Loop
                      3. DB2  Loop
                      4. IMS  Loop

                      5. Cobol Multi-Csects program
                      6. Cobol and PL/I Multi-Csects program

                      7. Named Set and Transaction Routing
                         start CICS/DB2 remote application
                      8. Named Set and Dynamic Program Link
                      9. Named Set and Function Shipping

                 Select a test (value between 1 and 9):  7


 ------------------------------------------------------------------------------
 PF3=Return                                                          PF12=Exit
```

Figure 28-67 Option 7 entered

17. The trader application logon screen, shown in Figure 28-17 on page 686 and Figure 28-75 on page 720, will appear. Do not enter any values in any fields and press the IDC escape key (PA1) to get to the IDC utility Escape Actions panel. Select option **2** to stop capturing data and press Enter as shown Figure 28-68.

```
IDCESCA              WSim IDC:  Escape Actions

Select one of the following, then press Enter.
Note:  Options 4-9 do not change the current data capture status.

2 1. Start capturing data
  2. Stop capturing data
  3. End the session with the host application

  4. Add STL statements directly to the IDC log
  5. Add WSim scripting language statements directly to the IDC log

  6. Change IDC log data sets
  7. Reset logging to the beginning of the data set or appended data

  8. Pass the escape key to the host application
  9. Change the IDC escape key

Data capture status . . : ON
Current IDC log data set: SLITVAK.WTMUSER.IDCLOG(INITST7)
Current escape key  . . : PA1


F1=Help  F3=Exit  F12=Cancel
```

Figure 28-68 Stopping data capture

18. The data capture status indicator changes to OFF. Select option **3** to end the session and press Enter as shown in Figure 28-69.

```
IDCESCA              WSim IDC:  Escape Actions

Select one of the following, then press Enter.
Note:  Options 4-9 do not change the current data capture status.

3 1. Start capturing data
  2. Stop capturing data
  3. End the session with the host application

  4. Add STL statements directly to the IDC log
  5. Add WSim scripting language statements directly to the IDC log

  6. Change IDC log data sets
  7. Reset logging to the beginning of the data set or appended data

  8. Pass the escape key to the host application
  9. Change the IDC escape key

Data capture status . . : OFF
Current IDC log data set: SLITVAK.WTMUSER.IDCLOG(INITST7)
Current escape key  . . : PA1


F1=Help  F3=Exit  F12=Cancel
```

Figure 28-69 Data capture is OFF; Session end requested

19. WTM returns to the IDC utility panel and displays the session end confirmation message, as shown in Figure 28-70.

```
IDCSSP          WSim IDC:  Start Session with Host Application

Type information, then press Enter.

Session Data
  Host application name . . . . . . . CICSC22F
  Logon mode name . . . . . . . . . . LSX32702 (Optional)
  Logon user data . . . . . . . . . . ________________________ (Optional)


IDC log data set name . . . . . . . . SLITVAK.WTMUSER.IDCLOG(INITST7)
  If data set already exists, specify R        (R=Replace or A=Append)

Start capturing data immediately? . . N        (Y=Yes or N=No)

IDC Escape key  . . . . . . . . . . . PA1    (PAn, PFnn, CLEAR, or ATTN)





ITP1508I SESSION ENDED WITH APPLICATION CICSC22F
F1=Help  F3=Exit  F5=Refresh  F11=Save  F12=Cancel
```

Figure 28-70 Session ended message

20. Press PF3. WTM will go through the test case creation steps, as shown in Figure 28-71.

```
          WSim: Interactively Capture and Build Message Decks and STL Programs
                          "Milestones"
Type i
         These are the steps to create a testcase using IDC:          More:  +
  Inpu
    De   Step 1 - ITPIDC   - Invoke IDC.
         Step 2 - ITPLSGEN - Create STL program from the IDC Log.
  Outp   Step 3 - ITMDCAS  - Add WSim/TM upgrades to STL program.
    Tr   Step 4 - ITPSTL   - Translate STL program into a MSGTXT.
    Pr

  Control Options
    Run mode  . . . . . . . . I  (I=Interactive, B=Batch, N=Batch-no panel)
      System console output   Y  (Y=Yes or N=No)
      Suppress user messages  Y  (Y=Yes or N=No)
    IDC VTAM application name ITPIDC0
    Query 3270 based on BIND  N  (Y=Yes or N=No)
    Build debug comments  . . N  (Y=Yes or N=No)



Command ===>
F1=Help   F2=Split   F3=Exit    F5=Refresh  F6=Browse prt  F7=Bkwd  F8=Fwd
F9=Swap  F11=Save   F12=Cancel
```

Figure 28-71 “Milestones” panel with test case creation steps

21. The test case INITST7 is created and the corresponding message is displayed. Enter the command **add** again to continue to create new test cases as shown in Figure 28-72. Press Enter.

```
                              Process Testcases                      Row 1 of 3
Command==> add
Enter a line command or add to create a new Testcase.         Press PF3 to end.
Line commands: D Delete, N Notes, O Open, T Translate, U UTBLs, V View.

  Name    Type UTBLs Notes Description
_ INITST7   V              start test trans, opt 7                    Added
_ LOGON     V              logon to cicsc22f                          Added
_ TSTTRAD   V              Logon,trader,logon,getquote,buystock,getqu
****************************** Bottom of data ******************************

PF 1=Help       2=Split     3=End       4=          5=Add       6=
PF 7=Up         8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-72 Test case INITST7 added; Continue with next test case

22. The next test case involves logging on to the trader application and is called TRADLOG, as shown in Figure 28-73.

```
                              Process Testcases                      Row 1 of 3
                                 Add New Testcase

 Type Testcase Name, Description and Source.  Then press Enter.

 Testcase Name: tradlog
 Description  : logon to trader appl
 Source       : 1  1. Add a 3270 testcase using IDC
                   2. Add a 3270 testcase using an SNA trace
                   3. Add a 3270 testcase using a WSim or IDC log
                   4. Add a testcase using the WSim STL models
                   5. Add a testcase using an STL skeleton
                   6. Add a CPI-C testcase using an LU 6.2 SNA trace

 PF 1=Help       2=Split     3=End       4=          5=          6=
 PF 7=Up         8=Down      9=Swap     10=Left     11=Right    12=Retrieve

PF 1=Help       2=Split     3=End       4=          5=          6=
PF 7=Up         8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-73 Adding test case TRADLOG

23. Start a session with the CICSC22F again and do not start data capture immediately as shown in Figure 28-74.

```
IDCSSP          WSim IDC:  Start Session with Host Application

Type information, then press Enter.

Session Data
  Host application name . . . . . . . cicsc22f
  Logon mode name . . . . . . . . . . LSX32702 (Optional)
  Logon user data . . . . . . . . . . ________________________ (Optional)


IDC log data set name . . . . . . . . SLITVAK.WTMUSER.IDCLOG(TRADLOG)
  If data set already exists, specify R        (R=Replace or A=Append)

Start capturing data immediately? . . n        (Y=Yes or N=No)

IDC Escape key  . . . . . . . . . . . PA1    (PAn, PFnn, CLEAR, or ATTN)





F1=Help  F3=Exit  F5=Refresh  F11=Save  F12=Cancel
```

Figure 28-74 Data capture delayed for test case TRADLOG

24. Repeat all the steps to log on to the CICSC22F, start the test transaction, and select option **7** to start the trader application (we do not show these screens to save space). At the data trader application logon screen, shown in Figure 28-75, press the IDC escape key (PA1).

```
                    Share Trading Demonstration                TRADER.T001

                     Share Trading Manager: Logon




                Enter your User Name:


                Enter your Password:




--------------------------------------------------------------------------
PF3=Exit                                                          PF12=Exit
```

Figure 28-75 Trader application logon screen

25. Select option **1** on the IDC utility Escape Actions panel to start data capturing, as shown in Figure 28-76. Press Enter and PF3 to return to the CICS session.

```
IDCESCA              WSim IDC:  Escape Actions

Select one of the following, then press Enter.
Note:  Options 4-9 do not change the current data capture status.

1 1. Start capturing data
  2. Stop capturing data
  3. End the session with the host application

  4. Add STL statements directly to the IDC log
  5. Add WSim scripting language statements directly to the IDC log

  6. Change IDC log data sets
  7. Reset logging to the beginning of the data set or appended data

  8. Pass the escape key to the host application
  9. Change the IDC escape key

Data capture status . . : OFF
Current IDC log data set: SLITVAK.WTMUSER.IDCLOG(TRADLOG)
Current escape key  . . : PA1


F1=Help  F3=Exit  F12=Cancel
```

Figure 28-76 Starting data capture for test case TRADLOG

26. Continue with the trader application workflow. Press the IDC escape key (PA1) when at the Company Selection screen, shown in Figure 28-77.

```
                    Share Trading Demonstration                TRADER.T002

              Share Trading Manager: Company Selection


                    1. Casey_Import_Export

                    2. Casey_Import_Export

                    3. Casey_Import_Export

                    4. Casey_Import_Export



             Please select a company (1,2,3 or 4) :




 ------------------------------------------------------------------------------
 PF3=Return                                                          PF12=Exit
```

Figure 28-77 Trader application company selection screen

27. On the familiar IDC Escape Actions panel, shown in Figure 28-78, select option **6** and press Enter.

```
IDCESCA              WSim IDC:  Escape Actions

Select one of the following, then press Enter.
Note:  Options 4-9 do not change the current data capture status.

6 1. Start capturing data
  2. Stop capturing data
  3. End the session with the host application

  4. Add STL statements directly to the IDC log
  5. Add WSim scripting language statements directly to the IDC log

  6. Change IDC log data sets
  7. Reset logging to the beginning of the data set or appended data

  8. Pass the escape key to the host application
  9. Change the IDC escape key

Data capture status . . : ON
Current IDC log data set: SLITVAK.WTMUSER.IDCLOG(TRADLOG)
Current escape key  . . : PA1


F1=Help  F3=Exit  F12=Cancel
```

Figure 28-78 IDC Escape actions panel

28. The IDC utility Change IDC Log Data Sets panel appears. Overtype the log data set name in the new IDC log data set name field with a new name, for example selcomp (select company) as shown in Figure 28-79, and press Enter.

```
IDCESC6              WSim IDC:  Change IDC Log Data Sets

Type information, then press Enter.

New IDC log data set name . . . . . .  SLITVAK.WTMUSER.IDCLOG(selcomp)_____
  If data set already exists, specify  R   (R=Replace or A=Append)









Data capture status . . . . . . . . :  ON
Current IDC log data set  . . . . . :  SLITVAK.WTMUSER.IDCLOG(TRADLOG)



F1=Help  F3=Exit  F12=Cancel
```

Figure 28-79 New log data set will be used

29. IDC displays the message confirming that the previous IDC log file is closed and the new one is opened as shown in Figure 28-80.

```
IDCESC6              WSim IDC:  Change IDC Log Data Sets

Type information, then press Enter.

New IDC log data set name . . . . . .  SLITVAK.WTMUSER.IDCLOG(SELCOMP)
  If data set already exists, specify  R   (R=Replace or A=Append)



Data capture status . . . . . . . . :  ON
Current IDC log data set  . . . . . :  SLITVAK.WTMUSER.IDCLOG(SELCOMP)



Current IDC data set closed, new one opened
F1=Help  F3=Exit  F12=Cancel
```

Figure 28-80 IDC log data sets switched

30. Press PF3 to return to the host application session. Continuing to work with the trader application, select a company and press Enter. When presented with the trader application screen for options selection, shown in Figure 28-81, press the IDC escape key (PA1).

```
                    Share Trading Demonstration                TRADER.T003

                   Share Trading Manager: Options

                    1. New Real-Time Quote

                    2. Buy Shares

                    3. Sell Shares



                Please select an option (1,2 or 3):  _



--------------------------------------------------------------------------
PF3=Return                                                       PF12=Exit
```

Figure 28-81 Trader application option selection screen

31. On the IDC utility Escape Actions panel select option **6**, as shown in Figure 28-82, to switch to the next IDC log file. Press Enter.

```
IDCESCA              WSim IDC:  Escape Actions

Select one of the following, then press Enter.
Note:  Options 4-9 do not change the current data capture status.

6 1. Start capturing data
  2. Stop capturing data
  3. End the session with the host application

  4. Add STL statements directly to the IDC log
  5. Add WSim scripting language statements directly to the IDC log

  6. Change IDC log data sets
  7. Reset logging to the beginning of the data set or appended data

  8. Pass the escape key to the host application
  9. Change the IDC escape key

Data capture status . . : ON
Current IDC log data set: SLITVAK.WTMUSER.IDCLOG(SELCOMP)
Current escape key  . . : PA1


F1=Help  F3=Exit  F12=Cancel
```

Figure 28-82 Changing log data sets

32. Overtype the current IDC log data set name with the new one and press Enter as shown in Figure 28-83.

```
IDCESC6              WSim IDC:  Change IDC Log Data Sets

Type information, then press Enter.

New IDC log data set name . . . . . .  SLITVAK.WTMUSER.IDCLOG(buy1000)_____
  If data set already exists, specify  R   (R=Replace or A=Append)









Data capture status . . . . . . . . :  ON
Current IDC log data set  . . . . . :  SLITVAK.WTMUSER.IDCLOG(SELCOMP)




F1=Help  F3=Exit  F12=Cancel
```

Figure 28-83 New log data set name entered

33. The old IDC log file is closed and the new one is opened, as shown in Figure 28-84.

```
IDCESC6              WSim IDC:  Change IDC Log Data Sets

Type information, then press Enter.

New IDC log data set name . . . . . .  SLITVAK.WTMUSER.IDCLOG(BUY1000)
  If data set already exists, specify  R   (R=Replace or A=Append)



Data capture status . . . . . . . . :  ON
Current IDC log data set  . . . . . :  SLITVAK.WTMUSER.IDCLOG(BUY1000)



Current IDC data set closed, new one opened
F1=Help  F3=Exit  F12=Cancel
```

Figure 28-84 Log data sets switched

34. Press PF3 to continue working with the trader application. Buy 1000 shares as shown in Figure 28-85.

```
                    Share Trading Demonstration                TRADER.T005

                 Share Trading Manager: Shares - Buy


            User Name:     ZZ

            Company Name:  Casey_Import_Export


            Number of Shares to Buy:  1000 ___________________________________
PF3=Return                                                          PF12=Exit
```

Figure 28-85 Trader application: Buying 1000 shares

35. When done, press the IDC escape key PA1, select option **6** on the IDC utility Escape Actions panel to switch to the new IDC log file, press Enter, and overtype the name of the current IDC log data set with the new one and press Enter, as shown in Figure 28-86.

```
IDCESC6              WSim IDC:  Change IDC Log Data Sets

Type information, then press Enter.

New IDC log data set name . . . . . .  SLITVAK.WTMUSER.IDCLOG(sell900)
  If data set already exists, specify  R   (R=Replace or A=Append)









Data capture status . . . . . . . . :  ON
Current IDC log data set  . . . . . :  SLITVAK.WTMUSER.IDCLOG(BUY1000)




F1=Help  F3=Exit  F12=Cancel
```

Figure 28-86 New data set log name SELL900 entered

36. The IDC utility closed the previous IDC log file and opened the new one, as shown in Figure 28-87.

```
IDCESC6              WSim IDC:  Change IDC Log Data Sets

Type information, then press Enter.

New IDC log data set name . . . . . .  SLITVAK.WTMUSER.IDCLOG(SELL900)
  If data set already exists, specify  R   (R=Replace or A=Append)









Data capture status . . . . . . . . :  ON
Current IDC log data set  . . . . . :  SLITVAK.WTMUSER.IDCLOG(SELL900)



Current IDC data set closed, new one opened
F1=Help  F3=Exit  F12=Cancel
```

Figure 28-87 Old log data set closed, new opened

37. Press PF3 to return to the trader application and select option **3** to sell shares, as shown in Figure 28-88. Press Enter.

```
                    Share Trading Demonstration                    TRADER.T003

                    Share Trading Manager: Options

                     1. New Real-Time Quote

                     2. Buy Shares

                     3. Sell Shares




                Please select an option (1,2 or 3):  3



Request Completed OK
--------------------------------------------------------------------------------
PF3=Return                                                            PF12=Exit
```

Figure 28-88 Trader application: Sell shares selected

38. The number of shares to be sold is 900, as shown in Figure 28-89.

```
                    Share Trading Demonstration                    TRADER.T005

                 Share Trading Manager: Shares - Sell


            User Name:     ZZ

            Company Name:  Casey_Import_Export


            Number of Shares to Sell: 900




--------------------------------------------------------------------------------
PF3=Return                                                            PF12=Exit
```

Figure 28-89 Trader application: Selling 900 shares

39. The request to sell is completed as shown in Figure 28-90.

```
                         Share Trading Demonstration                TRADER.T003

                         Share Trading Manager: Options


                          1. New Real-Time Quote

                          2. Buy Shares

                          3. Sell Shares




                   Please select an option (1,2 or 3):



Request Completed OK
--------------------------------------------------------------------------------
PF3=Return                                                              PF12=Exit
```

Figure 28-90 Trader application: Shares sold

40. Press the IDC escape key (PA1), select option **6** on the IDC utility Escape Actions panel as shown in Figure 28-91, and press Enter.

```
IDCESCA                 WSim IDC:  Escape Actions

Select one of the following, then press Enter.
Note:  Options 4-9 do not change the current data capture status.

6 1. Start capturing data
  2. Stop capturing data
  3. End the session with the host application

  4. Add STL statements directly to the IDC log
  5. Add WSim scripting language statements directly to the IDC log

  6. Change IDC log data sets
  7. Reset logging to the beginning of the data set or appended data

  8. Pass the escape key to the host application
  9. Change the IDC escape key

Data capture status . . : ON
Current IDC log data set: SLITVAK.WTMUSER.IDCLOG(SELL900)
Current escape key  . . : PA1


F1=Help  F3=Exit  F12=Cancel
```

Figure 28-91 Change of log data sets selected

41. Overtype the name of the current IDC log data set to start a new one and press Enter as shown in Figure 28-92.

```
IDCESC6              WSim IDC:  Change IDC Log Data Sets

Type information, then press Enter.

New IDC log data set name . . . . . .  SLITVAK.WTMUSER.IDCLOG(RTQUOTE)
  If data set already exists, specify  R   (R=Replace or A=Append)









Data capture status . . . . . :  ON
Current IDC log data set  . . . . . :  SLITVAK.WTMUSER.IDCLOG(RTQUOTE)



Current IDC data set closed, new one opened
F1=Help  F3=Exit  F12=Cancel
```

Figure 28-92 Log data set for RTQUOTE is opened

42. When the switch of the IDC log files is confirmed, press PF3 to return to the host session. Select option **1** to request a quote, as shown in Figure 28-93.

```
                    Share Trading Demonstration                TRADER.T003

                   Share Trading Manager: Options

                    1. New Real-Time Quote

                    2. Buy Shares

                    3. Sell Shares



                Please select an option (1,2 or 3):  1



Request Completed OK
--------------------------------------------------------------------------------
PF3=Return                                                            PF12=Exit
```

Figure 28-93 Trader application: Quote requested

43. The results are presented in Figure 28-94.

```
                         Share Trading Demonstration                TRADER.T004

                     Share Trading Manager: Real-Time Quote

       User Name:     ZZ

       Company Name:  Casey_Import_Export

       Share Values:                     Commission Cost:
          NOW:            70.00             for Selling:        010
          1 week ago:     72.00             for Buying:         005
          6 days ago:     71.00
          5 days ago:     70.00
          4 days ago:     69.00
          3 days ago:     68.00
          2 days ago:     68.00          Number of Shares Held: 0200
          1 day ago:      69.00          Value of Shares Held:       14000.0



 Request Completed OK
 ------------------------------------------------------------------------------
 PF3=Return                                                            PF12=Exit
```

Figure 28-94 Trader application: Real time quote

44. Press PF12 to exit from the trader application. The trader application session is over as shown in Figure 28-95.

```
Trader: Session Over     Share Trading Demonstration                TRADER.T004

                     Share Trading Manager: Real-Time Quote

       User Name:     ZZ

       Company Name:  Casey_Import_Export

       Share Values:                     Commission Cost:
          NOW:            70.00             for Selling:        010
          1 week ago:     72.00             for Buying:         005
          6 days ago:     71.00
          5 days ago:     70.00
          4 days ago:     69.00
          3 days ago:     68.00
          2 days ago:     68.00          Number of Shares Held: 0200
          1 day ago:      69.00          Value of Shares Held:       14000.0



 Request Completed OK
 ------------------------------------------------------------------------------
 PF3=Return                                                            PF12=Exit
```

Figure 28-95 Trader application: Session over

45. Press the IDC escape key (PA1) to switch IDC log files again, as shown in Figure 28-96. Overtype the current name with the new one and press Enter.

```
IDCESC6              WSim IDC:  Change IDC Log Data Sets

Type information, then press Enter.

New IDC log data set name . . . . . .  SLITVAK.WTMUSER.IDCLOG(LOGOFF)
  If data set already exists, specify  R   (R=Replace or A=Append)









Data capture status . . . . . :  ON
Current IDC log data set  . . . . . :  SLITVAK.WTMUSER.IDCLOG(LOGOFF)



Current IDC data set closed, new one opened
F1=Help  F3=Exit  F12=Cancel
```

Figure 28-96 Log data set for LOGOFF opened

46. Press PF3 to return to the host application session and start the **cesf logoff** transaction to end the session with CICSC22F as shown in Figure 28-97.

```
cesf logoff_
```

Figure 28-97 CICS logoff initiated

47. Note the difference in the results of the transactions **cesf** and **cesf logoff**. In the case of the former, the IDC utility Escape Actions panel was presented and option **3** had to be selected to actually finish the host application session. In the case of the latter, the host application session is ended immediately and the IDC utility presents the confirmation message as shown in Figure 28-98.

```
IDCSSP          WSim IDC:  Start Session with Host Application

Type information, then press Enter.

Session Data
  Host application name . . . . . . . CICSC22F
  Logon mode name . . . . . . . . . . LSX32702 (Optional)
  Logon user data . . . . . . . . . . ________________________ (Optional)


IDC log data set name . . . . . . . . SLITVAK.WTMUSER.IDCLOG(LOGOFF)
  If data set already exists, specify R        (R=Replace or A=Append)

Start capturing data immediately? . . N        (Y=Yes or N=No)

IDC Escape key  . . . . . . . . . . . PA1__    (PAn, PFnn, CLEAR, or ATTN)





ITP1508I SESSION ENDED WITH APPLICATION CICSC22F
F1=Help  F3=Exit  F5=Refresh  F11=Save  F12=Cancel
```

Figure 28-98 CICS session ended

48. Press PF3. WTM performs the steps required to generate a test case and presents the list of test cases confirming that the test case TRADLOG is added. What happened to all the other test cases we were preparing when changing IDC log files? WTM automatically generated only the first of them, as shown in Figure 28-99.

```
                              Process Testcases                    Row 1 of 4
Command==> add_
Enter a line command or add to create a new Testcase.         Press PF3 to end.
Line commands: D Delete, N Notes, O Open, T Translate, U UTBLs, V View.

  Name     Type UTBLs Notes Description
_ TRADLOG   V              logon to trader appl                        Added
_ INITST7   V              start test trans, opt 7                     Added
_ LOGON     V              logon to cicsc22f                           Added
_ TSTTRAD   V              Logon,trader,logon,getquote,buystock,getqu
***************************** Bottom of data ********************************









PF 1=Help      2=Split     3=End       4=          5=Add       6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-99 Process test cases

49. We have to add them one by one using source option **3** on the WTM Add New Test case panel shown in Figure 28-100.

```
                              Process Testcases                    Row 1 of 4
                                Add New Testcase

 Type Testcase Name, Description and Source.  Then press Enter.

 Testcase Name: selcomp
 Description  : select company
 Source       : 3  1. Add a 3270 testcase using IDC
                   2. Add a 3270 testcase using an SNA trace
                   3. Add a 3270 testcase using a WSim or IDC log
                   4. Add a testcase using the WSim STL models
                   5. Add a testcase using an STL skeleton
                   6. Add a CPI-C testcase using an LU 6.2 SNA trace

 PF 1=Help      2=Split     3=End       4=          5=          6=
 PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve

PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-100 Adding new test case SELCOMP

50. Provide the test case name and description and press Enter. On the next WTM panel overtype the name of the IDC log file for the corresponding test case and press Enter, as shown in Figure 28-101.

```
                              Process Testcases                    Row 1 of 4

 Type the WSim log data set name.  Then press Enter.
 WSim log data set name : 'SLITVAK.WTMUSER.IDCLOG(selcomp)'

 PF 1=Help      2=Split     3=End       4=          5=          6=
 PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve

PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-101 Log data set for test case SELCOMP

51. WTM generates the test case SELCOMP and displays the confirmation message. Enter the command **add** to continue to add new test cases from the captured IDC log files as shown in Figure 28-102.

```
                              Process Testcases                      Row 1 of 5
Command==> add_
Enter a line command or add to create a new Testcase.         Press PF3 to end.
Line commands: D Delete, N Notes, O Open, T Translate, U UTBLs, V View.

  Name    Type UTBLs Notes Description
_ SELCOMP   V              select company                              Added
_ TRADLOG   V              logon to trader appl                        Added
_ INITST7   V              start test trans, opt 7                     Added
_ LOGON     V              logon to cicsc22f                           Added
_ TSTTRAD   V              Logon,trader,logon,getquote,buystock,getqu
******************************* Bottom of data ********************************

PF 1=Help      2=Split     3=End       4=          5=Add       6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-102 Test case SELCOMP added

52. The next test case is “BUY1000” as shown in Figure 28-103.

```
                              Process Testcases                      Row 1 of 5
                                 Add New Testcase

 Type Testcase Name, Description and Source.  Then press Enter.

 Testcase Name: buy1000
 Description  : buy 1000 shares
 Source       : 3 _1. Add a 3270 testcase using IDC
                  2. Add a 3270 testcase using an SNA trace
                  3. Add a 3270 testcase using a WSim or IDC log
                  4. Add a testcase using the WSim STL models
                  5. Add a testcase using an STL skeleton
                  6. Add a CPI-C testcase using an LU 6.2 SNA trace

 PF 1=Help      2=Split     3=End       4=          5=          6=
 PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve

PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-103 Adding test case BUY1000

53. Overtype the name of the corresponding IDC log files, as shown in Figure 28-104.

```
                              Process Testcases                        Row 1 of 5
 Type the WSim log data set name.  Then press Enter.
 WSim log data set name : 'SLITVAK.WTMUSER.IDCLOG(buy1000)'

 PF 1=Help      2=Split     3=End       4=          5=          6=
 PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
******************************** Bottom of data ********************************

 PF 1=Help      2=Split     3=End       4=          5=          6=
 PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-104 Log data set for test case BUY1000

54. The test case BUY1000 is added. Continue to add test cases as shown in Figure 28-105.

```
                              Process Testcases                        Row 1 of 6
Command==> add_
Enter a line command or add to create a new Testcase.         Press PF3 to end.
Line commands: D Delete, N Notes, O Open, T Translate, U UTBLs, V View.

  Name    Type UTBLs Notes Description
_ BUY1000   V              buy 1000 shares                              Added
_ SELCOMP   V              select company                               Added
_ TRADLOG   V              logon to trader appl                         Added
_ INITST7   V              start test trans, opt 7                      Added
_ LOGON     V              logon to cicsc22f                            Added
_ TSTTRAD   V              Logon,trader,logon,getquote,buystock,getqu
******************************** Bottom of data ********************************

 PF 1=Help      2=Split     3=End       4=          5=Add       6=
 PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-105 Test case BUY1000 added

55. The next test case is SELL900, as shown in Figure 28-106.

```
                              Process Testcases                    Row 1 of 6
                                 Add New Testcase

 Type Testcase Name, Description and Source.  Then press Enter.

 Testcase Name: sell900
 Description  : selling 900 shares
 Source       : 3 _1. Add a 3270 testcase using IDC
                  2. Add a 3270 testcase using an SNA trace
                  3. Add a 3270 testcase using a WSim or IDC log
                  4. Add a testcase using the WSim STL models
                  5. Add a testcase using an STL skeleton
                  6. Add a CPI-C testcase using an LU 6.2 SNA trace

 PF 1=Help      2=Split     3=End       4=          5=          6=
 PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve

PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-106 Adding test case SELL900

56. Overtype the name of the corresponding IDC log file, as shown in Figure 28-107.

```
                              Process Testcases                    Row 1 of 6

 Type the WSim log data set name.  Then press Enter.
 WSim log data set name : 'SLITVAK.WTMUSER.IDCLOG(sell900)'

 PF 1=Help      2=Split     3=End       4=          5=          6=
 PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve

  TSTTRAD   V              Logon,trader,logon,getquote,buystock,getqu
******************************** Bottom of data ********************************

PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-107 Log data set for test case SELL900

57. Test case SELL900 is successfully added. Continue adding test cases, as shown in Figure 28-108.

```
                              Process Testcases                        Row 1 of 7
Command==> add_
Enter a line command or add to create a new Testcase.          Press PF3 to end.
Line commands: D Delete, N Notes, O Open, T Translate, U UTBLs, V View.

  Name    Type UTBLs Notes Description
_ SELL900   V              selling 900 shares                            Added
_ BUY1000   V              buy 1000 shares                               Added
_ SELCOMP   V              select company                                Added
_ TRADLOG   V              logon to trader appl                          Added
_ INITST7   V              start test trans, opt 7                       Added
_ LOGON     V              logon to cicsc22f                             Added
_ TSTTRAD   V              Logon,trader,logon,getquote,buystock,getqu
****************************** Bottom of data ******************************

PF 1=Help      2=Split     3=End       4=          5=Add       6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-108 Test case SELL900 added

58. The next test case is RTQUOTE as shown in Figure 28-109.

```
                              Process Testcases                        Row 1 of 7
                                 Add New Testcase

 Type Testcase Name, Description and Source.  Then press Enter.

 Testcase Name: rtquote
 Description  : real time quote
 Source       : 3 _1. Add a 3270 testcase using IDC
                  2. Add a 3270 testcase using an SNA trace
                  3. Add a 3270 testcase using a WSim or IDC log
                  4. Add a testcase using the WSim STL models
                  5. Add a testcase using an STL skeleton
                  6. Add a CPI-C testcase using an LU 6.2 SNA trace

 PF 1=Help      2=Split     3=End       4=          5=          6=
 PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve

PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-109 Adding test case RTQUOTE

59. Once again, overtype the name of the IDC of data set for the test case RTQUOTE, as shown in Figure 28-110.

```
                              Process Testcases                        Row 1 of 7
 Type the WSim log data set name.  Then press Enter.
 WSim log data set name : 'SLITVAK.WTMUSER.IDCLOG(rtquote)'

 PF 1=Help      2=Split     3=End       4=          5=          6=
 PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve

  LOGON     V              logon to cicsc22f                            Added
  TSTTRAD   V              Logon,trader,logon,getquote,buystock,getqu
****************************** Bottom of data ******************************

PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-110 Log data set for test case RTQUOTE

60. WTM generates the test case, going through the required steps as shown in Figure 28-111.

```
                              Process Testcases                        Row 1 of 7
Comman                          "Milestones"
Enter                                                                  o end.
Line c  These are the steps to create a testcase from a WSim log:

  Name  Step 1 -          - Choose a resource from a WSim log.
  SELL  Step 2 - ITPLSGEN - Create an STL program from the WSim Log.   ed
  BUY1  Step 3 - ITMDCAS  - Add WSim/TM upgrades to STL program.       ed
  SELC  Step 4 - ITPSTL   - Translate the STL program into a MSGTXT.   ed
  TRAD                                                                 ed
  INIT                                                                 ed
  LOGON     V              logon to cicsc22f                          Added
  TSTTRAD   V              Logon,trader,logon,getquote,buystock,getqu
****************************** Bottom of data ******************************
```

Figure 28-111 “Milestones” panel presents test case creation steps

61. The test case RTQUOTE generation is finished and is added to the list. One test case still has to be generated as shown in Figure 28-112.

```
                              Process Testcases                        Row 1 of 8
Command==> add_
Enter a line command or add to create a new Testcase.          Press PF3 to end.
Line commands: D Delete, N Notes, O Open, T Translate, U UTBLs, V View.

  Name    Type UTBLs Notes Description
_ RTQUOTE   V              real time quote                              Added
_ SELL900   V              selling 900 shares                           Added
_ BUY1000   V              buy 1000 shares                              Added
_ SELCOMP   V              select company                               Added
_ TRADLOG   V              logon to trader appl                         Added
_ INITST7   V              start test trans, opt 7                      Added
_ LOGON     V              logon to cicsc22f                            Added
_ TSTTRAD   V              Logon,trader,logon,getquote,buystock,getqu
******************************** Bottom of data ********************************

PF 1=Help      2=Split     3=End       4=          5=Add       6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-112 Test case RTQUOTE added

62. The test case LOGOFF to log off from CICS will be generated from the log file as shown in Figure 28-113.

```
                              Process Testcases                        Row 1 of 8
                                 Add New Testcase

 Type Testcase Name, Description and Source.  Then press Enter.

 Testcase Name: logoff
 Description  : logoff from CICS
 Source       : 3 _1. Add a 3270 testcase using IDC
                  2. Add a 3270 testcase using an SNA trace
                  3. Add a 3270 testcase using a WSim or IDC log
                  4. Add a testcase using the WSim STL models
                  5. Add a testcase using an STL skeleton
                  6. Add a CPI-C testcase using an LU 6.2 SNA trace

 PF 1=Help      2=Split     3=End       4=          5=          6=
 PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve

PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-113 Adding test case LOGOFF

63. WTM asks for the name of the corresponding log data set. Overtype the previous name with the new one, as shown in Figure 28-114, and press Enter.

```
                              Process Testcases                        Row 1 of 8
 Type the WSim log data set name.  Then press Enter.
 WSim log data set name : 'SLITVAK.WTMUSER.IDCLOG(logoff)'

 PF 1=Help      2=Split     3=End       4=          5=          6=
 PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve

  INITST7   V              start test trans, opt 7                    Added
  LOGON     V              logon to cicsc22f                          Added
  TSTTRAD   V              Logon,trader,logon,getquote,buystock,getqu
****************************** Bottom of data ********************************

PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-114 Log data set for test case LOGOFF

64. All captured IDC logs are now processed. The final list of the generated test cases is presented in Figure 28-115.

```
                              Process Testcases                        Row 1 of 9
Command==>
Enter a line command or add to create a new Testcase.         Press PF3 to end.
Line commands: D Delete, N Notes, O Open, T Translate, U UTBLs, V View.

  Name    Type UTBLs Notes Description
_ LOGOFF    V              logoff from CICS                           Added
_ RTQUOTE   V              real time quote                            Added
_ SELL900   V              selling 900 shares                         Added
_ BUY1000   V              buy 1000 shares                            Added
_ SELCOMP   V              select company                             Added
_ TRADLOG   V              logon to trader appl                       Added
_ INITST7   V              start test trans, opt 7                    Added
_ LOGON     V              logon to cicsc22f                          Added
_ TSTTRAD   V              Logon,trader,logon,getquote,buystock,getqu
****************************** Bottom of data ********************************

PF 1=Help      2=Split     3=End       4=          5=Add       6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-115 All test cases added

28.7.2 Creating a test group

A *test group* is an ordered list of test items (test cases and MSGTXTs). This section describes the steps for creating a test group.

1. Select option **2**, or enter the command **`group`** from the WTM main menu. The WTM panel Process Test groups is presented as shown in Figure 28-116. There are no test groups currently, so issue the command **`add`** to create one.

```
                              Process Testgroups
Command==> add_
Enter a line command or add to create a new Testgroup.        Press PF3 to end.
Line commands: D Delete, N Notes, S Select.

  Name     Type Notes Description
****************************** Bottom of data ******************************

PF 1=Help      2=Split     3=End       4=          5=Add       6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-116 WTM Process test groups panel

The Type column has the following valid values (which are specified when a test group is added):

V VTAMAPPL
T TCP/IP
C CPI-C

2. A new test group TRADFULL, which will include a full session with the trader application, is being added as shown in Figure 28-117.

```
                              Process Testgroups
                             Add New Testgroup

 Type Testgroup Name, Description and Testgroup Type.  Then press Enter.

 Testgroup Name: tradfull
 Description   : full session with trader appl
 Testgroup Type: v _   (V=VTAMAPPL, T=TCP/IP, C=CPI-C)

 PF 1=Help      2=Split     3=End       4=          5=          6=
 PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve

PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-117 New test group TRADFULL

3. The order of test cases to be included in the test group TRADFULL is defined on the next panel, shown in Figure 28-118. It is possible to use the same test case several times. In

this exercise the test cases BUY1000 and SELL900 are used more than once in the sequence.

```
                    Specify Order for Testgroup TRADFULL
                                                                  Row 1 of 9
 Command==>
 Enter test item order or summary.      Press PF3 to end.

 Order       Name       Test...  Type  Test Item Description
 5           BUY1000    Case       V   buy 1000 shares
 2           INITST7    Case       V   start test trans, opt 7
 8           LOGOFF     Case       V   logoff from CICS
 1           LOGON      Case       V   logon to cicsc22f
 7           RTQUOTE    Case       V   real time quote
 4           SELCOMP    Case       V   select company
 6           SELL900    Case       V   selling 900 shares
 3           TRADLOG    Case       V   logon to trader appl
             TSTTRAD    Case       V   Logon,trader,logon,getquote,buystock,
 **************************** Bottom of data ****************************

 PF 1=Help      2=Split     3=End       4=          5=          6=Summary
 PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-118 Order of test cases in testgroup TRADFULL

4. The test group TRADFULL is added. Enter the line command **s** to see its content, as shown in Figure 28-119. The ordered list of the test items for the test group TRADFULL is presented in Figure 28-120.

```
                          Process Testgroups                    Row 1 of 1
Command==>
Enter a line command or add to create a new Testgroup.       Press PF3 to end.
Line commands: D Delete, N Notes, S Select.

  Name     Type Notes Description
s TRADFULL   V        full session with trader application           Added
**************************** Bottom of data ******************************

PF 1=Help      2=Split     3=End       4=          5=Add       6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-119 Testgroup TRADFULL created

```
Test Item List for Testgroup TRADFULL
                                                                    Row 1 of 8
Command==>
Enter test item order or all.          Press PF3 to end.

Order       Name       Test...  Type  Test Item Description
1           LOGON      Case       V   logon to cicsc22f
2           INITST7    Case       V   start test trans, opt 7
3           TRADLOG    Case       V   logon to trader appl
4           SELCOMP    Case       V   select company
5           BUY1000    Case       V   buy 1000 shares
6           SELL900    Case       V   selling 900 shares
7           RTQUOTE    Case       V   real time quote
8           LOGOFF     Case       V   logoff from CICS
***************************** Bottom of data ******************************

PF 1=Help      2=Split     3=End       4=          5=All       6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-120 Ordered list of test cases in TRADFULL

5. To run a simulation for the newly created test group a schedule must be added. To do this, select option **4** or issue the command **run** from the WTM main panel as shown in Figure 28-121.

```
                          WSim Test Manager

Select one of the following.  Then press Enter.

      Command   Action
_  1. CASE      Create and Process Testcases
   2. GROUP     Create and Process Testgroups
   3. CYCLE     Create and Process Testcycles
   4. RUN       Create WSim Networks and Schedule WSim Simulation Runs
   D. DOC       Create Test Documentation
   P. PROJECT   Add/Change Project or Alternate HLI
   U. UTIL      Run WSim Test Manager Utilities
   W. WII       Invoke WSim/ISPF Interface

Project: RESPROJ1                      Alternate HLI:

Command ===> run
PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-121 Starting to work with schedules

6. On the WTM Process Schedules panel enter the command **add** to create a new schedule to run the test group, as shown in Figure 28-122.

```
                          Process WSim Schedules                    Row 1 of 1
Command==> add
Enter a line command or add to create a new Schedule.        Press PF3 to end.
Line commands: D Delete, N Notes, O Output, P Preprocess, R Resources,
               S Testcases, U UTBLs, X Execute, W NTWRK.

  Name     Type Notes Description                 UTBLs   Last Run
_ TSTTRAD1   V        test tsttrad testcase            10/26/04 14:49
******************************** Bottom of data ********************************



PF 1=Help      2=Split     3=End       4=          5=Add       6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-122 Adding schedule

7. Enter a name for the new schedule, then add a description. Identify its type and press Enter as shown in Figure 28-123.

```
                          Process WSim Schedules                    Row 1 of 1
                          Add New Test Schedule

 Type Schedule Name, Description and Schedule Type.  Then press Enter.

 Schedule Name: runfull
 Description  : run tradfull group
 Schedule Type: v _   (V=VTAMAPPL, T=TCP/IP, C=CPI-C)



 PF 1=Help      2=Split     3=End       4=          5=          6=
 PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve



PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-123 Creating schedule RUNFULL

8. On the next panel, Resource List for Schedule, enter the line command **s** to select the resources on the list as shown in Figure 28-124.

```
                    Resource List for Schedule RUNFULL            Row 1 of 1
Command==>
Enter command sort to sort by VTAMname.                     Press PF3 to end.
Line commands: S Select, I Insert, R Repeat, D Delete.

  VTAMname WSIMname  Test...  Name       Description
s VAPPL00  WSIMLU00
******************************** Bottom of data ********************************

PF 1=Help      2=Split     3=End       4=          5=Sort      6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-124 Simulated display WSIMLU00 is selected to specify test items

9. The only test item that will be used in this simulation is the test group TRADFULL, so its order **1** is entered, as shown in Figure 28-125.

```
Specify Order for Resource WSIMLU00 under VAPPL00
                                                              Row 1 of 10
Command==>
Enter order or summary.          Press PF3 to end.

Order                  Name      Test...  Type  Description
                       BUY1000   Case      V    buy 1000 shares
                       INITST7   Case      V    start test trans, opt 7
                       LOGOFF    Case      V    logoff from CICS
                       LOGON     Case      V    logon to cicsc22f
                       RTQUOTE   Case      V    real time quote
                       SELCOMP   Case      V    select company
                       SELL900   Case      V    selling 900 shares
                       TRADLOG   Case      V    logon to trader appl
                       TSTTRAD   Case      V    Logon,trader,logon,getquot
1                      TRADFULL  Group     V    full session with trader a
***************************** Bottom of data ******************************

PF 1=Help      2=Split     3=End       4=          5=          6=Summary
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-125 Testgroup assigned to WSIMLU00

10. The resource list for the schedule RUNFULL is now defined as shown in Figure 28-126.

```
                    Resource List for Schedule RUNFULL            Row 1 of 1
Command==>
Enter command sort to sort by VTAMname.                     Press PF3 to end.
Line commands: S Select, I Insert, R Repeat, D Delete.

  VTAMname WSIMname  Test...  Name       Description
_ VAPPL00  WSIMLU00  GROUP    TRADFULL   FULL SESSION WITH TRADER APPL
****************************** Bottom of data ********************************

PF 1=Help      2=Split     3=End       4=          5=Sort      6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-126 Test items assigned to schedule RUNFULL

11. Press PF3. Since this is the schedule that is being added, WTM presents the Completion Report Response Time Thresholds panel to enter threshold values as shown in Figure 28-127.

```
                Completion Report Response Time Thresholds       Values saved
Command==>
Update this panel, then press Enter.  Press PF3 to end.

Response Time:                              Percentiles:
        Mean   < 0.1__ seconds                 95  % < 0.2__ seconds
        Median < _____   "                     90  % < _____   "
        Mode   < _____   "                     80  % < _____   "
        High   < _____   "                     70  % < _____   "

PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-127 Completion reports time thresholds panel

12. The new schedule RUNFULL is created and is ready to be run. Enter the line command **X** to initiate the simulation as shown in Figure 28-128.

```
                          Process WSim Schedules                    Row 1 of 2
Command==>
Enter a line command or add to create a new Schedule.        Press PF3 to end.
Line commands: D Delete, N Notes, O Output, P Preprocess, R Resources,
               S Testcases, U UTBLs, X Execute, W NTWRK.

  Name     Type Notes Description                 UTBLs   Last Run
x RUNFULL    V        run tradfull group                          Added
_ TSTTRAD1   V        test tsttrad testcase            10/26/04 14:49
****************************** Bottom of data ********************************



PF 1=Help      2=Split     3=End       4=          5=Add       6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-128 Executing schedule RUNFULL

13. On the next panel, shown in Figure 28-129, WTM gives the user an opportunity to define the log data set names that will be used for this simulation. Press Enter to proceed with the simulation.

```
                          Process WSim Schedules                    Row 1 of 2
                           Specify WSim Log Names
 Change the lognames and press Enter to continue or PF3 to end.

 ITPENTER: 'SLITVAK.WTMTEST.RESPROJ1.MLOG.RUNFULL'      Mandatory
 Baseline: ____________________________________________ Optional

 Completion Report (Y/N): Y


 PF 1=Help      2=Split     3=End       4=          5=          6=
 PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve



PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-129 Simulation log data set name

14. The WTM starts the simulation and displays the WSim run-time progress messages (the DISPLAY mode was specified). The simulation is successfully completed, as shown in Figure 28-130.

```
ITP016I Workload Simulator (WSim) Version 1, Release 1.0, May 23, 2002, 11:05

ITP003I WSim INITIALIZATION COMPLETE
ITP200I DISPLAY MONITOR FACILITY ACTIVE USING APPL WTMDM00
ITP029I INITIALIZATION COMPLETE FOR NETWORK RUNFULL
ITP006I NETWORK RUNFULL STARTED
ITP137I RUNFULL  WSIMLU00-00001 - WTM: LOGON Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: INITST7 Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: TRADLOG Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: SELCOMP Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: BUY1000 Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: SELL900 Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: RTQUOTE Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: LOGOFF Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTMEND: 1 out of 1 finished.
ITP137I RUNFULL  WSIMLU00-00001 - WTMEND: All scripts run.  Issuing ZEND.
ITP002I RUNFULL  WSIMLU00-00001 - ZEND
ITP201I DISPLAY MONITOR FACILITY IS CLOSED DOWN
ITP079I WSim IS CLOSED DOWN
***
```

Figure 28-130 Simulation run messages for RUNFULL

15. The requested optional completion report for the simulation is shown in Figure 28-131.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       SLITVAK.WTMTEST.RESPROJ1.CREP(R12) - 01.00      Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
****** ***************************** Top of Data ******************************
000001 Report for RUNFULL - run tradfull group.
000002 Run completed at 15:36:08 on 28 Oct 2004.
000003 This run successfully met the response time targets:
000004   95%_Percentile actual (0.10)  target(0.2)
000005   Mean actual (0.01)  target(0.1)
000006 Loglist Extract:
000007 RUN  TIME 15.35.32,   OCTOBER 28, 2004  VERSION  1  RELEASE  1.0.0
000008 15354131 ITP137I RUNFULL  WSIMLU00-00001 - WTM: LOGON Finished
000009 15354532 ITP137I RUNFULL  WSIMLU00-00001 - WTM: INITST7 Finished
000010 15354732 ITP137I RUNFULL  WSIMLU00-00001 - WTM: TRADLOG Finished
000011 15354933 ITP137I RUNFULL  WSIMLU00-00001 - WTM: SELCOMP Finished
000012 15355334 ITP137I RUNFULL  WSIMLU00-00001 - WTM: BUY1000 Finished
000013 15355735 ITP137I RUNFULL  WSIMLU00-00001 - WTM: SELL900 Finished
000014 15360337 ITP137I RUNFULL  WSIMLU00-00001 - WTM: RTQUOTE Finished
000015 15360537 ITP137I RUNFULL  WSIMLU00-00001 - WTM: LOGOFF Finished
000016 Comments:
000017
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 28-131 Simulation report

16. Press PF3 to return to the list of the available schedules, shown in Figure 28-132.

```
                          Process WSim Schedules                    Row 1 of 2
Command==>
Enter a line command or add to create a new Schedule.        Press PF3 to end.
Line commands: D Delete, N Notes, O Output, P Preprocess, R Resources,
               S Testcases, U UTBLs, X Execute, W NTWRK.

  Name     Type Notes Description                UTBLs   Last Run
o RUNFULL    V        run tradfull group               10/28/04 15:35 Execute
_ TSTTRAD1   V        test tsttrad testcase            10/28/04 13:55
****************************** Bottom of data ********************************



PF 1=Help      2=Split     3=End       4=          5=Add       6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-132 RUNFULL simulation executed

17. To see the output from the simulation enter the line command **o**. Available output reports from simulations were discussed previously.

28.8 User data tables

A User Data Table (UTBL) is a list of string constants, for example, a list of client or account numbers, or a list of user IDs and passwords. The WSim Test Manager organizes UTBLs into fields and each line may contain one or more fields.

WTM can automatically generate UTBLs and the STL programs to use them. This is done by editing the source of STL programs in the WTM and using the command **utbl**.

There are three options available when the user table function is invoked in WTM:

- Create a new UTBL.
- Create a new field in the existing UTBL.
- Use an existing field or UTBL.

There are three types of access to data in an UTBL:

- **Random**: STL code is accessing UTBL randomly.
- **Single sequential**: STL code is accessing UTBL in strictly sequential order with only one pass. When the end of the UTBL is reached, further access is not allowed.
- **Single sequential repeated**: STL code is accessing UTBL in strictly sequential order in multiple passes from the first to the last record.

In this section we take you through the process of creating a UTBL, so instead of using hard-coded data a script can access a UTBL.

1. Start with the command **case** from the WTM main panel shown in Figure 28-133.

```
                              WSim Test Manager

Select one of the following.  Then press Enter.

      Command   Action
_  1. CASE      Create and Process Testcases
   2. GROUP     Create and Process Testgroups
   3. CYCLE     Create and Process Testcycles
   4. RUN       Create WSim Networks and Schedule WSim Simulation Runs
   D. DOC       Create Test Documentation
   P. PROJECT   Add/Change Project or Alternate HLI
   U. UTIL      Run WSim Test Manager Utilities
   W. WII       Invoke WSim/ISPF Interface

Project: RESPROJ1                       Alternate HLI:




Command ===> case
PF 1=Help       2=Split      3=End        4=           5=           6=
PF 7=Up         8=Down       9=Swap      10=Left      11=Right     12=Retrieve
```

Figure 28-133 Command CASE to work with test cases

2. Enter the line command **o** to open the corresponding test case STL code and edit it. This example is working with the LOGON test case, as shown in Figure 28-134.

```
                              Process Testcases                        Row 1 of 9
Command==>
Enter a line command or add to create a new Testcase.          Press PF3 to end.
Line commands: D Delete, N Notes, O Open, T Translate, U UTBLs, V View.

  Name     Type UTBLs Notes Description
_ BUY1000   V               buy 1000 shares
_ INITST7   V               start test trans, opt 7
_ LOGOFF    V               logoff from CICS
o LOGON     V               logon to cicsc22f
_ RTQUOTE   V               real time quote
_ SELCOMP   V               select company
_ SELL900   V               selling 900 shares
_ TRADLOG   V               logon to trader appl
_ TSTTRAD   V               Logon,trader,logon,getquote,buystock,getqu
****************************** Bottom of data ********************************




PF 1=Help       2=Split      3=End        4=           5=Add        6=
PF 7=Up         8=Down       9=Swap      10=Left      11=Right     12=Retrieve
```

Figure 28-134 Opening test case LOGON

3. The source of the STL program for the test case LOGON is opened in an ISPF editing session as shown in Figure 28-135.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------
EDIT       SLITVAK.WTMTEST.RESPROJ1.STL(LOGON) - 01.01     Columns 00001 00072
Command ===> ____________________________________________________ Scroll ===> CSR
****** **************************** Top of Data ******************************
000001 @program=LOGONT
000002 @include wtmvars
000003 LOGON: msgtxt
000004 /*-------------------------------------------------------------------*/
000005 /* ITPIDC: DISPLAY=TSOCON   APPLICATION=CICSC22F 14:59:08.52 10/28/04*/
000006 /*   ----------- DISPLAY CHARACTERISTICS AND FEATURES ------------   */
000007 /*   ALTCSET=APL                APLCSID=(963,310)                    */
000008 /*   BASECSID=(697,37)          CCSIZE=(8,15)     COLOR=MULTI        */
000009 /*   DBCS=NO                                                         */
000010 /*   DISPLAY=(24,80,24,80)      DLOGMOD=LSX32702  EXTFUN=YES         */
000011 /*   FLDOUTLN=NO                FLDVALID=NO       HIGHLITE=YES       */
000012 /*   MAXNOPTN=0                 PS=NONE           UOM=INCH           */
000013 /*-------------------------------------------------------------------*/
000014 /* ITPLSGEN: SCRIPT GENERATION PARAMETERS        15:05:48.46 10/28/04*/
000015 /*   INPUT    SLITVAK.WTMUSER.IDCLOG(LOGON)                         */
000016 /*   OUTPUT   SLITVAK.WTMTEST.RESPROJ1.STL                          */
000017 /*   MSGTXT   LOGON                                                 */
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 28-135 STL program for test case LOGON

4. Scroll down using the PF8 key and find the STL statement `type`. Enter the **`utbl`** command on the ISPF command line and place the cursor on the line where a UTBL will be used. Press Enter. The WTM pop-up panel Create WSim User Table appears.
5. Select option **1** to create a new UTBL, as shown in Figure 28-136.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------
|                      Create WSim User Table (UTBL)                      |
|                                                                         |
| Select an option.  Then press Enter.                                    |
|                                                                         |
| 1  1.  Create a new UTBL.                                               |
|    2.  Create a new field in an existing UTBL.                          |
|    3.  Use an existing field or UTBL.                                   |
|                                                                         |
|                                                                         |
| PF 1=Help      2=Split     3=End       4=          5=          6=       |
| PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve |
 ------------------------------------------------------------------------
****** **************************** Bottom of Data ***************************

 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 28-136 Creating UTBL

6. Provide the information about the UTBL, such as the name and description, field (there may be more than one) name and description, field delimiter (use the default), and access type. Press Enter as shown in Figure 28-137.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
- ┌──────────────────────── Create WSim User Table (UTBL) ────────────────────────┐ ────
E │                                                                                │ 0072
C │ Type the values for the following fields.  Then press Enter.                   │ CSR
0 │                                                                                │ 3 */
0 │ User Table Name        : cicsuis                                               │
0 │ Table Description      : cics user ids and passwords                           │
0 │                                                                                │
0 │ Field Name             : userid  __                                            │
0 │ Field Description      : cics user ids_________________________                │
0 │ Field Delimiter        : ¢ __                                                  │
0 │                                                                                │
0 │ Access Type (1,2 or 3) : 2  1.  Random                                         │
* │                             2.  Single Sequential                              │ *****
  │                             3.  Single Sequential (repeated)                   │
  │                                                                                │
  │                                                                                │
  │ PF 1=Help   2=Split  3=End    4=       5=       6=       7=Up                  │
  │ PF 8=Down   9=Swap  10=Left  11=Right                                          │
  └────────────────────────────────────────────────────────────────────────────────┘

 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 28-137 UTBL CICSUIS

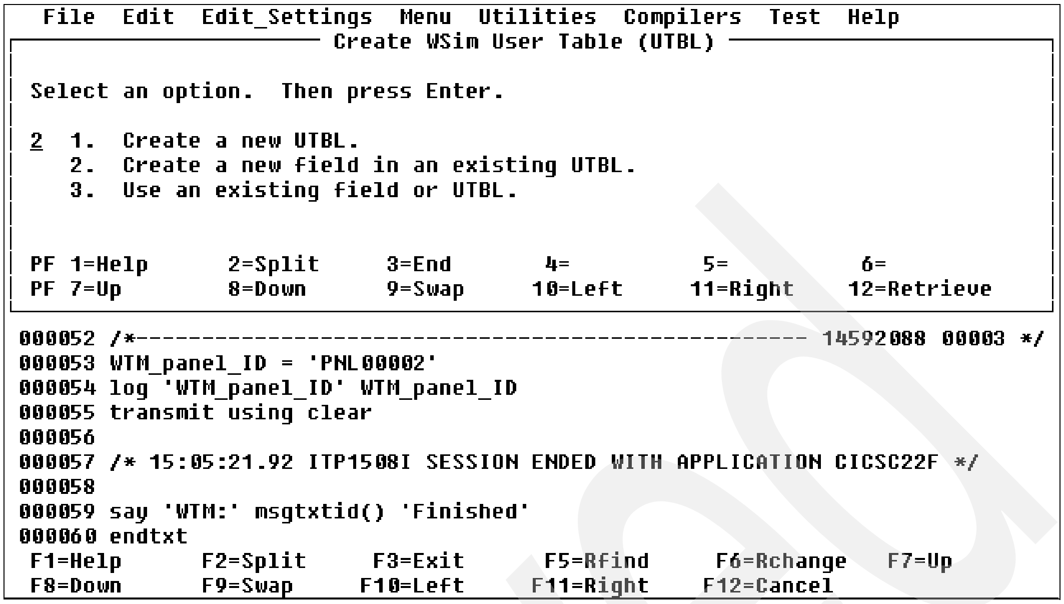
Attention: The access type selected is **2** (single sequential), which allows just one pass through the UTBL CICSUIS. This choice will have an impact on the load test which we discuss later in this chapter and in which we reuse the test case LOGON.

7. WTM makes the appropriate changes to the STL code and presents the updated program, as shown in Figure 28-138.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       SLITVAK.WTMTEST.RESPROJ1.STL(LOGON) - 01.02     Columns 00001 00072
Command ===> ____________________________________________ Scroll ===> CSR
000031  wait until onin
000032 end
000033 deact onin0001
000034
000035 /* 14:59:08.54 ITP1507I SESSION STARTED WITH APPLICATION CICSC22F */
000036
000037 /*------------------------------------------------- 14591780 00001 */
000038 WTM_panel_ID = 'PNL00001'
000039 log 'WTM_panel_ID' WTM_panel_ID
000040 cursor(10,26)
000041 ereof
000042 charset 'field'
000043 /* WTM has replaced the following line with a user table: */
000044 /* type 'slitvak' */
000045 call CICSUISX     /* Access CICSUIS usertable */          /* WTM */
000046 type CICSUIS_USERID                                        /* WTM */
000047 cursor(11,26)
000048 ereof
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 28-138 Changes to STL program for LOGON

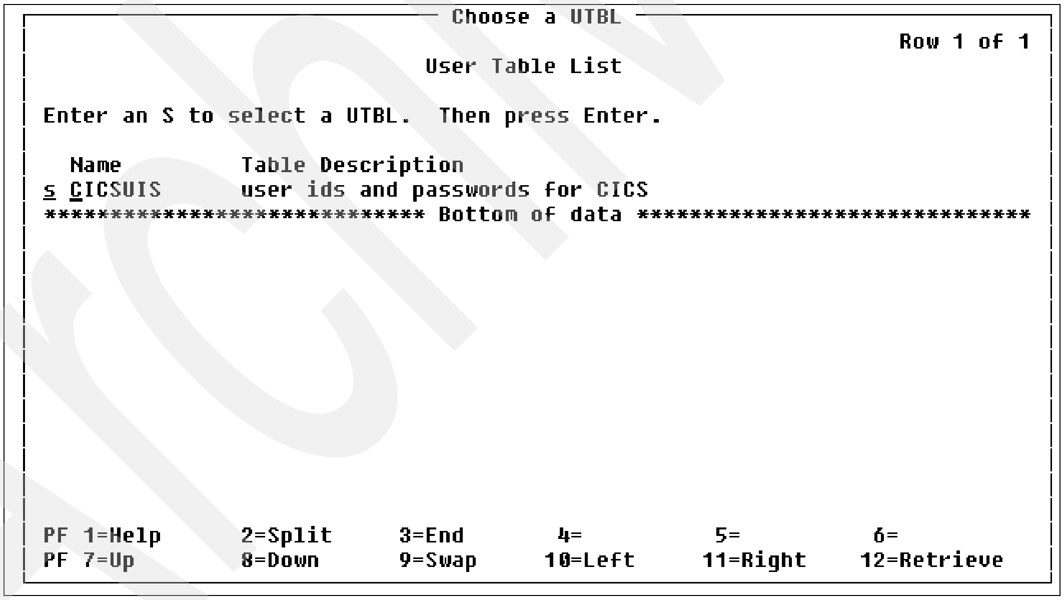
8. Add another field to the just created CICSUIS UTBL. Enter the command **utbl** on the ISPF command line and place the cursor at the next `type` STL statement. Press Enter.
9. On the pop-up panel that appears, select option **2** to create a new field in the existing UTBL and press Enter as shown in Figure 28-139.



```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------ Create WSim User Table (UTBL) ---------------------
|                                                                           |
| Select an option.  Then press Enter.                                      |
|                                                                           |
| 2  1.  Create a new UTBL.                                                 |
|    2.  Create a new field in an existing UTBL.                            |
|    3.  Use an existing field or UTBL.                                     |
|                                                                           |
|                                                                           |
| PF 1=Help      2=Split     3=End       4=         5=         6=          |
| PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve |
 ---------------------------------------------------------------------------
000052 /*---------------------------------------------- 14592088 00003 */
000053 WTM_panel_ID = 'PNL00002'
000054 log 'WTM_panel_ID' WTM_panel_ID
000055 transmit using clear
000056
000057 /* 15:05:21.92 ITP1508I SESSION ENDED WITH APPLICATION CICSC22F */
000058
000059 say 'WTM:' msgtxtid() 'Finished'
000060 endtxt
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 28-139 Adding new field to existing UTBL

10. The WTM UTBL selection panel appears as shown in Figure 28-140. Enter the line command **s** in front of the appropriate UTBL and press Enter.



```
 -------------------------------- Choose a UTBL ----------------------------
|                                                               Row 1 of 1 |
|                              User Table List                             |
|                                                                          |
| Enter an S to select a UTBL.  Then press Enter.                          |
|                                                                          |
|   Name          Table Description                                        |
| s CICSUIS      user ids and passwords for CICS                           |
| ****************************** Bottom of data ******************************
|                                                                          |
|                                                                          |
| PF 1=Help      2=Split     3=End       4=         5=         6=         |
| PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve|
 --------------------------------------------------------------------------
```

Figure 28-140 UTBL CICSUIS selected

11. Provide the information for the new field in the CICSUIS UTBL, like name, description, and delimiter. Press Enter as shown in Figure 28-141.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ┌──────────────────────── Create WSim User Table (UTBL) ────────────────────────┐
E │                                                                               │ 0072
C │ Type the values for the following fields.  Then press Enter.                  │ CSR
0 │                                                                               │
0 │ User Table Name        : CICSUIS                                              │
0 │                                                                               │
0 │ Field Name             : password                                             │
0 │ Field Description      : CICS user passwords_____________________             │
0 │ Field Delimiter        : ¢__                                                  │
0 │                                                                               │
0 │                                                                               │
0 │ PF 1=Help   2=Split  3=End    4=       5=       6=       7=Up                 │
0 │ PF 8=Down   9=Swap  10=Left  11=Right                                         │ 3 */
0 └───────────────────────────────────────────────────────────────────────────────┘
000054 log 'WTM_panel_ID' WTM_panel_ID
000055 transmit using clear
000056
000057 /* 15:05:21.92 ITP1508I SESSION ENDED WITH APPLICATION CICSC22F */
000058
000059 say 'WTM:' msgtxtid() 'Finished'
000060 endtxt
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 28-141 New field in UTBL CICUIS

12. WTM added a new statement to the STL program, as shown in Figure 28-142. Press PF3 to save the changes made by WTM and exit from the ISPF editing session.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       SLITVAK.WTMTEST.RESPROJ1.STL(LOGON) - 01.02     Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
000043 /* WTM has replaced the following line with a user table: */
000044 /* type 'slitvak' */
000045 call CICSUISX     /* Access CICSUIS usertable */          /* WTM */
000046 type CICSUIS_USERID                                       /* WTM */
000047 cursor(11,26)
000048 ereof
000049 /* WTM has replaced the following line with a user table: */
000050 /* type '*******' */
000051 type CICSUIS_PASSWORD                                     /* WTM */
000052 transmit using enter
000053
000054 /*------------------------------------------------- 14592088 00003 */
000055 WTM_panel_ID = 'PNL00002'
000056 log 'WTM_panel_ID' WTM_panel_ID
000057 transmit using clear
000058
000059 /* 15:05:21.92 ITP1508I SESSION ENDED WITH APPLICATION CICSC22F */
000060
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 28-142 More changes in STL program for LOGON

13. WTM returns to the Process Test cases panel and displays that one UTBL now will be used by the test case LOGON. Enter the line command **u** to begin working with UTBLs as shown in Figure 28-143.

```
                              Process Testcases                      Row 1 of 9
Command==>
Enter a line command or add to create a new Testcase.        Press PF3 to end.
Line commands: D Delete, N Notes, O Open, T Translate, U UTBLs, V View.

  Name    Type UTBLs Notes Description
_ BUY1000   V              buy 1000 shares
_ INITST7   V              start test trans, opt 7
_ LOGOFF    V              logoff from CICS
u LOGON     V      1       logon to cicsc22f                          Open
_ RTQUOTE   V              real time quote
_ SELCOMP   V              select company
_ SELL900   V              selling 900 shares
_ TRADLOG   V              logon to trader appl
_ TSTTRAD   V              Logon,trader,logon,getquote,buystock,getqu
****************************** Bottom of data ******************************




PF 1=Help       2=Split      3=End        4=           5=Add        6=
PF 7=Up         8=Down       9=Swap      10=Left      11=Right     12=Retrieve
```

Figure 28-143 Starting work with UTBL used in LOGON

14. On the WTM User Table List panel enter the line command **s** to start working with the STL code corresponding to the UTBL used in the test case LOGON (there is only one at this time) as shown in Figure 28-144 and press Enter.

```
                              User Table List                        Row 1 of 1
Command==>
Enter a line command or all.                 Press PF3 to end.
Line commands: S STL, M MSGTXT, D Delete.

  Name         Table Description
s CICSUIS      user ids and passwords for CICS
****************************** Bottom of data ******************************




PF 1=Help       2=Split      3=End        4=           5=All        6=
PF 7=Up         8=Down       9=Swap      10=Left      11=Right     12=Retrieve
```

Figure 28-144 UTBL CICSUIS is selected

15. An ISPF editing session is opened for the STL code of the UTBL CICSUSIS. The actual user ID and password are on line 7 as shown in Figure 28-145.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------
EDIT       SLITVAK.WTMTEST.RESPROJ1.STL(CICSUIS) - 01.01   Columns 00001 00072
Command ===> ____________________________________________________ Scroll ===> CSR
****** **************************** Top of Data ******************************
000001 /* This user table was created by WSim/TM to be accessed */
000002 /* by the Single Sequential method. */
000003 /* Table Description: user ids and passwords for CICS */
000004 /* WTM_FIELD: USERID ¢ user id for CICS */
000005 /* WTM_FIELD: PASSWORD ¢ CICS user passwords */
000006 CICSUIS: msgutbl
000007 'slitvak¢*******¢'
000008 endutbl
****** **************************** Bottom of Data ***************************
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 28-145 Editing CICSUIS

16. Add one more line with the second user ID and the password (masked by asterisks for obvious reasons), as shown in Figure 28-146. Press PF3 to save the changes to the UTBL CICSUIS.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------
EDIT       SLITVAK.WTMTEST.RESPROJ1.STL(CICSUIS) - 01.02   Columns 00001 00072
Command ===> ____________________________________________________ Scroll ===> CSR
****** **************************** Top of Data ******************************
000001 /* This user table was created by WSim/TM to be accessed */
000002 /* by the Single Sequential method. */
000003 /* Table Description: user ids and passwords for CICS */
000004 /* WTM_FIELD: USERID ¢ user id for CICS */
000005 /* WTM_FIELD: PASSWORD ¢ CICS user passwords */
000006 CICSUIS: msgutbl
000007 'slitvak¢*******¢'
000008 'slitva2¢*******¢'
000009 endutbl
****** **************************** Bottom of Data ***************************
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 28-146 Second line with user id and password added

17. WTM returns to the User Table List panel shown in Figure 28-147. Press PF3.

```
                              User Table List                          Row 1 of 1
Command==>
Enter a line command or all.                Press PF3 to end.
Line commands: S STL, M MSGTXT, D Delete.

  Name          Table Description
_ CICSUIS       user ids and passwords for CICS                          STL
******************************** Bottom of data ********************************



PF 1=Help       2=Split      3=End        4=           5=All        6=
PF 7=Up         8=Down       9=Swap      10=Left      11=Right     12=Retrieve
```

Figure 28-147 STL program for CICSUIS saved

18. WTM returns to the test cases list, as shown in Figure 28-148.

```
                              Process Testcases                        Row 1 of 9
Command==>
Enter a line command or add to create a new Testcase.          Press PF3 to end.
Line commands: D Delete, N Notes, O Open, T Translate, U UTBLs, V View.

  Name    Type UTBLs Notes Description
_ BUY1000   V              buy 1000 shares
_ INITST7   V              start test trans, opt 7
_ LOGOFF    V              logoff from CICS
_ LOGON     V       1      logon to cicsc22f                              UserTable
_ RTQUOTE   V              real time quote
_ SELCOMP   V              select company
_ SELL900   V              selling 900 shares
_ TRADLOG   V              logon to trader appl
_ TSTTRAD   V              Logon,trader,logon,getquote,buystock,getqu
******************************** Bottom of data ********************************



PF 1=Help       2=Split      3=End        4=           5=Add        6=
PF 7=Up         8=Down       9=Swap      10=Left      11=Right     12=Retrieve
```

Figure 28-148 List of all test cases

19. Press PF3 again to return back to the WTM main panel. Enter the command **run** or select option **4** to run a simulation to test the new version of the LOGON test case as shown in Figure 28-149.

```
                          WSim Test Manager

Select one of the following.  Then press Enter.

      Command   Action
_  1. CASE      Create and Process Testcases
   2. GROUP     Create and Process Testgroups
   3. CYCLE     Create and Process Testcycles
   4. RUN       Create WSim Networks and Schedule WSim Simulation Runs
   D. DOC       Create Test Documentation
   P. PROJECT   Add/Change Project or Alternate HLI
   U. UTIL      Run WSim Test Manager Utilities
   W. WII       Invoke WSim/ISPF Interface

Project: RESPROJ1                      Alternate HLI:





Command ===> run
PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-149 RUN command from WTM main panel

20. Enter the line command **r** to work with resources for the schedule RUNALL, as shown in Figure 28-150.

```
                          Process WSim Schedules                  Row 1 of 2
Command==>
Enter a line command or add to create a new Schedule.        Press PF3 to end.
Line commands: D Delete, N Notes, O Output, P Preprocess, R Resources,
               S Testcases, U UTBLs, X Execute, W NTWRK.

  Name     Type Notes Description                UTBLs   Last Run
r RUNFULL    U        run tradfull group               10/28/04 17:00
_ TSTTRAD1   U        test tsttrad testcase            10/28/04 13:55
****************************** Bottom of data ******************************









PF 1=Help      2=Split     3=End       4=          5=Add       6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-150 Working with resources for RUNFULL

21. Enter the line command **s** to work with the network resource, as shown in Figure 28-151.

```
                    Resource List for Schedule RUNFULL          Row 1 of 1
Command==>
Enter command sort to sort by VTAMname.                     Press PF3 to end.
Line commands: S Select, I Insert, R Repeat, D Delete.

  VTAMname WSIMname  Test...  Name       Description
s VAPPL00  WSIMLU00  GROUP    TRADFULL   FULL SESSION WITH TRADER APPLICATION
******************************** Bottom of data ********************************

PF 1=Help      2=Split     3=End       4=          5=Sort      6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-151 Simulated resource (terminal) WSIMLU00 is selected

22. You want the test group TRADFULL to be executed twice in a row on this simulated terminal, as shown in Figure 28-152.

```
Specify Order for Resource WSIMLU00 under VAPPL00
                                                          Row 1 of 10
Command==>
Enter order or summary.          Press PF3 to end.

Order                  Name      Test...  Type  Description
                       BUY1000   Case      V    buy 1000 shares
                       INITST7   Case      V    start test trans, opt 7
                       LOGOFF    Case      V    logoff from CICS
                       LOGON     Case      V    logon to cicsc22f
                       RTQUOTE   Case      V    real time quote
                       SELCOMP   Case      V    select company
                       SELL900   Case      V    selling 900 shares
                       TRADLOG   Case      V    logon to trader appl
                       TSTTRAD   Case      V    Logon,trader,logon,getquot
1 2_                   TRADFULL  Group     V    full session with trader a
***************************** Bottom of data ******************************

PF 1=Help      2=Split     3=End       4=          5=          6=Summary
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-152 TRADFULL to be executed twice

23. WTM repeats the network definition to make the script run two times, as shown in Figure 28-153. Press PF3.

```
                    Resource List for Schedule RUNFULL          Row 1 of 2
Command==>
Enter command sort to sort by VTAMname.                     Press PF3 to end.
Line commands: S Select, I Insert, R Repeat, D Delete.

  VTAMname WSIMname  Test...  Name       Description
_ VAPPL00  WSIMLU00  GROUP    TRADFULL   FULL SESSION WITH TRADER APPLICATION
    "        "       GROUP    TRADFULL   FULL SESSION WITH TRADER APPLICATION
****************************** Bottom of data ******************************

PF 1=Help      2=Split     3=End       4=          5=Sort      6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-153 WSIMLU00 will be used twice

24. WTM displays a warning about forthcoming changes to the network definition for the schedule RUNFULL, shown in Figure 28-154. We did not make any manual changes to the generated source code in this case so there is no reason for concern. Press Enter to continue.

```
About to recreate WSim network RUNFULL

Recreation of a WSim network involves the following:

1. Restoring the original NTWRK statement definitions.

2. Saving other statements except the UTBL, PATH and resource statements.
   Each saved statement will become a comment in the recreated WSim
   network.  Comments in the original WSim network are not saved.

3. Current UTBL and PATH statements will be replaced with values based on
   the reference order selected for all the resources in the WSim network.
   The referenced user tables are reflected in the UTBL statements.

4. Updating the resource definitions to reflect changes made to the
   WSIM and VTAM names.  Current path definitions will be replaced with
   path definitions that reflect the reference order that has been
   specified for each resource.  In addition, if a path operand exists
   on the NTWRK statement, it will be removed.

Press Enter to continue or PF3 to cancel.
```

Figure 28-154 Warning messages about changes to RUNFULL

25. The schedule RUNFULL is updated and ready to run. Enter the **x** line command to execute it as shown in Figure 28-155.

```
                          Process WSim Schedules                    Row 1 of 2
Command==>
Enter a line command or add to create a new Schedule.        Press PF3 to end.
Line commands: D Delete, N Notes, O Output, P Preprocess, R Resources,
               S Testcases, U UTBLs, X Execute, W NTWRK.

  Name     Type Notes Description                 UTBLs   Last Run
x RUNFULL    V        run tradfull group             1  10/28/04 17:00 Resources
_ TSTTRAD1   V        test tsttrad testcase             10/28/04 13:55
****************************** Bottom of data ********************************









PF 1=Help      2=Split     3=End       4=          5=Add       6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-155 RUNFULL run requested

26. On the next WTM panel, change the log names if desired and press Enter to continue with the execution. WSim starts the simulation and executes all the test cases twice, as shown in the Figure 28-156.

```
ITP016I Workload Simulator (WSim) Version 1, Release 1.0, May 23, 2002, 11:05

ITP003I WSim INITIALIZATION COMPLETE
ITP200I DISPLAY MONITOR FACILITY ACTIVE USING APPL WTMDM00
ITP029I INITIALIZATION COMPLETE FOR NETWORK RUNFULL
ITP006I NETWORK RUNFULL STARTED
ITP137I RUNFULL  WSIMLU00-00001 - WTM: LOGON Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: INITST7 Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: TRADLOG Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: SELCOMP Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: BUY1000 Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: SELL900 Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: RTQUOTE Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: LOGOFF Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: LOGON Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: INITST7 Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: TRADLOG Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: SELCOMP Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: BUY1000 Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: SELL900 Finished
ITP077I NETWORK RUNFULL MESSAGE RATES 43 RECEIVED, 32 SENT
ITP137I RUNFULL  WSIMLU00-00001 - WTM: RTQUOTE Finished
ITP137I RUNFULL  WSIMLU00-00001 - WTM: LOGOFF Finished
***
```

Figure 28-156 RUNFULL simulation messages

27. Press Enter to continue to the next screen, shown in Figure 28-157.

```
ITP137I RUNFULL  WSIMLU00-00001 - WTMEND: 1 out of 1 finished.
ITP137I RUNFULL  WSIMLU00-00001 - WTMEND: All scripts run.  Issuing ZEND.
ITP002I RUNFULL  WSIMLU00-00001 - ZEND
ITP201I DISPLAY MONITOR FACILITY IS CLOSED DOWN
ITP079I WSim IS CLOSED DOWN
*** _
```

Figure 28-157 RUNFULL simulation messages (continued)

28. We skip presenting here the optional completion report and proceed to the simulation output as shown in Figure 28-158.

```
                          Process WSim Schedules                    Row 1 of 2
Command==>
Enter a line command or add to create a new Schedule.        Press PF3 to end.
Line commands: D Delete, N Notes, O Output, P Preprocess, R Resources,
               S Testcases, U UTBLs, X Execute, W NTWRK.

  Name     Type Notes Description                 UTBLs   Last Run
o RUNFULL    V        run tradfull group             1  10/29/04 15:55 Execute
_ TSTTRAD1   V        test tsttrad testcase            10/28/04 13:55
****************************** Bottom of data ******************************


PF 1=Help      2=Split     3=End       4=          5=Add       6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-158 RUNFULL executed

29. Select option **3** to see the Display Monitor log as shown in Figure 28-159.

```
                         Reports for schedule RUNFULL

Select one of the following.  Then press Enter.

      Command   Action                                                More:  +
3  1. TLOG      View loglist report from the latest WSim run
   2. TRESP     View response time report from the latest WSim run
   3. TDM       Log display monitor for the latest WSim run
   4. TSP       View SYSPRINT from the latest WSim run

   5. MLOG      View baseline loglist report
   6. MRESP     View baseline response time report
   7. MDM       Log display monitor for the baseline log

   8. COMP      View screen compare report
   9. CDM       Log display comparator
  10. RTCOMP    Edit response time compare report

  11. COMPREP   Edit completion reports


Command===> _____________________________________________________________
PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=          8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-159 WTM Reports panel

30. Select to view the index, as shown in Figure 28-160.

```
                         Reports for schedule RUNFULL
                            Panel Display Options

 Press ENTER to continue or PF3 to cancel.

 Index or View  (I/V): i

 For View only:
   Logged or Fixed Delay (L/F): F
   Fixed Delay Value in Seconds: 2
   Display Panels    :    1 to 59



 PF 1=Help      2=Split     3=End       4=          5=          6=
 PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve

  11. COMPREP   Edit completion reports


Command===>
PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-160 Options for log display monitor

31. Enter the line command **v** to view the screen captured when executing the test case LOGON when presented with the index. The user ID is the first one from the UTBL CICSUIS, as shown in the Figure 28-161.

```
                          Signon to CICS                        APPLID CICSC22F

IBM'S  INTERNAL  SYSTEMS  MUST  ONLY  BE  USED  FOR  CONDUCTING
IBM'S BUSINESS OR FOR PURPOSES AUTHORIZED BY IBM MANAGEMENT



Type your userid and password, then press ENTER:

         Userid . . . . slitvak    Groupid . . . ________
         Password . . . ________
         Language . . . ___

     New Password . . . ________



DFHCE3520 Please type your userid.
F3=Exit                                                                  ENTER
```

Figure 28-161 First captured trader application logon screen

32. Press PF3 to return to the index of screens, scroll down to the next logon screen and use the line command **v** to view it, too, as shown Figure 28-162.

```
                          Resource Display Index                Row 18 of 59
Command==>
Line commands: V to view panel, S to edit STL, D to delete.   Press PF3 to end.

  Timestamp AID    Testcase Panel
_ 15554222         BUY1000                          Share Trading D
_ 15554222  ENTER  SELL900                          Share Trading D
_ 15554422         SELL900                          Share Trading D
_ 15554422  ENTER  SELL900                          Share Trading D
_ 15554623         SELL900                          Share Trading D
_ 15554623  ENTER  RTQUOTE                          Share Trading D
_ 15554824         RTQUOTE                          Share Trading D
_ 15554824  PF12   RTQUOTE                          Share Trading D
_ 15555024         RTQUOTE  Trader: Session Over    Share Trading D
_ 15555024  CLEAR  RTQUOTE
_ 15555225         RTQUOTE
_ 15555225  ENTER  LOGOFF   cesf logoff
_ 15555425         LOGOFF
_ 15555625         LOGON                              Signon to CIC
v 15555625  ENTER  LOGON                              Signon to CIC
_ 15555827         LOGON
_ 15555827  CLEAR  LOGON
PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-162 Capture screen selected for view

33. The user ID used this time is the second one from the UTBL used, as shown in Figure 28-163.

```
                          Signon to CICS                        APPLID CICSC22F

IBM'S  INTERNAL  SYSTEMS  MUST  ONLY  BE  USED  FOR  CONDUCTING
IBM'S BUSINESS OR FOR PURPOSES AUTHORIZED BY IBM MANAGEMENT



Type your userid and password, then press ENTER:

         Userid . . . . slitva2     Groupid . . . ________
         Password . . . ________
         Language . . . ___

     New Password . . . ________



DFHCE3520 Please type your userid.
F3=Exit                                                                 ENTER
```

Figure 28-163 Second captured trader application logon screen

28.9 WSim in batch mode

As we already mentioned, WTM can run in HIDE, DISPLAY and INTERACT mode. In the INTEACT mode WTM displays WSim panels and allows the user to manually change values, which gives the user more control. In this mode WTM can generate JCL for submitting for batch processing.

1. Enter the command **interact** from the WTM main panel to directly switch WTM into the INTERACT mode, as shown in Figure 28-164, or use any other way to navigate to the WTM Variables and Options panel and change the value of the Interface Access field.

```
                          WSim Test Manager

Select one of the following.  Then press Enter.

      Command    Action
_  1. CASE       Create and Process Testcases
   2. GROUP      Create and Process Testgroups
   3. CYCLE      Create and Process Testcycles
   4. RUN        Create WSim Networks and Schedule WSim Simulation Runs
   D. DOC        Create Test Documentation
   P. PROJECT    Add/Change Project or Alternate HLI
   U. UTIL       Run WSim Test Manager Utilities
   W. WII        Invoke WSim/ISPF Interface

Project: RESPROJ1                       Alternate HLI:



Command ===> interact_____________________________________________________
PF 1=Help       2=Split      3=End        4=           5=           6=
PF 7=Up         8=Down       9=Swap      10=Left      11=Right     12=Retrieve
```

Figure 28-164 Switching to INTERACT mode

2. To run a simulation, select option **4** or enter the command **run** on the WTM main panel as shown in Figure 28-165.

```
                          WSim Test Manager                        INTERACT set

Select one of the following.  Then press Enter.

      Command   Action
4  1. CASE      Create and Process Testcases
   2. GROUP     Create and Process Testgroups
   3. CYCLE     Create and Process Testcycles
   4. RUN       Create WSim Networks and Schedule WSim Simulation Runs
   D. DOC       Create Test Documentation
   P. PROJECT   Add/Change Project or Alternate HLI
   U. UTIL      Run WSim Test Manager Utilities
   W. WII       Invoke WSim/ISPF Interface

Project: RESPROJ1                      Alternate HLI:





Command ===> _____________________________________________________________
PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-165 Run simulation selected

3. Enter the line command **x** to run the RUNFULL schedule as presented in Figure 28-166. Press Enter.

```
                          Process WSim Schedules                   Row 1 of 2
Command==>
Enter a line command or add to create a new Schedule.        Press PF3 to end.
Line commands: D Delete, N Notes, O Output, P Preprocess, R Resources,
               S Testcases, U UTBLs, X Execute, W NTWRK.

  Name     Type Notes Description                 UTBLs   Last Run
x RUNFULL    V        run tradfull group             1  10/29/04 15:55
_ TSTTRAD1   V        test tsttrad testcase             10/28/04 13:55
****************************** Bottom of data ********************************




PF 1=Help      2=Split     3=End       4=          5=Add       6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-166 Running schedule RUNFULL

4. WTM provides an opportunity to specify names for log data sets. Accept the defaults or change names if preferred. A completion report is optional as shown in Figure 28-167. Press Enter.

```
                          Process WSim Schedules                    Row 1 of 2
 ------------------------- Specify WSim Log Names -------------------------
|
| Change the lognames and press Enter to continue or PF3 to end.
|
| ITPENTER: 'SLITVAK.WTMTEST.RESPROJ1.TLOG1.RUNFULL'     Mandatory
| Baseline: 'SLITVAK.WTMTEST.RESPROJ1.TLOG.RUNFULL'      Optional
|
| Completion Report (Y/N): Y
|
|
| PF 1=Help      2=Split     3=End       4=          5=          6=
| PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
 ---------------------------------------------------------------------------

PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-167 Simulation log names

5. The next panel is a WSim ISPF interface and not a WTM panel. It would not be displayed in HIDE mode. DISPLAY mode allows the user to see the panel and the INTERACT mode allows the user to work with it. Part 1 is shown in Figure 28-168. The batch run mode is specified by entering **b** in the corresponding field.

```
                          WSim: Run a Simulation

Type information.  Then press Enter.
                                                                    More:  +
  Input Data Sets
    Networks . . . . . . . 'SLITVAK.WTMTEST.RESPROJ1.NTWRKS'
                           ________________________________________________
    Message decks  . . . . 'SLITVAK.WTMTEST.RESPROJ1.MSGTXTS'
                           ________________________________________________

  Output Data Sets
    Log data . . . . . . . 'SLITVAK.WTMTEST.RESPROJ1.TLOG1.RUNFULL'
      Tape: Serial numbers ______ , ______
            File number    ____   (0-9999)
            Label type . . __     (NL or SL)
    Printer output . . . . 'SLITVAK.WSIMRUN.SYSPRINT'

  Control Options
    Run mode . . . . . . . b     (I=Interactive, B=Batch, N=Batch-no panel)

Command ===> ____________________________________________________________
F1=Help  F2=Split  F3=Exit   F4=Edit input         F5=Refresh  F6=Browse prt
F7=Bkwd  F8=Fwd    F9=Swap  F10=Additional input  F11=Save    F12=Cancel
```

Figure 28-168 Batch simulation selected

6. Part 2 (scroll down using PF8) is shown in Figure 28-169. Press Enter.

```
                              WSim: Run a Simulation

Type information.  Then press Enter.
                                                                        More: -
  Control Options (Continued)
    Network name to start  . . . . . . . . RUNFULL
      List in printer output . . . . . . . Y    (Y=Yes or N=No)
    Display monitor: VTAM APPL name  . . . WTMDM00
                     User password . . . . ________
    Number of buffers per log data set . . 5   (1-255)
    Save host-processor wait time stats    N    (Y=Yes or N=No)

Command ===> ____________________________________________________________
F1=Help  F2=Split  F3=Exit   F4=Edit input        F5=Refresh  F6=Browse prt
F7=Bkwd  F8=Fwd    F9=Swap  F10=Additional input  F11=Save    F12=Cancel
```

Figure 28-169 Second part of Run simulation panel

7. On the next panel, shown in Figure 28-170, the user can modify the JOB card as required and optionally specify the data set to save the generated JCL. This JCL can be later resubmitted without going through either WSim or WTM.

```
                              WSim: Submit a Batch Job

Type information.  Then press Enter.

  JCL JOB Statement
     //SLITVAKA JOB (ACCOUNT),'YOUR NAME',REGION=0M,CLASS=A,
     // MSGCLASS=X,MSGLEVEL=(1,1),NOTIFY=&SYSUID
     //*
     //*

  JCL OUTPUT Statement(s)
     ____________________________________________________________________
     ____________________________________________________________________
     ____________________________________________________________________
     ____________________________________________________________________

  Data set to save JCL in 'SLITVAK.general.jcl(wsimjcl)'

  Select one or more message routing codes by typing a '/' or 'S'.
     _ 1    _ 2    _ 3    _ 4    _ 5    _ 6    _ 7    / 8
     _ 9    _ 10   _ 11   _ 12   _ 13   _ 14   _ 15   _ 16

F1=Help  F2=Split  F5=Refresh  F9=Swap  F11=Save  F12=Cancel
```

Figure 28-170 JOB card for batch simulation

8. The batch job is submitted and WSim displays the message shown in Figure 28-171.

```
                              WSim: Run a Simulation

Type information.  Then press Enter.
                                                                      More:  +
  Input Data Sets
    Networks . . . . . . . 'SLITVAK.WTMTEST.RESPROJ1.NTWRKS'
                           ____________________________________________
    Message decks  . . . . 'SLITVAK.WTMTEST.RESPROJ1.MSGTXTS'
                           ____________________________________________

  Output Data Sets
    Log data . . . . . . . 'SLITVAK.WTMTEST.RESPROJ1.TLOG1.RUNFULL'
      Tape: Serial numbers ______ , ______
            File number    ____  (0-9999)
            Label type . . __    (NL or SL)
    Printer output . . . . 'SLITVAK.WSIMRUN.SYSPRINT'

  Control Options
    Run mode . . . . . . . B     (I=Interactive, B=Batch, N=Batch-no panel)

Command ===>      | Batch job SLITVAKA(JOB00521) submitted. |
F1=Help  F2=Spli                                              F6=Browse prt
F7=Bkwd  F8=Fwd    F9=Swap  F10=Additional input  F11=Save    F12=Cancel
```

Figure 28-171 Job is submitted

9. The job runs successfully as shown in Figure 28-172.

```
15.07.03 JOB00521 $HASP165 SLITVAKA ENDED AT STLABF7  MAXCC=0 CN(INTERNAL)
*** _
```

Figure 28-172 Successful run

The user can look at the output in the system job log or use the online WTM panels to review reports.

The generated JCL is presented in Figure 28-173. It can be reused as needed.

```
  Menu  Utilities  Compilers  Help
 ------------------------------------------------------------------------------
 BROWSE    SLITVAK.GENERAL.JCL(WSIMJCL)                Line 00000000 Col 001 080
 Command ===> ____________________________________________________ Scroll ===> CSR
********************************** Top of Data ***********************************
//SLITVAKA JOB (ACCOUNT),'YOUR NAME',REGION=0M,CLASS=A,
// MSGCLASS=X,MSGLEVEL=(1,1),NOTIFY=&SYSUID
//*
//*
//RUNWSIM  EXEC PGM=ITPENTER
//STEPLIB  DD DSN=ADTOOLS.WORKLOAD.SIM11.SITPLOAD,DISP=SHR
//PARMDD   DD *
PRTLNCNT=60
DMAPPL=WTMDM00
NCP=5
NTWRKL=RUNFULL
ROUTCDE=(8)
//SYSPRINT DD DSN=SLITVAK.WSIMRUN.SYSPRINT,DISP=SHR
//INITDD   DD DSN=SLITVAK.WTMTEST.RESPROJ1.NTWRKS,DISP=SHR
//MSGDD    DD DSN=SLITVAK.WTMTEST.RESPROJ1.MSGTXTS,DISP=SHR
//LOGDD    DD DISP=SHR,
//            DSN=SLITVAK.WTMTEST.RESPROJ1.TLOG1.RUNFULL
******************************** Bottom of Data ******************************
```

Figure 28-173 Generated JCL

28.10 Regression test

When preparing for the simulation batch run, since it was not the first time we were running the RUNFULL simulation, we specified names for new and baseline simulation logs on the WTM panel used to specify the names of the log data sets, as shown in Figure 28-167 on page 767.

We now compare the results of these two runs and see if there are any differences.

1. Enter the line command **0** to work with the output from the RUNFULL simulation as shown in Figure 28-174.

```
                          Process WSim Schedules                    Row 1 of 2
Command==>
Enter a line command or add to create a new Schedule.        Press PF3 to end.
Line commands: D Delete, N Notes, O Output, P Preprocess, R Resources,
               S Testcases, U UTBLs, X Execute, W NTWRK.

  Name     Type Notes Description                 UTBLs   Last Run
o RUNFULL    V        run tradfull group             1  11/02/04 14:38
_ TSTTRAD1   V        test tsttrad testcase             10/28/04 13:55
****************************** Bottom of data ******************************

PF 1=Help      2=Split     3=End       4=          5=Add       6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-174 Working with schedule RUNFULL output

2. On the WTM Reports panel select option **8** to view the screen compare report, as shown in Figure 28-175. Press Enter.

```
                          Reports for schedule RUNFULL

Select one of the following.  Then press Enter.

      Command   Action                                                More:  +
8  1. TLOG      View loglist report from the latest WSim run
   2. TRESP     View response time report from the latest WSim run
   3. TDM       Log display monitor for the latest WSim run
   4. TSP       View SYSPRINT from the latest WSim run

   5. MLOG      View baseline loglist report
   6. MRESP     View baseline response time report
   7. MDM       Log display monitor for the baseline log

   8. COMP      View screen compare report
   9. CDM       Log display comparator
  10. RTCOMP    Edit response time compare report

  11. COMPREP   Edit completion reports


Command===> _____________________________________________________________
PF 1=Help       2=Split     3=End       4=          5=          6=
PF 7=           8=Down      9=Swap      10=Left     11=Right    12=Retrieve
```

Figure 28-175 Viewing screen compare report

3. WTM displays the panel with the names of the log data sets to be compared and where to put the results, as shown in Figure 28-176. Press Enter.

```
                       WSim: Compare Logged Display Data

Type information.  Then press Enter.
                                                                        More:  +
  Input Data Sets
    Master log data  . . . 'SLITVAK.WTMTEST.RESPROJ1.TLOG.RUNFULL'
      Tape: Serial numbers ______ , ______
            File number    ____   (0-9999)
            Label type . . __     (NL or SL)
    Test log data  . . . . 'SLITVAK.WTMTEST.RESPROJ1.TLOG1.RUNFULL'
      Tape: Serial numbers ______ , ______
            File number    ____   (0-9999)
            Label type . . __     (NL or SL)
    Control commands . . . 'SLITVAK.WTMUSER.SKELS(ITMSCCOM)'

  Output Data Set
    Printer output . . . . 'SLITVAK.WTM.SYSPRINT'


 5655-I39 (C)  Copyright IBM Corporation 1976, 2002.  All rights reserved.

F9=Swap  F10=Edit ctl  F11=Save  F12=Cancel
```

Figure 28-176 Log data sets to compare

4. WTM reports that a difference was found for at least one resource as shown in Figure 28-177.

```
                        WSim: Compare Logged Display Data

Type information.  Then press Enter.
                                                                    More:  +
  Input Data Sets
    Master log data  . . . 'SLITVAK.WTMTEST.RESPROJ1.TLOG.RUNFULL'
      Tape: Serial numbers ______ , ______
            File number    ____  (0-9999)
            Label type . . __    (NL or SL)
    Test log data  . . . . 'SLITVAK.WTMTEST.RESPROJ1.TLOG1.RUNFULL'
      Tape: Serial numbers ______ , ______
            File number    ____  (0-9999)
            Label type . . __    (NL or SL)
    Control commands . . . 'SLITVAK.WTMUSER.SKELS(ITMSCCOM)'

  Output Data Set
    Printer output . . . . 'SLITVAK.WTM.SYSPRINT'


 Comparison completed.  A difference was found for at least one resource.

F9=Swap  F10=Edit ctl  F11=Save  F12=Cancel
```

Figure 28-177 Differences found

5. To see it press PF6. Scroll down to see lines 148 and 149; they contain information about the master and the test runs, as shown in Figure 28-178.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------
VIEW       SLITVAK.WTM.SYSPRINT                          Columns 00001 00072
Command ===> ____________________________________________ Scroll ===> CSR
000147 1WSim COMPARE UTILITY OUTPUT
000148  MASTER RUN  TIME 15.55.21,   OCTOBER 29, 2004  VERSION  1  RELEASE  1.0
000149  TEST   RUN  TIME 15.05.59,  NOVEMBER  2, 2004  VERSION  1  RELEASE  1.0
000150 0-----------------------------------------------------------------------
000151                                                   Complete Records List
000152  Master:  NETWORK     RUNFULL
000153           VTAMAPPL    VAPPL00
000154           DEV/LU      WSIMLU00-00001
000155  ----------------------------------------------------------------------
000156 0  MASTER Records
000157    --------------
000158 0    Sequence
000159       Number       MSGTXT         Usage         Reason
000160      --------     ----------     ----------     -------------------------
000161          0        LOGON          Used
000162          1        LOGON          Used
000163          2        LOGON          Used
000164          3        LOGON          Used
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 28-178 Master and test run information

6. The master (baseline) screens (the corresponding MSGTXT programs) are listed starting from line 156, as shown in Figure 28-179.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------------
VIEW       SLITVAK.WTM.SYSPRINT                              Columns 00001 00072
Command ===> ____________________________________________ Scroll ===> CSR
000156 0  MASTER Records
000157    --------------
000158 0    Sequence
000159       Number        MSGTXT          Usage          Reason
000160      --------     ----------      ----------     -------------------------
000161            0       LOGON           Used
000162            1       LOGON           Used
000163            2       LOGON           Used
000164            3       LOGON           Used
000165            5       LOGON           Used
000166            6       INITST7         Used
000167            7       INITST7         Used
000168            8       INITST7         Used
000169            9       INITST7         Used
000170           10       TRADLOG         Used
000171           11       TRADLOG         Used
000172           12       SELCOMP         Used
000173           13       SELCOMP         Used
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 28-179 Master run records listed

7. Scroll down to line 230; starting from this line the screens used in the test run are listed, as shown in Figure 28-180.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------------
VIEW       SLITVAK.WTM.SYSPRINT                              Columns 00001 00072
Command ===> ____________________________________________ Scroll ===> CSR
000230 -  TEST Records
000231    ------------
000232 0    Sequence
000233       Number        MSGTXT          Usage          Reason
000234      --------     ----------      ----------     -------------------------
000235            0       LOGON           Used
000236            1       LOGON           Used
000237            2       LOGON           Used
000238            3       LOGON           Used
000239            5       LOGON           Used
000240            6       INITST7         Used
000241            7       INITST7         Used
000242            8       INITST7         Used
000243            9       INITST7         Used
000244           10       TRADLOG         Used
000245           11       TRADLOG         Used
000246           12       SELCOMP         Used
000247           13       SELCOMP         Used
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 28-180 Test run records listed

8. Continue scrolling down. There are differences reported (for more clarity, we passed some differences found earlier in the report) as shown in Figure 28-181.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------------
VIEW       SLITVAK.WTM.SYSPRINT                              Columns 00001 00072
Command ===> ____________________________________________ Scroll ===> CSR
000367  Master:  NETWORK     RUNFULL
000368           VTAMAPPL    VAPPL00
000369           DEV/LU      WSIMLU00-00001
000370  ---------------------------------------------------------------------
000371 0    MASTER            TEST                                ALL
000372  Sequence Number   Sequence Number   Checkonly  Mask   Mask    Result
000373  ---------------   ---------------   ---------  -----  ----   ---------
000374          49                49                                    Equal
000375          50                50                                    Equal
000376          51                51                                    Equal
000377          52                52                                  Not Equal
000378          53                53                                  Not Equal
000379          54                54                                  Not Equal
000380          56                56                                    Equal
000381          57                57                                    Equal
000382          58                58                                    Equal
000383 1WSim COMPARE UTILITY OUTPUT
000384  MASTER RUN  TIME 15.55.21,   OCTOBER 29, 2004  VERSION  1  RELEASE  1.0
 F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
 F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 28-181 Differences are reported

9. The captured screens, where differences were found, are also presented at the end of this report.

The log display comparator report

Use the following steps to see another type of WTM report – the Log display comparator.

1. Return to the WTM reports panel and select option **9**, as shown in Figure 28-182.

```
                        Reports for schedule RUNFULL

Select one of the following.  Then press Enter.

     Command   Action                                                More:  +
9_  1. TLOG      View loglist report from the latest WSim run
   2. TRESP     View response time report from the latest WSim run
   3. TDM       Log display monitor for the latest WSim run
   4. TSP       View SYSPRINT from the latest WSim run

   5. MLOG      View baseline loglist report
   6. MRESP     View baseline response time report
   7. MDM       Log display monitor for the baseline log

   8. COMP      View screen compare report
   9. CDM       Log display comparator
  10. RTCOMP    Edit response time compare report

  11. COMPREP   Edit completion reports


Command===> ____________________________________________________________
PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=          8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-182 Log display comparator selected

2. On the Panel Display Options select to view the index of screen images as shown in Figure 28-183.

```
                          Reports for schedule RUNFULL
 ------------------------------ Panel Display Options ------------------------------
|
| Press ENTER to continue or PF3 to cancel.
|
| Index or View  (I/V): i
|
| For View only:
|   Logged or Fixed Delay (L/F): F
|   Fixed Delay Value in Seconds: 1
|   Display Panels    :    1 to 59
|
|
|
| PF 1=Help      2=Split     3=End       4=          5=          6=
| PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
 ----------------------------------------------------------------------------

  11. COMPREP   Edit completion reports


Command===>
PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-183 Report options

3. The next panel shows the index of the screens captured from the test run with the detected differences flagged (six in total). Enter the line command **v** in front of one of the screen images with differences and press Enter as shown in Figure 28-184.

```
                          Resource Display Index                Row 17 of 59
Command ==>
Enter line command                        Number of screen differences:     6
Line commands: V to view panel, S to edit STL, D to delete.   Press PF3 to end.

  Timestamp AID    Testcase D Panel
_ 15554019  ENTER  BUY1000                              Share Trading
_ 15554222         BUY1000                              Share Trading
_ 15554222  ENTER  SELL900                              Share Trading
_ 15554422         SELL900                              Share Trading
_ 15554422  ENTER  SELL900                              Share Trading
_ 15554623         SELL900                              Share Trading
_ 15554623  ENTER  RTQUOTE                              Share Trading
v 15554824         RTQUOTE  * Data Difference Detected
_ 15554824  PF12   RTQUOTE  * Data Difference Detected
_ 15555024         RTQUOTE  * Data Difference Detected
_ 15555024  CLEAR  RTQUOTE
_ 15555225         RTQUOTE
_ 15555225  ENTER  LOGOFF     cesf logoff
_ 15555425         LOGOFF
_ 15555625         LOGON                                 Signon to C
_ 15555625  ENTER  LOGON                                 Signon to C
PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-184 Resource display index with differences reported

4. Press Enter to see how WTM “flashes” the two screen images one over another to highlight the differences.

28.11 Display Monitor Facility

Display Monitor Facility is a VTAM application program within WSim which can be used to:

- Display simulated 3270 screens
- Display transmitted and received data flows for any simulated device

It is activated whenever WSim simulation is running.

Display Monitor Facility can be used to develop and debug scripts for display devices, to dynamically monitor tests when they are running, and to show interactions with host applications.

To use the Display Monitor Facility, the user needs to know the name of the Display Monitor VTAM APPL. The WSim installer should provide this name, which also should have been specified on the WTM Variables panel (see Figure 28-185). To get there, enter the command **vars** on the WTM main panel.

```
                  Specify WSim Test Manager Variables and Options

Update the fields, then press Enter to save the values.   Press PF3 to end.

WSim/ISPF Interface Access: INTERACT  (hide, display or interact)
Panel Message Delay  : 2  seconds     (1-10)
Log Display Monitor Chars: {}`        Automatic REFRESH?       : N  (Y/N)
Display Panel ID?    : N  (Y/N)       Display Function Keys?   : Y  (Y/N)

WSim Load Library    : ADTOOLS.WORKLOAD.SIM11.SITPLOAD
IDC VTAM APPL name   : ITPIDC0        Display Monitor VTAM APPL name: WTMDM00
Fully Validate WSim Data Set Names? : Y  (Y/N)

Work Data Sets HLI   : SLITVAK
Low Level Names:  STL: STL            MSGTXTs: MSGTXTS      NTWRKS: NTWRKS

VTAM Name Model      : VAPPL##
WSim Name Model      : WSIMLU##
Numeric substitution start value: 00


PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-185 WTM Variables panel

In our case, the Display Monitor VTAM APLL name is WTMDM00. This is the name the user has to log on to in order to use the facility.

Enter the command **run** from the WTM main panel or use option **4** to start working with WTM schedules.

Enter the line command **x** to start the simulation for the schedule TSTTRAD1. When it is started, have another terminal session open to connect to the Display Monitor VTAM APPL WTMDM00. The command shown in Figure 28-186 worked on the demonstration system we used. Consult with your system programmer for details in your specific environment.

```
NETMON         WELCOME TO THE ==>STLABF6 <= NATIVE TEST INTERACTIVE NETWORK
                 THIS TERMINAL IS ATTACHED TO SYSTEM STLABF6
                       AND THE TERMINAL ID IS TCP00008
                      11:21:09 WEDNESDAY, NOV.  3, 2004

Listed below are highlighted commands that pass your terminal through the
network to the desired system.  Most commands may be followed by userid.

STLABF6  - Logon to MVS/TSO (also use  'L userid' or 'LOGON userid' )

 C31F     to Logon to C31F
 C22F     to Logon to C22F
 IM8F     to Logon to IM8F
 IM9F     to Logon to IM9F
 WT00     to Logon to WTMDM00




               NATIVE ROOM 3-4578      PF1=HELP      PF3=EXIT
          ALL COMMANDS AND DATA MUST BE ENTERED ON THE BOTTOM LINE

===> wt00_
```

Figure 28-186 Logon to WTMDM00

As the simulation continues, the messages on the simulation console show the used network resource names, which were generated by WTM using the naming schema defined on the WTM Variables panel (in our case it is WSIMLU##). The user can also see the list of the resources for a schedule by entering the line command `r` on the WTM Process schedules panel. The schedule TSTTRAD1 is using the WSim name WSIMLU00.

On the Display Monitor Facility control panel shown on Figure 28-187 enter the name of the simulated resource (terminal) to be monitored (WSIMLU00) and press Enter.

```
             WSim Version 1 Release 1.0.0 Display Monitor Facility

Name   = wsimlu00                WSim name of simulated device or 3270 display
View   = SCREEN         DATA or SCREEN - show data stream or 3270 screen image

Screen image display only:
Update = XMITRECV       Monitoring display updated when:
                          MONITOR  - MONITOR statement is executed from script,
                          TIMER    - the specified time value expires, or
                          XMITRECV - data is transmitted/received by display.
Source = BLOCKS         Data stream sent to the monitoring display built from:
                          BLOCKS - WSim internal control blocks
                          DATA   - data transmitted/received by display.
Timer  = 10             1-600 Seconds when Update = TIMER
Aid    = ON             ON, OFF, or (row,column) location of AID display field

Data stream display only:
Lines  = 2              Maximum number of displayed data lines
Code   = EBCDIC         ASCII or EBCDIC - interpret data as ASCII or EBCDIC


ENTER    - Submits parameters to start monitoring of simulated display.
PA1/ATTN - Stops monitoring of simulated display.
PF3/PF15 - Ends Display Monitor Facility session.
```

Figure 28-187 Display Monitor Facility control panel

The traffic on the simulated terminal will appear on the Display Monitor Facility screen. We will not present these screens in this book, as they are just a replay of the captured screens.

Press PF3 to exit or use PA1 to interrupt and go back to the Display Monitor Facility control panel to change any options.

28.12 Load test

Load testing of applications requires running multiple simulated terminals concurrently and that each of the terminals is used by a different user. Use the following steps to perform load testing.

1. Enter the command **run** from the WTM main panel and then enter the command **add** on the next WTM panel to start working with the new schedule RUNLOAD, as shown in Figure 28-188.

```
                          Process WSim Schedules                    Row 1 of 2
                            Add New Test Schedule

 Type Schedule Name, Description and Schedule Type.  Then press Enter.

 Schedule Name: runload
 Description  : load test  transaction
 Schedule Type: v     (V=VTAMAPPL, T=TCP/IP, C=CPI-C)



 PF 1=Help      2=Split     3=End       4=          5=          6=
 PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve



PF 1=Help      2=Split     3=End       4=          5=          6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-188 New schedule RUNLOAD

2. Press Enter. WTM displays a Milestones panel with information about the steps to be performed in order to create a new schedule as shown in Figure 28-189.

```
                          Process WSim Schedules                        Row 1 of 2
mman ┌──────────────────────────────── "Milestones" ────────────────────────────┐
ter  │_                                                                          │ to end.
ne c │                                                                           │
     │ These are the steps to create a new WSim Schedule.                        │
     │ Press Enter to continue or PF3 to cancel.                                 │
Name │                                                                           │
RUNF │ Step 1 - Define the simulated resources and scripts.                      │
TSTT │ Step 2 - Generate a WSim network (NTWRK).                                 │
**** │ Step 3 - Define completion report thresholds.                             │ ********
     │                                                                           │
     └───────────────────────────────────────────────────────────────────────────┘
```

Figure 28-189 “Milestones” panel shows steps to be performed

3. The WTM Resource list panel for the schedule RUNLOAD is displayed, as shown in Figure 28-190 with the assigned VTAM and WSim names of the network resources. Enter the line command **s** to continue, press Enter.

```
                      Resource List for Schedule RUNLOAD            Row 1 of 1
Command==>
Enter command sort to sort by VTAMname.                     Press PF3 to end.
Line commands: S Select, I Insert, R Repeat, D Delete.

  VTAMname WSIMname  Test...  Name       Description
s VAPPL00  WSIMLU00
****************************** Bottom of data ********************************



PF 1=Help      2=Split     3=End       4=          5=Sort      6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-190 Simulated resources list for schedule RUNLOAD

4. In this simulation example, the test group TRADFULL will have to be executed twice, as shown in Figure 28-191. The user can schedule any combination of test items to be executed multiple times in any compatible sequence. Press PF3 to continue

```
Specify Order for Resource WSIMLU00 under VAPPL00
                                                                Row 1 of 10
Command==>
Enter order or summary.          Press PF3 to end.

Order                  Name       Test...  Type  Description
                       BUY1000    Case       V   buy 1000 shares
                       INITST7    Case       V   start test trans, opt 7
                       LOGOFF     Case       V   logoff from CICS
                       LOGON      Case       V   logon to cicsc22f
                       RTQUOTE    Case       V   real time quote
                       SELCOMP    Case       V   select company
                       SELL900    Case       V   selling 900 shares
                       TRADLOG    Case       V   logon to trader appl
                       TSTTRAD    Case       V   Logon,trader,logon,getquot
1 2_                   TRADFULL   Group      V   full session with trader a
****************************** Bottom of data ******************************

PF 1=Help      2=Split     3=End       4=          5=          6=Summary
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-191 Testgroup TRADFULL to be executed twice

5. WTM returns to the Resources list panel and shows the test group TRADFULL being executed twice for this simulated terminal. To add an additional terminal use the line command **r** to repeat the terminal definition as shown in Figure 28-192 and press Enter.

```
                    Resource List for Schedule RUNLOAD          Row 1 of 2
Command==>
Enter command sort to sort by VTAMname.                   Press PF3 to end.
Line commands: S Select, I Insert, R Repeat, D Delete.

  VTAMname WSIMname  Test...  Name       Description
r VAPPL00  WSIMLU00  GROUP    TRADFULL   FULL SESSION WITH TRADER APPLICATION
     "        "      GROUP    TRADFULL   FULL SESSION WITH TRADER APPLICATION
****************************** Bottom of data ******************************

PF 1=Help      2=Split     3=End       4=          5=Sort      6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-192 New simulated resource to be added

6. The number of the resources which can be simulated is limited; the maximum is defined by the product installer in the WTM VTAM application major node. For this simulation, leave the default value of one and press Enter as shown in Figure 28-193.

```
                                  Repeat Request

 Number to repeat: 1    (0-999)

 To add new VTAM names as required to satisfy the repeat request, press Enter
 or PF5.  To repeat WSIM names within the current VTAM name, press PF6.

 Press PF3 to cancel the repeat request.



 PF 1=Help      2=Split     3=End       4=          5=VTAM      6=WSIM
 PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve

PF 1=Help      2=Split     3=End       4=          5=VTAM      6=WSIM
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-193 Adding new VTAM resource

7. The schedule RUNLOAD has two simulated network resources (terminals) defined now, as shown in Figure 28-194. These terminals will run simultaneously.

```
                    Resource List for Schedule RUNLOAD            Row 1 of 4
Command==>
Enter command sort to sort by VTAMname.                     Press PF3 to end.
Line commands: S Select, I Insert, R Repeat, D Delete.

  VTAMname WSIMname  Test...  Name       Description
_ VAPPL00  WSIMLU00  GROUP    TRADFULL   FULL SESSION WITH TRADER APPLICATION
    "        "       GROUP    TRADFULL   FULL SESSION WITH TRADER APPLICATION
_ VAPPL01  WSIMLU01  GROUP    TRADFULL   FULL SESSION WITH TRADER APPLICATION
    "        "       GROUP    TRADFULL   FULL SESSION WITH TRADER APPLICATION
****************************** Bottom of data ******************************



PF 1=Help      2=Split     3=End       4=          5=Sort      6=
PF 7=Up        8=Down      9=Swap     10=Left     11=Right    12=Retrieve
```

Figure 28-194 Two simulated resources (displays) defined for RUNLOAD

8. Press PF3. Since this is the first time, WTM will proceed to display the Completion Report thresholds panel as shown in Figure 28-195.

```
                 Completion Report Response Time Thresholds
Command==>
Update this panel, then press Enter.  Press PF3 to end.

Response Time:                                Percentiles:
         Mean   < 0.05  seconds                    95  % < 0.1   seconds
         Median < _____   "                       90  % < _____   "
         Mode   < _____   "                       80  % < _____   "
         High   < _____   "                       70  % < _____   "

PF 1=Help       2=Split      3=End        4=           5=           6=
PF 7=Up         8=Down       9=Swap      10=Left      11=Right     12=Retrieve
```

Figure 28-195 Time thresholds defined

9. Change the threshold values if required, press Enter to save them, and finally, press PF3 to return to the WTM Schedules panel. The new schedule RUNFULL is ready to be executed. To do some modifications to the script, enter the line command **W** and press Enter as shown in Figure 28-196.

```
                           Process WSim Schedules                    Row 1 of 3
Command==>
Enter a line command or add to create a new Schedule.        Press PF3 to end.
Line commands: D Delete, N Notes, O Output, P Preprocess, R Resources,
               S Testcases, U UTBLs, X Execute, W NTWRK.

  Name      Type Notes Description                 UTBLs   Last Run
W RUNLOAD    V         load test  transaction        1                  Added
_ RUNFULL    V         run tradfull group            1  11/02/04 14:38
_ TSTTRAD1   V         test tsttrad testcase            11/03/04 11:43
****************************** Bottom of data ********************************

PF 1=Help       2=Split      3=End        4=           5=Add        6=
PF 7=Up         8=Down       9=Swap      10=Left      11=Right     12=Retrieve
```

Figure 28-196 Work with network definition for RUNLOAD requested

10. The network definition for the schedule RUNLOAD opens for editing in an ISPF session. Scroll down to see the parameter UTI = 100, on line 19, as presented in Figure 28-197. This parameter defines the user time interval measured in 0.01 sec. This is a delay between user messages, a “think time”. The value of 100 is equivalent to the “think” time of 1 sec.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       SLITVAK.WTMTEST.RESPROJ1.NTWRKS(RUNLOAD) - 01.0 Columns 00001 00072
Command ===> ____________________________________________ Scroll ===> CSR
000015                 OPTIONS=(CONRATE,DEBUG,MONCMND),
000016                 RSTATS=YES,
000017                 STLTRACE=YES,
000018                 THKTIME=UNLOCK,
000019                 UTI=100
000020 *         **User Tables**
000021 101       UTBL  CICSUIS    ** user ids and passwords for CICS
000022 *
000023 *         Paths
000024 WTMEND    PATH  WTMEND
000025 *
000026 *         TestGroup TRADFULL            ¬¬¬ DO NOT CHANGE THIS COMMENT
000027 1         PATH  LOGON,INITST7,TRADLOG,SELCOMP,BUY1000,
000028                 SELL900,RTQUOTE,LOGOFF
000029 *
000030 *         Simulated Resources
000031 VAPPL00   VTAMAPPL
000032 WSIMLU00  LU   PATH=(1,1,WTMEND)
000033 VAPPL01   VTAMAPPL
000034 WSIMLU01  LU   PATH=(1,1,WTMEND)
```

Figure 28-197 Network definition for RUNLOAD

11. If the value of the UTI parameter is changed, all the user messages will be affected, and the intermessage interval will be set to the new value but will remain constant. We leave the value of the UTI to be equal 100 (1 sec), but define a random delay for each simulated interval, as shown in Figure 28-198.

    DELAY=R(n,m) results in a random intermessage delay in the range from n*UTI to m*UTI.

    The values entered result in the random delays from 5 to 50 sec for the display WSIMLU00 and from 1 to 30 sec for the display WSIMLU01. Press PF3 to exit the editing session and save the changes.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
EDIT       SLITVAK.WTMTEST.RESPROJ1.NTWRKS(RUNLOAD) - 01.0 Columns 00001 00072
Command ===> ____________________________________________ Scroll ===> CSR
000015                 OPTIONS=(CONRATE,DEBUG,MONCMND),
000016                 RSTATS=YES,
000017                 STLTRACE=YES,
000018                 THKTIME=UNLOCK,
000019                 UTI=100
000020 *         **User Tables**
000021 101       UTBL  CICSUIS    ** user ids and passwords for CICS
000022 *
000023 *         Paths
000024 WTMEND    PATH  WTMEND
000025 *
000026 *         TestGroup TRADFULL            ¬¬¬ DO NOT CHANGE THIS COMMENT
000027 1         PATH  LOGON,INITST7,TRADLOG,SELCOMP,BUY1000,
000028                 SELL900,RTQUOTE,LOGOFF
000029 *
000030 *         Simulated Resources
000031 VAPPL00   VTAMAPPL
000032 WSIMLU00  LU   PATH=(1,1,WTMEND),delay=r(5,50)
000033 VAPPL01   VTAMAPPL
000034 WSIMLU01  LU   PATH=(1,1,WTMEND),delay=r(1,30)
```

Figure 28-198 Random delays for simulated resources added

12. It is a good idea to ensure that manual changes to a network definition did not introduce any errors, so enter the line command **p** to preprocess the schedule RUNLOAD. WTM will

process the network definition statements, the STL programs and the message generation decks (MSGTXTs) and report problems, if any, as shown in Figure 28-199.

```
                          Process WSim Schedules                    Row 1 of 3
Command==>
Enter a line command or add to create a new Schedule.        Press PF3 to end.
Line commands: D Delete, N Notes, O Output, P Preprocess, R Resources,
               S Testcases, U UTBLs, X Execute, W NTWRK.

  Name     Type Notes Description                UTBLs   Last Run
p RUNLOAD    V        load test  transaction        1                  NTWRK
_ RUNFULL    V        run tradfull group            1  11/02/04 14:38
_ TSTTRAD1   V        test tsttrad testcase            11/03/04 11:43
****************************** Bottom of data ******************************
```

Figure 28-199 Preprocessing (no simulation run) for RUNLOAD requested

13. As shown in Figure 28-200, WTM did not find any coding problems, so preprocess is successfully completed. You can browse the output by pressing PF6.

```
                  WSim: Preprocess Networks and Message Decks

Type information.  Then press Enter.
                                                                      More:  +
  Input Data Set
    Networks and message decks 'SLITVAK.WTMTEST.RESPROJ1.NTWRKS'
      Member name  . . . . . . RUNLOAD

  Output Data Sets
    Networks . . . . . . . . . 'SLITVAK.WTMTEST.RESPROJ1.NTWRKS'
    Message decks  . . . . . . 'SLITVAK.WTMTEST.RESPROJ1.MSGTXTS'
    Printer output . . . . . . 'SLITVAK.WTM.SYSPRINT'

  Control Options
    Run mode . . . . . . . . . I  (I=Interactive, B=Batch, N=Batch-no panel)
    Check syntax . . . . . . . Y  (Y=Yes or N=No)
      Replace existing members N  (Y=Yes or N=No)
      Network listing  . . . . Y  (Y=Yes or N=No)
        Cross reference report Y  (Y=Yes or N=No)
      Network summary report   N  (Y=Yes or N=No)

Command ===>        | Preprocess successfully completed. |
F1=Help  F2=Split                                          esh  F6=Browse prt
F7=Bkwd  F8=Fwd    F9=Swap  F10=Additional input  F11=Save    F12=Cancel
```

Figure 28-200 Prepocessing fro RUNLOAD is successful

14. Press PF3; on the next panel enter the line command **x** and press Enter to start the simulation, as shown in Figure 28-201.

```
                          Process WSim Schedules                    Row 1 of 3
Command==>
Enter a line command or add to create a new Schedule.        Press PF3 to end.
Line commands: D Delete, N Notes, O Output, P Preprocess, R Resources,
               S Testcases, U UTBLs, X Execute, W NTWRK.

  Name     Type Notes Description                UTBLs   Last Run
x RUNLOAD    V        load test  transaction        1                  Preproce
_ RUNFULL    V        run tradfull group            1  11/02/04 14:38
_ TSTTRAD1   V        test tsttrad testcase            11/03/04 11:43
****************************** Bottom of data ********************************
```

Figure 28-201 Simulation run for RUNLOAD requested

15. As usually, WTM will present an opportunity to change the names for the log data sets and make a choice whether to create the optional completion report as shown in Figure 28-202. Make any required changes and press Enter.

```
                          Process WSim Schedules                    Row 1 of 3
                          Specify WSim Log Names

 Change the lognames and press Enter to continue or PF3 to end.

 ITPENTER: 'SLITVAK.WTMTEST.RESPROJ1.MLOG.RUNLOAD'      Mandatory
 Baseline: ____________________________________________ Optional

 Completion Report (Y/N): Y
```

Figure 28-202 Log data set for RUNLOAD simulation run defined

16. WTM starts the simulation RUNLOAD. The WSim run messages are presented in two parts shown in Figure 28-203 and Figure 28-204.

```
ITP016I Workload Simulator (WSim) Version 1, Release 1.0, May 23, 2002, 11:05

ITP003I WSim INITIALIZATION COMPLETE
ITP200I DISPLAY MONITOR FACILITY ACTIVE USING APPL WTMDM00
ITP029I INITIALIZATION COMPLETE FOR NETWORK RUNLOAD
ITP006I NETWORK RUNLOAD STARTED
ITP137I RUNLOAD  WSIMLU01-00001 - WTM: LOGON Finished
ITP077I NETWORK RUNLOAD MESSAGE RATES 17 RECEIVED, 7 SENT
ITP137I RUNLOAD  WSIMLU01-00001 - WTM: INITST7 Finished
ITP137I RUNLOAD  WSIMLU00-00001 - WTM: LOGON Finished
ITP137I RUNLOAD  WSIMLU01-00001 - WTM: TRADLOG Finished
ITP077I NETWORK RUNLOAD MESSAGE RATES 6 RECEIVED, 6 SENT
ITP137I RUNLOAD  WSIMLU01-00001 - WTM: SELCOMP Finished
ITP137I RUNLOAD  WSIMLU00-00001 - WTM: INITST7 Finished
ITP137I RUNLOAD  WSIMLU01-00001 - WTM: BUY1000 Finished
ITP077I NETWORK RUNLOAD MESSAGE RATES 5 RECEIVED, 5 SENT
ITP137I RUNLOAD  WSIMLU00-00001 - WTM: TRADLOG Finished
ITP137I RUNLOAD  WSIMLU01-00001 - WTM: SELL900 Finished
ITP137I RUNLOAD  WSIMLU00-00001 - WTM: SELCOMP Finished
ITP077I NETWORK RUNLOAD MESSAGE RATES 5 RECEIVED, 5 SENT
ITP137I RUNLOAD  WSIMLU01-00001 - WTM: RTQUOTE Finished
ITP137I RUNLOAD  WSIMLU01-00001 - WTM: LOGOFF Finished
ITP137I RUNLOAD  WSIMLU00-00001 - WTM: BUY1000 Finished
*** _
```

Figure 28-203 RUNLOAD simulation run messages (Part 1 of 2)

```
ITP077I NETWORK RUNLOAD MESSAGE RATES 13 RECEIVED, 7 SENT
ITP137I RUNLOAD  WSIMLU01-00001 - Attempt to read beyond end of CICSU
ITP137I RUNLOAD  WSIMLU01-00001 - WTMEND: 1 out of 2 finished.
ITP077I NETWORK RUNLOAD MESSAGE RATES 1 RECEIVED, 1 SENT
ITP137I RUNLOAD  WSIMLU00-00001 - WTM: SELL900 Finished
ITP077I NETWORK RUNLOAD MESSAGE RATES 2 RECEIVED, 2 SENT
ITP137I RUNLOAD  WSIMLU00-00001 - WTM: RTQUOTE Finished
ITP137I RUNLOAD  WSIMLU00-00001 - WTM: LOGOFF Finished
ITP137I RUNLOAD  WSIMLU00-00001 - Attempt to read beyond end of CICSU
ITP137I RUNLOAD  WSIMLU00-00001 - WTMEND: 2 out of 2 finished.
ITP137I RUNLOAD  WSIMLU00-00001 - WTMEND: All scripts run.  Issuing Z
ITP002I RUNLOAD  WSIMLU00-00001 - ZEND
ITP201I DISPLAY MONITOR FACILITY IS CLOSED DOWN
ITP079I WSim IS CLOSED DOWN
*** _
```

Figure 28-204 RUNLOAD simulation run messages (Part 2 of 2)

17. The simulated terminals WSIMLU00 and WSIMLU01 were running concurrently. The messages `Attempt to read beyond end of CICSUIS table` are caused by the fact that the UTBL CICSUIS access was set to be the single sequential as shown in Figure 28-137 on page 752.

    The optional completion report for this simulation is shown on Figure 28-205.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help

EDIT       SLITVAK.WTMTEST.RESPROJ1.CREP(R21) - 01.00      Columns 00001 00072
Command ===> ____________________________________________ Scroll ===> CSR
000007 RUN  TIME 14.04.30,  NOVEMBER  3, 2004  VERSION  1  RELEASE  1.0.0
000008 14052667 ITP137I RUNLOAD  WSIMLU01-00001 - WTM: LOGON Finished
000009 14060868 ITP137I RUNLOAD  WSIMLU01-00001 - WTM: INITST7 Finished
000010 14062469 ITP137I RUNLOAD  WSIMLU00-00001 - WTM: LOGON Finished
000011 14062569 ITP137I RUNLOAD  WSIMLU01-00001 - WTM: TRADLOG Finished
000012 14064669 ITP137I RUNLOAD  WSIMLU01-00001 - WTM: SELCOMP Finished
000013 14065369 ITP137I RUNLOAD  WSIMLU00-00001 - WTM: INITST7 Finished
000014 14071070 ITP137I RUNLOAD  WSIMLU01-00001 - WTM: BUY1000 Finished
000015 14074169 ITP137I RUNLOAD  WSIMLU00-00001 - WTM: TRADLOG Finished
000016 14074672 ITP137I RUNLOAD  WSIMLU01-00001 - WTM: SELL900 Finished
000017 14081770 ITP137I RUNLOAD  WSIMLU00-00001 - WTM: SELCOMP Finished
000018 14085973 ITP137I RUNLOAD  WSIMLU01-00001 - WTM: RTQUOTE Finished
000019 14092773 ITP137I RUNLOAD  WSIMLU01-00001 - WTM: LOGOFF Finished
000020 14092971 ITP137I RUNLOAD  WSIMLU00-00001 - WTM: BUY1000 Finished
000021 14093775 ITP137I RUNLOAD  WSIMLU01-00001 - Attempt to read beyond end of
000022 14103172 ITP137I RUNLOAD  WSIMLU00-00001 - WTM: SELL900 Finished
000023 14115173 ITP137I RUNLOAD  WSIMLU00-00001 - WTM: RTQUOTE Finished
000024 14115773 ITP137I RUNLOAD  WSIMLU00-00001 - WTM: LOGOFF Finished
000025 14121373 ITP137I RUNLOAD  WSIMLU00-00001 - Attempt to read beyond end of
000026 Comments:
```

Figure 28-205 Completion report for RUNLOAD simulation

18. Press PF3 to return to the WTM Schedules panel and enter the line command **o** to work with output reports as shown in Figure 28-206.

```
                          Process WSim Schedules                    Row 1 of 3
Command==>
Enter a line command or add to create a new Schedule.        Press PF3 to end.
Line commands: D Delete, N Notes, O Output, P Preprocess, R Resources,
               S Testcases, U UTBLs, X Execute, W NTWRK.

  Name     Type Notes Description                 UTBLs   Last Run
o RUNLOAD    V        load test  transaction         1  11/03/04 14:04 Execute
_ RUNFULL    V        run tradfull group             1  11/02/04 14:38
_ TSTTRAD1   V        test tsttrad testcase             11/03/04 11:43
****************************** Bottom of data ********************************
```

Figure 28-206 Work with RUNLOAD simulation run output requested

19. On the next panel, Reports for schedule RUNLOAD, select option **2** to view the response time report as shown in Figure 28-207.

```
                         Reports for schedule RUNLOAD

Select one of the following.  Then press Enter.

      Command   Action                                                More:  +
2_  1. TLOG      View loglist report from the latest WSim run
    2. TRESP     View response time report from the latest WSim run
    3. TDM       Log display monitor for the latest WSim run
    4. TSP       View SYSPRINT from the latest WSim run

    5. MLOG      View baseline loglist report
    6. MRESP     View baseline response time report
    7. MDM       Log display monitor for the baseline log

    8. COMP      View screen compare report
    9. CDM       Log display comparator
   10. RTCOMP    Edit response time compare report

   11. COMPREP   Edit completion reports


Command===> ____________________________________________________________
```

Figure 28-207 Response time report fro RUNLOAD selected

20. A fragment of the WSim time response report is shown in Figure 28-208.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------
VIEW       SLITVAK.WTM.SYSPRINT                        Columns 00002 00073
Command ===> ____________________________________________ Scroll ===> CSR
000115 WSim RESPONSE TIME REPORT
000116 RUN  TIME 14.04.30,  NOVEMBER  3, 2004  VERSION  1  RELEASE  1.0.0
000117 ------------------------------------------------------------------------
000118 SUMMARY  REPORT    NETWORK  ALL NETWORKS      PROCESS  SYSTEM      TIME
000119                                               EXIT                 START
000120                                               TERMTYPE             END T
000121 ------------------------------------------------------------------------
000122 RESPONSE TIME    COUNT   RESPONSE TIME    COUNT   RESPONSE TIME    COUNT
000123          0.00       30            0.10        2
000124 MEAN   RESPONSE              0.00    MESSAGES SENT            36    NUMBE
000125 MEDIAN RESPONSE              0.10       AVERAGE LENGTH        22       PE
000126 MODE   RESPONSE              0.00       PER MINUTE             4    RESPO
000127 LOW    RESPONSE              0.00    MESSAGES RECEIVED        40    VARIA
000128 HIGH   RESPONSE              0.10       AVERAGE LENGTH       317    95 PE
000129 AVERAGE QUEUE TIME           0.00       PER MINUTE             5
000130 PERCENTILE    RESPONSE TIME         AVERAGE
000131     10                0.00            0.00
000132     20                0.00            0.00
000133     30                0.00            0.00
000134     40                0.00            0.00
```

Figure 28-208 Response time report fragment

28.13 Using SNA trace to create test case

In this section we briefly discuss what should be done to create a test case using option **2** on the WTM Add New Test case panel, shown in Figure 28-9 on page 682.

The Generalized Trace Facility (GTF) is a service aid that can be used to record and diagnose system and program problems. GTF is a part of the MVS system product. Using GTF, a user can record a variety of system and program events on all of the processors in the installation.

The events that GTF traces are specified as options in a parmlib member, as shown in Example 28-2.

Details of GTF operation, which include storage needed, where output goes and recovery for GTF, are defined in a cataloged procedure in SYS1.PROCLIB. GTF can trace system and program events both above and below 16 megabytes. For each event it traces, GTF produces trace records as its output. Writing trace data into data sets impacts system performance.

When using GTF to create SNA traces, the following options are required:

RNIO Requests recording of all Virtual Telecommunications Access Method (VTAM) network activity.

USRP Requests recording of all data that the GTRACE macro passes to GTF.

The option USRP requires a list of specific event identifiers (EIDs), which can represent users, program products or IBM subsystem and component events. The following is the list of assigned EIDs related to VTAM:

FE2 TSO/VTAM TGET/TPUT trace
FE3 VTAM reserved
FF1 VTAM buffer contents trace (USER)
FF0 VTAM SMS (buffer use) trace
FE1 VTAM internal table
FEF VTAM buffer contents trace

As a result, the following must be present in the parmlib member to specify what data will be recorded for WSim VTAM trace:

Example 28-2 SYS1.PARMLIB(GTFWSIM) member

```
TRACE=RNIO,USRP
USR=(FE2,FE3,FF1,FF0,FE1,FEF)
END
```

Example 28-3 represents a fragment of a parmlib proc member which can be used to start the GTF traces with parameters defined in GTFWSIM member presented in Example 28-2.

Example 28-3 SYS1.PROCLIB(GTFPROC) member fragment

```
//GTFPROC PROC MEMBER=GTFWSIM
//        EXEC PGM=AHLGTF,PARM='MODE=EXT,DEBUG=NO,TIME=YES'
//        TIME=1440,REGION=6M
//SYSLIB  DD DSNAME=SYS1.PARMLIB(&PARMLIB),DISP=SHR
//IEFRDER DD DSNAME=xxxxxx,DISP=(NEW,CATLG),
// ...
```

The parameter MODE=EXT forces GTF to direct the output to a trace data set defined by GTFOUTXX or IEFRDER DD statements.

The parameter DEBUG=NO specifies whether GTF will attempt recovery after encountering an error. If DEBUG=YES, GTF will not attempt any recovery. Instead, GTF will issue an error message and end after encountering any error, so that the contents of the trace table immediately prior to the error remain intact. If DEBUG=NO, which is the default, GTF does the following:

- For errors in GTF processing, GTF continues processing after doing one or more of the following:
  - Flagging the trace record or trace record field associated with the error

- Issuing a message to the console to notify the operator that an error occurred
- Suppressing the error or function in which the error occurred

- For errors that do not occur in GTF itself, GTF ends abnormally. If GTF stops processing, that will not cause any other task to also stop.

The parameter TIME=YES specifies that every GTF trace record have a time stamp, as well as the block time stamp associated with every block of data. The time stamp is the eight-byte time of day (TOD) clock value at the local time in which GTF puts the record into the trace table. GTF does not accept TIME=NO; all output records will have time stamps.

We assume that the VTAM startup procedure name is NET and that the user has all required authorizations.

To start the GTF trace, on the SDSF Primary Option Menu, issue the **log** command. On the next screen, issue the following:

```
/F NET,TRACE,TYPE=BUF,ID=vtamappl,AMOUNT=FULL
```

This informs the VTAM (NET application) that the user wants to collect information, FULL trace is mandatory for the trace output to be used by the Workload Simulator, for the subsystem with the APPLID VTAMAPPL.

The next command is:

```
/S GTFPROC.USER
```

It will start GTF trace using the proc presented in Example 28-3 with parameters as defined in Example 28-2. USER is just a name to manage this specific trace.

The users can now log on to application VTAMAPPL, run transactions and so forth, and log off from the application VTAMAPPL. It is very important, that the full process is traced. If the traced terminals are connected before the trace started, the Workload Simulator will not be able to use this trace.

To stop the GTF trace, issue the command:

```
/P USER
```

When adding test cases using option **2** on the WTM Add New Test case panel, the user must specify which GTF trace data set is to be used and which WSim log data set will be produced.

WTM will display all real network resources (displays) used during the trace, so the user must make a selection which one is going to be used for the current test case.

29

Basics of scripting for Workload Simulator

In this chapter we discuss basics of the scripting for Workload Simulator (WSim).

First, we review the scripts that were generated by Workload Simulator Test Manager during some of the activities described in the previous chapter.

Next, we review some statements used for network definitions and basic concepts of coding message generation decks in Structured Translator Language (STL).

Last, we present some operator commands, which can be used to communicate with WSim.

29.1 WTM-generated scripts review

Every WSim script has two components

- Network definition statements describing the configuration of the network being simulated
- Message generation decks that send and receive messages

In this section we review some of the scripts that were created by WTM to illustrate what was being done behind the curtain. The network definition files, created by WTM when the simulations were being prepared (described in the previous chapter), are presented in Figure 29-1.

```
  Menu  Functions  Confirm  Utilities  Help
 ___________________________________________________________________________
 BROWSE            SLITVAK.WTMTEST.RESPROJ1.NTWRKS          Row 00001 of 00003
 Command ===> ___________________________________________ Scroll ===> CSR
              Name     Prompt        Size   Created          Changed          ID
 _________ RUNFULL                    32  2004/11/02  2004/11/02 14:37:54  SLITVAK
 _________ RUNLOAD                    34  2004/11/03  2004/11/03 14:02:41  SLITVAK
 _________ TSTTRAD1                   29  2004/11/03  2004/11/03 11:43:31  SLITVAK
           **End**
```

Figure 29-1 List of Network definitions generated by WTM

The network definition for the simulation RUNFULL is presented in Figure 29-2 and Figure 29-3. Definitions and significance of the entries are presented following the figures.

```
  Menu  Utilities  Compilers  Help
 ___________________________________________________________________________
 BROWSE    SLITVAK.WTMTEST.RESPROJ1.NTWRKS(RUNFULL) - Line 00000004 Col 001 080
 Command ===> ___________________________________________ Scroll ===> CSR
RUNFULL   NTWRK HEAD='run tradfull group',
                BUFSIZE=2048,
                COLOR=MULTI,
                DELAY=F2,
                DISPLAY=(24,80,32,80),
                INIT=SEC,
                ITIME=1,
                LOGDSPLY=BOTH,
                LUTYPE=LU2,
                MLOG=YES,
*               MSGTRACE=YES,
                OPTIONS=(CONRATE,DEBUG,MONCMND),
                RSTATS=YES,
                STLTRACE=YES,
                THKTIME=UNLOCK,
                UTI=100
*          **User Tables**
101        UTBL  CICSUIS    ** user ids and passwords for CICS
*
*          Paths
```

Figure 29-2 Network definition RUNFULL: Part 1

The NTWRK statement is always in the first position in the network definition. This network is named RUNFULL. The parameters include the following:

HEAD A comment that will be shown on reports

COLOR Multiple color support

DELAY Delay time, in this example set to 2 UTIs

DISPLAY Primary and alternate screen sizes

INIT	The secondary LU (terminal) will initiate the session
ITIME	Time (in minutes) between interval reports
LOGDSPLY	Write display buffer to log both before and after message generation
LUTYPE	3270 type terminal
OPTIONS	Options for this network are:
CONRATE	Print interval reports at the WSim console
DEBUG	Write trace data for TCP and CPI-C
MONCMND	Show console commands initiated in a MSGTXT on the WSim console
RSTATS	Response time statistics to be accumulated for LUs
STLTRACE	STL trace records to be written to the log
THKTIME	When “think time” delay interval begins (in this case when the keyboard is unlocked)
UTI	Network-level user time interval in 0.01 sec

The UTBL statement defines that the user table CICSUIS will be used in at least one of the corresponding STL programs. It references the member CICSUIS in the partitioned data set (PDS) SLITVAK.WTMTEST.RESPROJ1.MSGTXTS. The name of 101 was assigned by WSim to this UTBL.

The first PATH statement, named WTMEND, defines the MSGTXT WTMEND which was automatically generated and inserted by WTM to issue the **zend** WSim console command to end the simulation run.

```
  Menu  Utilities  Compilers  Help
 ------------------------------------------------------------------------------
 BROWSE    SLITVAK.WTMTEST.RESPROJ1.NTWRKS(RUNFULL) - Line 00000022 Col 001 080
 Command ===> ___________________________________________________ Scroll ===> CSR
*
*         Paths
WTMEND    PATH  WTMEND
*
*         TestGroup TRADFULL            ¬¬¬ DO NOT CHANGE THIS COMMENT ¬¬¬
1         PATH  LOGON,INITST7,TRADLOG,SELCOMP,BUY1000,
                SELL900,RTQUOTE,LOGOFF
*
*         Simulated Resources
VAPPL00   VTAMAPPL
WSIMLU00  LU   PATH=(1,1,WTMEND)
******************************** Bottom of Data ********************************
```

Figure 29-3 Network definition RUNFULL: Part 2

The second PATH statement, named 1, defines the list of MSGTXTs and the order in which they will be executed.

The VTMAPPL statement must precede any LU statements and defines that the following LUs will simulate VTAM resources.

The LU statement defines a single network resource and the script program path to be executed by it, in this case the path 1 first and then the path WTMEND.

The set of STL programs, which includes the STL programs for each of the test cases created by WTM, is shown in Figure 29-4.

```
  Menu  Functions  Confirm  Utilities  Help
 ------------------------------------------------------------------------------
 BROWSE            SLITVAK.WTMTEST.RESPROJ1.STL                Row 00001 of 00014
 Command ===> ____________________________________________________ Scroll ===> CSR
            Name     Prompt        Size   Created          Changed          ID
 _________ CICSUIS                   9  2004/10/28  2004/10/29 15:01:22  SLITVAK
 _________ LOGON                    62  2004/10/28  2004/10/28 19:13:38  SLITVAK
 _________ WTMVARS                 543  2004/10/25  2004/10/28 19:08:42  SLITVAK
 _________ INITST7                  47  2004/10/28  2004/10/28 15:11:54  SLITVAK
 _________ LOGOFF                   36  2004/10/27  2004/10/27 14:47:09  SLITVAK
 _________ RTQUOTE                  45  2004/10/27  2004/10/27 14:42:17  SLITVAK
 _________ SELL900                  44  2004/10/27  2004/10/27 14:37:45  SLITVAK
 _________ BUY1000                  44  2004/10/27  2004/10/27 14:33:19  SLITVAK
 _________ SELCOMP                  35  2004/10/27  2004/10/27 14:28:33  SLITVAK
 _________ TRADLOG                  40  2004/10/27  2004/10/27 14:22:22  SLITVAK
 _________ TSTTRAD                 180  2004/10/25  2004/10/25 18:54:11  SLITVAK
 _________ CPICCON                 103  2004/10/25  2004/10/25 14:32:53  SLITVAK
 _________ CPICVARA                 49  2004/10/25  2004/10/25 14:32:53  SLITVAK
 _________ CICSUISX
           **End**
```

Figure 29-4 List of STL programs

We continue with the review of the WTM-generated STL programs using the STL program for the test case LOGON as an example, as shown in Figure 29-5.

```
  Menu  Utilities  Compilers  Help
 ------------------------------------------------------------------------------
 BROWSE    SLITVAK.WTMTEST.RESPROJ1.STL(LOGON) - 01.0 Line 00000000 Col 001 080
 Command ===> ____________________________________________________ Scroll ===> CSR
******************************** Top of Data ********************************
@program=LOGONT
@include wtmvars
LOGON: msgtxt
/*-------------------------------------------------------------------*/
/* ITPIDC: DISPLAY=TSOCON   APPLICATION=CICSC22F 14:59:08.52 10/28/04*/
/*   ----------- DISPLAY CHARACTERISTICS AND FEATURES ------------   */
/*   ALTCSET=APL                APLCSID=(963,310)                    */
/*   BASECSID=(697,37)          CCSIZE=(8,15)     COLOR=MULTI        */
/*   DBCS=NO                                                         */
/*   DISPLAY=(24,80,24,80)      DLOGMOD=LSX32702  EXTFUN=YES         */
/*   FLDOUTLN=NO                FLDVALID=NO       HIGHLITE=YES       */
/*   MAXNOPTN=0                 PS=NONE           UOM=INCH           */
/*-------------------------------------------------------------------*/
/* ITPLSGEN: SCRIPT GENERATION PARAMETERS        15:05:48.46 10/28/04*/
/*   INPUT    SLITVAK.WTMUSER.IDCLOG(LOGON)                          */
/*   OUTPUT   SLITVAK.WTMTEST.RESPROJ1.STL                           */
/*   MSGTXT   LOGON                                                  */
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 29-5 STL program for test case LOGON, part 1

The statement @PROGRAM specifies the start of the STL program and defines its name, in this case LOGONT.

The statement @INCLUDE retrieves a member from the SYSLIB concatenation as program statements. The member WTMVARS is a standard include member for WTM.

The statement MSGTXT defines the name of the MSGTXT member to be generated from this STL program, in this case LOGON.

The statement ONIN, as shown in Figure 29-6, is an asynchronous statement.

```
  Menu  Utilities  Compilers  Help
 ------------------------------------------------------------------------
 BROWSE    SLITVAK.WTMTEST.RESPROJ1.STL(LOGON) - 01.0 Line 00000020 Col 001 080
 Command ===> ____________________________________________ Scroll ===> CSR
/*   LU        IDCSLU-1                                                 */
/*   STL       TRACE=LOGONT                                             */
/*   NOVERIFY                                                           */
/*-------------------------------------------------------------------*/

/*---------------------------------------------------- 14590852 00001 */
onin0001: onin substr(ru,1,1) = 'F5'x,
                then found = on
found = off
initself('CICSC22F','LSX32702')
do while found = off             /* wait for onin0001 data received */
 wait until onin
end
deact onin0001

/* 14:59:08.54 ITP1507I SESSION STARTED WITH APPLICATION CICSC22F */

/*---------------------------------------------------- 14591780 00001 */
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 29-6 STL program for test case LOGON, part 2

It will set the value of the variable FOUND to ON when the first character of the variable RU is equal to the hexadecimal ‘F5’. The variable RU is a special variable that contains the SNA request/response unit portion of incoming or outgoing data.

The value of the variable FOUND is first set to OFF.

The statement INITSELF starts a session with VTAM application with APPLID CICSC22F from this LU. User data, which equals to LSX32702, is also sent to that VTAM application.

The statement DO WHILE loops executing the statement WAIT UNTIL ONIN until the session gets initialized and a response is received.

The statement DEACT ONIN0001 deactivates the asynchronous ONIN statement labeled ONIN0001.

The statement LOG writes a record, containing the name that WTM assigned to the panel, to the log data set as shown in Figure 29-7.

```
  Menu  Utilities  Compilers  Help
 ------------------------------------------------------------------------
 BROWSE    SLITVAK.WTMTEST.RESPROJ1.STL(LOGON) - 01.0 Line 00000037 Col 001 080
 Command ===> ____________________________________________ Scroll ===> CSR
/*------------------------------------------------- 14591780 00001 */
WTM_panel_ID = 'PNL00001'
log 'WTM_panel_ID' WTM_panel_ID
cursor(10,26)
ereof
charset 'field'
/* WTM has replaced the following line with a user table: */
/* type 'slitvak' */
call CICSUISX     /* Access CICSUIS usertable */          /* WTM */
type CICSUIS_USERID                                       /* WTM */
cursor(11,26)
ereof
/* WTM has replaced the following line with a user table: */
/* type '*******' */
type CICSUIS_PASSWORD                                     /* WTM */
transmit using enter

/*------------------------------------------------- 14592088 00003 */
  F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
 F10=Left   F11=Right  F12=Cancel
```

Figure 29-7 STL program for test case LOGON, part 3

The cursor is moved to the position specified by row and column on the screen (10,26).

The statement EREOF erases from the cursor current position to the end of the field.

The next statement defines that character set will be determined by field attributes (this is the default).

The CALL statement executes the MSGTXT which was generated by WTM to access the UTBL CICSUIS.

The statement TYPE places data into the screen image. In this case, it is the data from the column CICSUIS_USERID in the UTBL CICSUIS.

These steps are repeated for the password.

TRANSMIT the data and simulate pressing the ENTER key.

The statement SAY writes a line to the WSim console as shown in Figure 29-8.

```
  Menu  Utilities  Compilers  Help
 ------------------------------------------------------------------------
 BROWSE    SLITVAK.WTMTEST.RESPROJ1.STL(LOGON) - 01.0 Line 00000054 Col 001 080
 Command ===> ____________________________________________ Scroll ===> CSR
/*------------------------------------------------- 14592088 00003 */
WTM_panel_ID = 'PNL00002'
log 'WTM_panel_ID' WTM_panel_ID
transmit using clear

/* 15:05:21.92 ITP1508I SESSION ENDED WITH APPLICATION CICSC22F */

say 'WTM:' msgtxtid() 'Finished'
endtxt
******************************** Bottom of Data ********************************
```

Figure 29-8 STL program for test case LOGON, part 4

This is the end of the STL program generated by WTM for the test case LOGON.

29.2 Network definition statements

Network definition statements specify the following information:

- Types of the simulated resources on the network
- Attributes of the simulated resources
- Connections between the simulated resources and the system under test
- Special information about delays, logic tests, the order in which message generation decks are used, logging or tracing of the messages, and so forth

Network definition statements consist of the name (optional except on the NTWRK statement), the statement, and one or more operands (optional), as shown in Example 29-1.

Example 29-1 NTWRK statement

```
RESNET1 NTWRK REPORT=RATE,HEAD=' RESNET1 RATE REPORT '
```

The network named RESNET1 is defined, the interval report will include only the network totals and message rates, and the heading of the report will be "RESNET1 RATE REPORT."

NTWRK is always the first statement used to define a network. It names the network and specifies characteristics that apply to the network as a whole. It also may specify operands that establish defaults for lower-level statements. All other statements in the network definition follow the NTWRK statement in a prescribed order. The statements from the general simulation statements group immediately follow the NTWRK statement.

Different statements are used depending on what type of a network is being simulated. For example: when simulating LUs accessing VTAM application, the VTAMAPPL and LU statements must be used; when simulating CPI-C TPs, APPCLU and TP statements; when simulating TCP-IP clients, TCP/IP and DEV statements.

Not all statements are mandatory and some could be coded more than once, but all the statements in each and every group, including the optional ones, should follow the prescribed order.

VTAMAPPL statement

Use the VTAMAPPL statement to define a VTAM application program.

Use either the name field or the APPLID operand to specify the symbolic name of the application. This name must match the corresponding entry in VTAM's configuration tables (VTAMLST), created using an APPL definition statement in VTAM.

Use the PASSWD operand to specify the password associated with the symbolic name of the VTAM application. The value specified must match the value of the PRTCT operand on the APPL statement in VTAM.

LU statement

Use the LU statement to define one or more LU half-sessions and specify the type of half-session to be simulated. WSim simulates each LU half-session as a separate entity (single display, terminal, device) for message generation and message logging. Operands on the LU statement can be used to define half-session characteristics needed to generate and interpret data streams associated with corresponding products, for example, 3270 devices.

At least one LU statement must follow a VTAMAPPL statement as shown in Example 29-2.

Example 29-2 VTAMAPPL and LU statements

```
RESVTAM1 NTWRK INIT=SEC
WSIMLU VTAMAPPL
USERRES1 LU LUTYPE=LU2,RESOURCE=TSO
```

APPCLU statement

Use the APPCLU statement to define CPI-C LU.

Use either the name field or the APPLID operand to specify the symbolic name of the LU. This name must match the corresponding entry in VTAM's configuration tables (VTAMLST) which was created using an APPL definition statement in VTAM, and this corresponding entry must specify APPC=YES.

Use the PASSWD operand to specify the password associated with the symbolic name of the VTAM application. The value specified must match the value of the PRTCT operand on the APPL statement in VTAM.

TP statement

Use the TP statement to define one or more transactions programs and to specify whether this TP is a client or a server. WSim simulates each TP as a separate entity for message generation and message logging.

At least one TP statement must follow an APPCLU statement, as shown in Example 29-3.

Example 29-3 APPCLU and TP statements

```
RESCPIC1 NTWRK
LU1 APPCLU
TP1 TP TPTYPE=CLIENT
```

TCPIP statement

Use the TCPIP statement to define a connection to the IBM TCP/IP for MVS product on the local host. Use the TCPNAME operand to specify the name of the TCP/IP virtual machine or address space.

DEV statement

Use the DEV statement to define a TCP/IP client to be simulated.

Use the TYPE operand to specify the type of client. Multiple clients of different types may be specified for a single TCP/IP statement. Use the SERVADDR operand to specify a server to which the client can connect. Use the PORT operand to specify a port to be used for connection.

At least one DEV statement must follow a TCPIP statement as shown in Example 29-4.

Example 29-4 TCPIP and DEV statements

```
RESTCPIP NTWRK
RESCONN TCPIP
RES1 DEV TYPE=TN3270,SERvaDDR=9.255.255.255,PORT=23
RES2 DEV TYPE=FTP,SERvaDDR=9.255.255.254
```

General order of network definition statements

The order of valid network definition statements is listed in Table 29-1.

Table 29-1 Order of valid network definition statements

Statement	Can be followed by	Comments
NTWRK	NTWRKLOG, MSGDISK, RATE, UTBL, SIDEINFO, RN, UDIST, INCLUDE, IF, PATH	Required and must be the first statement of a network
NTWRKLOG	MSGDISK, RATE, UTBL, SIDEINFO, RN, UDIST, INCLUDE, IF, PATH	Optional
MSGDISK	RATE, UTBL, SIDEINFO, RN, UDIST, INCLUDE, IF, PATH	Optional
RATE	RATE, UTBL, SIDEINFO, RN, UDIST, INCLUDE, IF, PATH	Required if using rate table delays
UTBL	UTBL, SIDEINFO, RN, UDIST, INCLUDE, IF, PATH	Optional
SIDEINFO	SIDEENT	Optional
SIDEENT	SIDEENT, SIDEEND	At least one is required after SIDEINFO
SIDEEND	RN, UDIST, INCLUDE, IF, PATH	Required after the last SIDEENT
RN	RN, UDIST, INCLUDE, IF, PATH	Optional
UDIST	UDIST, INCLUDE, IF, PATH	Optional
INCLUDE	INCLUDE, IF, PATH	Optional, mostly used to define required but not referenced message generation decks
IF	INCLUDE, IF, PATH	Optional
PATH	PATH, DIST, UTI, FILE, APPCLU, VTAMAPPL, TCPIP	At least one is required
DIST	DIST, UTI, FILE, APPCLU, VTAMAPPL, TCPIP	Optional
UTI	UTI, FILE, APPCLU, VTAMAPPL, TCPIP	Optional
FILE	FILE, APPCLU, VTAMAPPL, TCPIP	Optional
APPCLU	TP	Optional
TP	TP, APPCLU, VTAMAPPL, TCPIP, FE	At least one is required after APPCLU
VTAMAPPL	LU	Optional
LU	LU, VTAMAPPL, TCPIP, FE	At least one is required after VTAMAPPL
TCPIP	DEV	Optional
DEV	DEV, TCPIP, FE	At least one is required after TCPIP
FE	FE	Optional

29.3 Coding STL programs

STL is a high-level structured programming language that can be used to create message generation decks and define terminals and devices to be simulated by WSim. STL utilizes constants, variables, expressions, and structured control statements.

An STL program is usually divided into one or more procedures. The STL Translator translates STL programs into message generation decks, each message generation deck corresponding to one STL procedure. Network definitions can be included into STL programs. The STL Translator invokes the preprocessor to validate and store the network definition statements.

STL is similar but not equivalent to REXX. STL uses many of the same commands and functions as REXX. However, STL is not the run-time scripting language; therefore, WSim must process or translate STL into MSGTXT format prior to test execution.

Variables and constants

In STL, you can use variables and constants as data items in your programs. A variable contains data that is used by a program in a certain way, but whose value can vary. In a program, each variable has a unique symbolic name.

Constants are values that do not change in the course of program execution. Constants can be used to initialize variables, to test the contents of variables, and to generate messages. They can also make up part of STL expressions. Names can be assigned to most constants. STL substitutes the actual constant value for the constant name.

Variables are classified as one of three data types:

- Integer
- String
- Bit

Integer variables can take any positive integer value from 0 to 2147483647. When a network is first initialized, the value of an integer variable is 0. Integer variables translate into counters in the scripting language. An integer variable's value will wrap if incremented beyond 2147483647 or decreased below 0. That is, if a variable has a value of 2147483647 and 1 is added to it, the result is 0. If a variable has a value of 2 and 3 is subtracted from it, the result is 2147483647.

String variables can contain only characters. They must be from 0 to 32767 characters long. A character can be any 1-byte value, from '00'X to 'FF'x. When a network is first initialized, the value of a string variable is '' (the null string, which is represented by a pair of single or double quotation marks). String variables translate into save areas in the scripting language.

Bit variables represent binary data items. STL bit variables can have the value of ON or OFF. When a network is initialized, all bit variables have a value of OFF. Bit variables translate into switches in the scripting language.

Special variables are available to be used by your STL program:

BUFFER or **SCREEN** For nondisplay terminals, the device buffer; for display terminals, the screen image. Use the SUBSTR and ROWCOL functions to query a screen area, for example, x = substr(screen,rowcol(10,20),6).

DATA or **TH** Incoming or outgoing data, including SNA headers if present.

RH The SNA request/response header (RH) portion of incoming or outgoing data plus the SNA request/response unit (RU).

RU The SNA Request/response unit (RU) portion of incoming or outgoing data.

Constants are classified as:

- Integer
- String
- Hexadecimal string
- Bit data

Once a type is associated with a variable or named constant, you cannot change the type in the course of a program.

Integer constants are positive decimal integers from 0 to 2147483647.

String constants are any set of characters. A string constant must be enclosed in a pair of single or double quotation marks (the string delimiter character). If a string constant contains the string delimiter character (a single or double quotation mark), that character must be entered twice so that it will be recognized. You will probably want to use single quotes as your delimiter character for strings containing double quotes, and double quotes as a delimiter for strings containing single quotes. You do *not* have to use the same delimiter character throughout your program; you can use whichever is most appropriate for each string you are enclosing. The string constants '' and "" have a length of zero and are called the null string.

Hexadecimal string constants are specified by enclosing pairs of hexadecimal digits in string delimiters followed by the character x or X. Each pair of hexadecimal digits represents a single character in the string.

Bit constants can take the value ON or OFF. These constants can be assigned to bit variables or used to test the current setting of a bit variable.

Functions

STL provides a number of built-in functions that can be used to access and manipulate data. All functions return a value, which can be an integer, string, or bit value. You can use functions as expressions or parts of expressions wherever variables can be used. STL function names are reserved words. You cannot use them as names or labels.

A function consists of the name of an STL function, followed by a left parenthesis, any arguments for the function, and a right parenthesis. The function arguments consist of values you are supplying to the function. Even if the function does not use arguments, you must code the parentheses. If you specify more than one argument, you must separate them with commas.

You can use the SUBSTR function (substring function) to reference a portion (or a substring) of a string expression.

The INDEX function returns an integer value that gives the position of a target string in a source string. If the target string is not found in the source string, the function returns a value of 0.

A number of functions perform translations between data types, for example:

B2X() and **X2B()** From binary string to hexadecimal string and the opposite

C2D() and **D2C()** From hexadecimal string to its decimal (integer) value and the opposite

C2X() and **X2C()** From character (EBCDIC) string to hexadecimal string and the opposite

CHAR() From integer value to its EBCDIC character representation

Statements

STL has two types of statements:

- Synchronous
- Asynchronous

Synchronous statements are executed in logical order, as programmed, as shown in Example 29-5.

Example 29-5 Synchronous statement

```
if substr(screen,1,4) = 'WSIM' then do
cursor(1,1)
type 'Workload Simulator'
end
transmit using enter
```

Asynchronous statements are executed when WSim detects a specified condition when a message is sent or received, as shown in Example 29-6.

Example 29-6 Asynchronous statements

```
onin01: onin substr(ru,1,1)='F5'X then,
found = on
```

The following asynchronous STL statements allow you to test messages received and transmitted:

ONIN Defines a condition that is to be tested when data is received by a simulated terminal

ONOUT Defines a condition that is to be tested when data is transmitted by a simulated terminal

ON SIGNALED Defines an action to be taken when the specified event is signaled (by means of a SIGNAL or QSIGNAL command)

WAIT UNTIL and **QUIESCE UNTIL** Interrupts STL execution and defines a condition that, when met, allows STL program execution to resume (can also be coded on the TRANSMIT statement)

IF Coded in a NTWRK statement, it checks a condition at a network level when data is sent or received

The DEACT statement will deactivate an asynchronous condition.

Several statements are available for working with the screen images, simulating the operator pressing various keys. Some of these statements are presented in Table 29-2.

Table 29-2 STL Statements for work with screen images

Statement	Action
TYPE expression	Type the expression at the current cursor position.
CURSOR(*n.m*)	Move the cursor to row *n* and column *m.*
CURSOR("UP")	Move the cursor up 1 row.

Statement	Action
CURSOR("DOWN", 10)	Move the cursor down 10 rows.
TAB	Tab to the right (move cursor to next input field on screen).
BTAB	Tab to the left (back up one input field).
HOME	Home key (move cursor to beginning of first input field on screen).
RESET	Reset key.
INSERT	Insert key.
DELETE	Delete beginning with the character at the current cursor position.
EREOF	Erase to end of field.

STL provides several statements to transmit messages from simulated terminals, receive messages, and take actions based upon messages sent or received.

Using the TRANSMIT statement is the most common method of sending messages. (The TRANSMIT statement cannot be used in CPI-C transaction program simulations. Use the CMSEND statement instead).

The most typically used form of the TRANSMIT statement is shown in Example 29-7.

Example 29-7 Common form of TRANSMIT statement

```
TRANSMIT USING key
```

In this example, *key* can be ENTER (default if omitted), CLEAR, PFnn, PAn, SYSREQ, and some others.

The TRANSMIT statement does not wait for a response; execution of the program continues immediately, unless the user optionally coded a WAIT statement with TRANSMIT, as shown in Example 29-8.

Example 29-8 TRANSMIT and WAIT statements

```
TRANSMIT USING ENTER AND WAIT UNTIL ONIN
```

The WAIT statement causes the STL program to wait until a message is received; the STL program waits and then continues execution after a message is received.

The common forms of the WAIT statement are presented in Table 29-3.

Table 29-3 Common forms of WAIT statement

WAIT statement	Action
WAIT UNTIL ONIN	Wait until a message is received
WAIT UNTIL ONOUT	Wait until a message is sent (by an asynchronous statement)
WAIT UNTIL POSTED(*event name*)	Wait until the named event has been posted
WAIT UNTIL SIGNALED(*event name*)	Wait until the named event has been signaled

STL provides mechanisms to control the flow of programs by using a subset of statements called structured flow-of-control statements. These statements enable WSim to move through a program in a nonlinear fashion.

Structured flow-of-control statements give an ability to have a procedure call another procedure, passing control to the second procedure. They also allow taking a specific action depending upon conditions that occur.

There are four types of control statements:

- CALL
- IF/THEN/ELSE
- SELECT
- DO

The CALL statement shifts control of program execution from one procedure to another procedure. The IF/THEN/ELSE statement and the SELECT statement group execute statements selectively depending on conditions that exist. The DO statement group enables the user to group statements logically and in some cases to execute statements repetitively.

29.4 WSim operator commands

The user can issue WSim operator commands to:

- Initialize a simulated network
- Start a simulation run
- Query the status of a simulated network device
- End the simulation run

The WSim operator commands can be entered when running WSim in TSO foreground session and when running WSim as a batch job.

A few examples of WSim operator commands are shown in Table 29-4.

Table 29-4 Samples of WSim operator commands

WSim operator command	Result
I testnet	Initialize the network named testnet.
I testnet,S	Initialize the network named testnet and start the network activity.
S	Start activity for all simulated resources in all initialized networks.
S testnet	Start activity for all simulated resources in the network named testnet.
Q	Display the status of all initialized networks.
Q lu	Display the status of the simulated resource named lu.
W lu	Display response time statistics (RSTATS) for the simulated resource named lu.
P	Stop activity for all simulated resources in all networks.
P resourcename	Stop activity for the resource named resourcename.
ZEND	Causes orderly shutdown of the WSim and writes the message log buffers to the log data set.

Part 8

Scenarios

Part 8 covers various scenarios using the problem determination and deployment tools. The chapters cover:

- Introduction to scenarios
- Guided tour of the Trader application
- Scenario 1: Invalid VSAM data generating an abend
- Scenario 2: Using Debug Tool
- Is there an error in the DB2 data or program logic?
- Using Fault Analyzer and File Manager
- Application Monitor and CICSplex configuration
- Application Monitor and Parallel Sysplex

30

Introduction to the scenarios

These scenarios were designed to highlight features of the Problem Determination and Deployment Tools in a brief, but effective manner.

In this chapter we cover these topics:

- An overview of the scenarios
- How to install the application software
- The system configuration
- How to validate the installation

30.1 Overview

The scenarios presented in this redbook are based on the “Trader” stock trading application, which was written specifically to demonstrate the features of the Application Development Tools. The users of this application might be investors checking their holdings, or buying and selling shares of stock. The application takes two forms:

- CICS transactions
- Batch jobs

Each form can access data on VSAM and DB2.

> **Note:** This application does not reflect real-world securities processing. It is merely designed to demonstrate the features of the Application Development Tools.

The next chapter presents a more detailed look at the Trader application. In the other chapters in this part, we create scenarios based on the Trader application. In each scenario but the last two, we deliberately introduce errors into the application to allow us to demonstrate the functionality of the tools. We then describe, in detail, the steps that you take to isolate the error and to correct the problem.

In the scenarios described in Chapter 32 (Invalid VSAM data generating an abend), Chapter 33 (Using Debug Tool), Chapter 34 (Is the error in DB2 data or program logic), Chapter 35 (Workload Simulator and Application Monitor) and Chapter 36 (Application Monitor and CICSplex configuration), all the applications are running on a single partition.

In the scenario described in Chapter 37 (Application Monitor and Parallel Sysplex) all the resources used by the applications are defined to use a sysplex configuration.

30.1.1 Overview of the programs

The Trader application is used to maintain a stock portfolio held by an individual. This application enables you to:

- Obtain quotes (in batch mode, you list portfolios and their values)
- Buy more shares of a company’s stock
- Sell currently held shares of a company’s stock

The Trader application uses two data sources:

- Company data
- Customer data

The company source contains the stock name and the past week’s quotes. The customer source contains a record for each customer and company that he or she owns, including the number of shares held.

There can be two VSAM files, two DB2 tables, in the MYTRADD DB2 plan.

In the CICS application, the transaction input is taken directly from an online user’s interactions. In the batch application, the user’s input is replaced with a sequential file that contains several records representing the day’s transactions.

Overview of the CICS program

Figure 30-1 shows the processing that occurs in the CICS application.

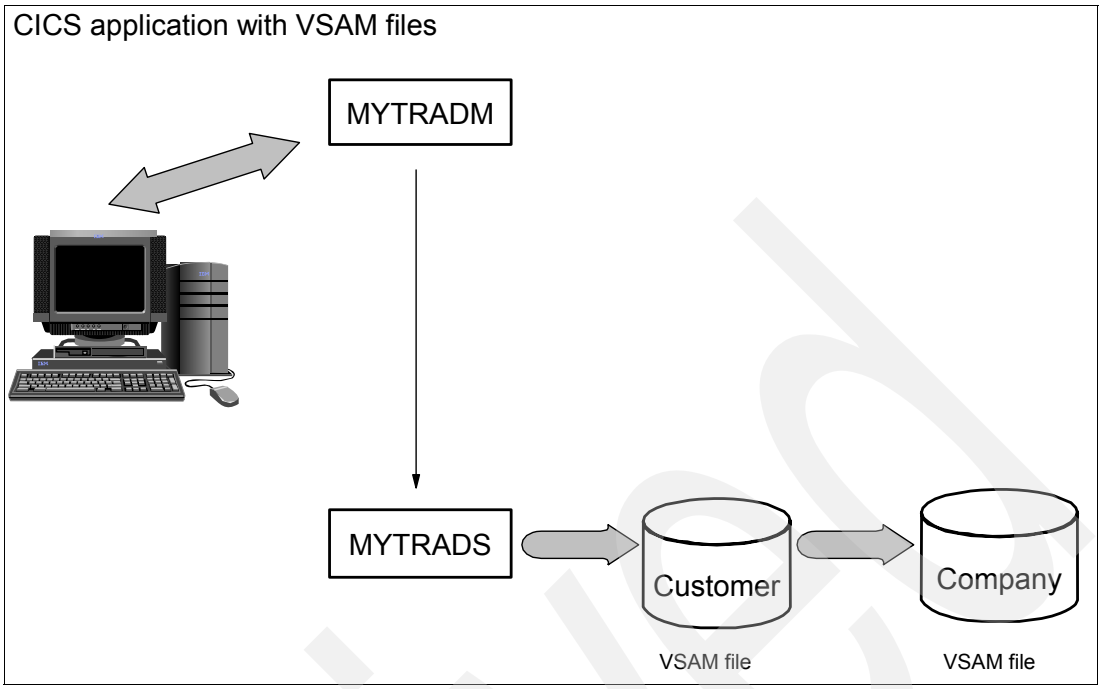


Figure 30-1 Trader application: Single user transaction with CICS

Note: When you invoke this application, you can use any user name and password. But if you want to see the status from previous trading, use the same user name each time.

Overview of the batch program

Figure 30-2 shows the processing that occurs in the batch application.

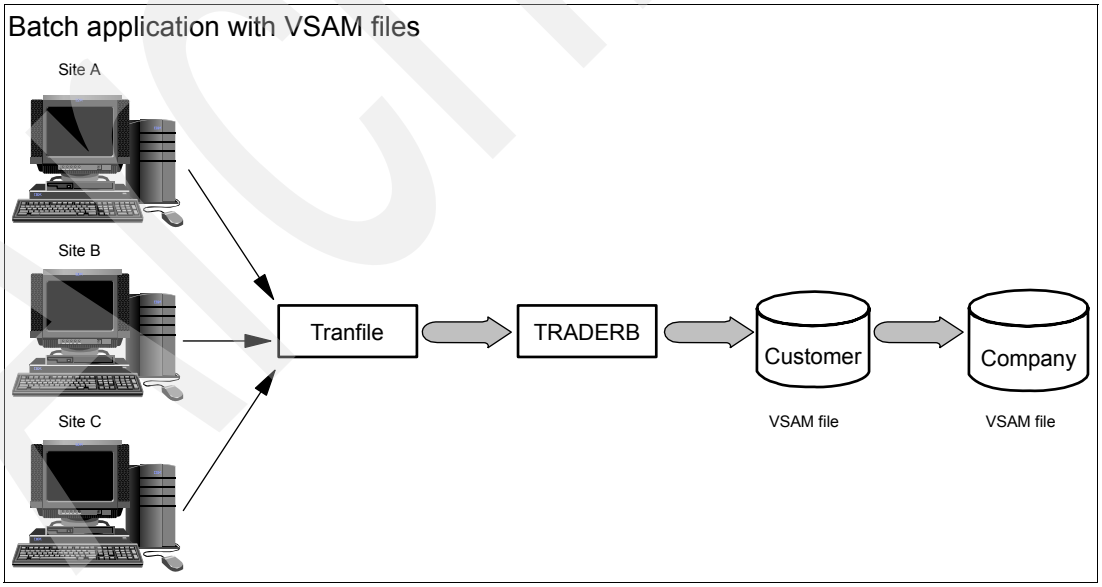


Figure 30-2 Trader application: Multiple remote site transactions with batch

Note: You should always list the holdings of a user name to determine the number of shares in a portfolio before you begin to trade with it.

30.1.2 Software prerequisites

The application programs listed in Table 30-1 were created for this IBM Redbook and are installed on our system. These application programs were designed to demonstrate the functionality of the Application Development Tools.

Table 30-1 Application programs used in Trader application

Application program	Subsystem	Purpose
MYTRADMV MYTRADS	CICS	Retrieve customer information from VSAM files
TRADERB	Batch	Process customer transactions from sequential and VSAM files
MYTRADMD MYTRADD	CICS and DB2	Retrieve information from DB2 tables
TRADERD	Batch and DB2	Process customer informations from sequential file and DB2 tables

30.2 Install the application software

In this section, we describe how to install the application software that you use to run the different forms of the Trader application.

You can install and use only the parts relevant to your system configuration.

If you intend to follow the examples on your own, you will need the system software. Refer to Section 30.3.1, “About the software prerequisites” on page 815. We assume you have access to a similar configuration.

30.2.1 Install the demo files

You need to install the application software that you will use to run the different forms of the Trader application.

Locating the Web material

The Web material associated with this redbook is available in soft copy on the Internet from the IBM Redbooks Web server. Point your Web browser to:

ftp://www.redbooks.ibm.com/redbooks/SG246492

Alternatively, you can go to the IBM Redbooks Web site at:

ibm.com/redbooks

Select the **Additional materials** and open the directory that corresponds with the redbook form number, SG246492.

Using the Web material

The additional Web material that accompanies this redbook includes the following file:

File name	Description
SG246492.zip	Zipped code samples

System requirements for downloading the Web material

The following system configuration is recommended:

Hard disk space:	4 MB for the downloaded zip file and unpacked files
Operating System:	Windows 2000/XP
Processor:	Pentium
Memory:	128 MB

How to use the Web material

Create a subdirectory (folder) on your workstation, and unzip the contents of the Web material zip file into this folder.

The extracted files are all in binary format. They are the output of the TSO TRANSMIT command.

Use your mainframe file transfer protocol to upload the binary files. You must use the following attributes: FB, LRECL=80, BLKSIZE=3120.

After each file is uploaded, issue the following command from the TSO READY prompt:

```
RECEIVE INDA(xxxx)
```

In this command, *xxxx* is the name of the file.

If you issue the command against the **source** file, you will receive the messages shown in Example 30-1.

Example 30-1 Receive INDA(xxxx) messages

```
INMR901I Dataset DAVINR1.REDBOOK.PDPAK.SOURCE from DAVINR1 on NODENAME
INMR906A Enter restore parameters or 'DELETE' or 'END' +
```

You can to reply as shown in Example 30-2.

Example 30-2 Receive INDA(xxxx) reply

```
indsn(pdtools.redbook.source)
```

The default high-level qualifier assigned to the file will be your TSO user ID, named *userid.*

> **Note:** You can delete the zipped file and the temporary folder after you finish uploading all of the files.

30.2.2 Customization

Edit the JCL data set and change all instances of TSOUSERID to your TSO user ID. You can use the File Manager Find/Change utility to perform this step.

You need to edit some members of the *userid*.PDTOOLS.REDBOOK.JCL data set to validate the following required information:

- For DB2:
  - DB2 load library
  - DB2 run-time library
  - DB2 DBRM library

- For COBOL:
  - COBOL compiler load library
- For Language Environment
  - LE run-time library
- For CICS:
  - CICS system load library
  - CICS user load library
  - CICS macro/DSECT library
  - CICS CSD data set name
- For Debug Tool:
  - Debug Tool system load library

The modifications must be made in the following members of the *userid*.PDTOOLS.REDBOOK.JCL:

- BIND
- CICSDB2C
- COBPROC
- COBPROCB
- DB2CXCOB
- DEFPDPAK
- DFHMAPS
- DSNHICOB
- GENMAP
- GRANT
- TABLES

In addition, if you have not already done so, you need to change the string, TSOUSERID, to your own TSO user ID.

You also need to get your DB2 subsystem name; the following jobs will use it:

- BIND
- DATA
- GRANT
- TABLES

30.2.3 Set up the applications

The starting point for the scenarios is an established stock trading application.

Generate the data

Perform the following steps if you want to set up this application at your site:

1. Generate the NEWTRAD mapset using the GENMAP job.
2. Define the two VSAM data sets (COMPFILE and CUSTFILE) with the DEFVSAM1 job.

   This loads the VSAM files with sample data. If you wish, you can use the File Manager copy utility to perform this step.
3. Define the DB2 tables, CUSTOMER_DETAILS and COMPANY_DETAILS, with *userid*.PDTOOLS.REDBOOK.JCL(TABLES).
4. Populate these DB2 tables with *userid*.PDTOOLS.REDBOOK.JCL(DATA) or using File Manager.

Generate the executables

Perform the following steps to generate the executables:

1. Compile all of the COBOL programs with the appropriate batch compile job, as shown in Table 30-2.

> **Note:** The batch job *userid*.PDTOOLS.REDBOOK.JCL(CICSDB2C) allows you to preprocess, compile, and link-edit the program MYTRADD.
>
> **Note:** The batch job *userid*.PDTOOLS.REDBOOK.JCL(COBCICS) contains the JCL to compile a COBOL/CICS program, which can be used to compile MTRADMD. You have to change the program name in the symbolic parameter MEM and in the NAME card of link-edit step.

Table 30-2 COBOL application programs with compile job names

Program	Member	Proc
MYTRADMV	COBCICS	COBPROC
MYTRADS	COBCICS	COBPROC
TRADERB	COBBATCH	COBPROCB
MYTRADD	CICSDB2C	DB2CXCOB
MYTRADMD	COBCICS	COBPROC
TRADERD	COBBDB2	DB2COBBA
MYTRADMI	COBCICS	COBPROC

> **Note:** Make certain you validate the names of all the product libraries before you submit these batch jobs. For some batch jobs, you may need to pre-allocate your output data sets.

To compile your COBOL, DB2, or CICS applications, we recommend that you use the DB2 (DB2 V7 and later) or CICS (CICS-TS V2.2 and later) integrated precompiler or translator for two reasons:

– Debugging becomes much more comfortable, because you will not be presented with the inserted resolution of your EXEC SQL or EXEC CICS statements.

– If you use the external precompiler or translator, you will need to save the precompiler/translator output (which becomes the input to the compiler) to a permanent data set. This is not needed if you compile with the integrated precompiler/translator, since your source is directly the input to the compilers. Example 30-3 shows an example of the COBPROC procedure modified as such.

Example 30-3 Extracts of a procedure with integrated translator

```
//*********************************************************************
//CIXCOB   PROC MEM='TRADPROG',        DEFAULT SOURCE MODULE MEMBER NAME
...
// COMPARM='NODYNAM,LIB,OBJECT,RENT,APOST,MAP,XREF,LIST,SOURCE',
// COMPAR1='TEST(NONE,SYM,SEPARATE),CICS("COBOL3 ANSI85")',
...
//*********************************************************************
//COBCOMP  EXEC PGM=IGYCRCTL,
```

```
//             PARM='&COMPARM,&COMPAR1'
//STEPLIB  DD  DISP=SHR,DSN=&COBLOC
//         DD  DISP=SHR,DSN=CICSTS23.CICS.SDFHLOAD
//*******************************************************************
/* DD above added for CICS level
//*******************************************************************
//SYSLIB   DD  DISP=SHR,DSN=&DSCTLIB
//         DD  DISP=SHR,DSN=&USRCCPY
//*******************************************************************
//* PRE is commented out
//*******************************************************************
//*SYSIN    DD  DISP=(OLD,DELETE),DSN=&&DFHHOUT
//SYSIN    DD  DISP=SHR,DSN=&COBSRC(&MEM)
...
```

2. Create the DB2 plan, MYTRADD with *userid*.PDTOOLS.REDBOOK.JCL(BIND).
3. Grant execution access to this plan with *userid*.PDTOOLS.REDBOOK.JCL(GRANT).
4. Define all the necessary application resources to CICS:
   a. The MYTRADxx programs from Step 5.
   b. The mapset NEWTRAD from Step 1.
   c. The transactions MYTD, TDB2.
   d. The two VSAM files from Step 2.

   These resource definitions are contained in *userid*.PDTOOLS.REDBOOK.JCL(PDPAK).

   Review this file for changes that are relevant to your site's standards.
5. To add these definitions to the DFHCSD, the CICS definitions list, use *userid*.PDTOOLS.REDBOOK.JCL(DEFPDPAK).

   Install the defined resources.

   Check that everything is here with transaction CEDA under CICS to display the contents of the PDPAK group as shown in Example 30-4.

Example 30-4 Result of CEDA DIS GROUP(PDPAK)

```
NAME     TYPE
COMPFILE FILE
CUSTFILE FILE
NEWTRAD  MAPSET
MYTRADS  PROGRAM
MYTRADD  PROGRAM
MYTRADI  PROGRAM
MYTRADMV PROGRAM
MYTRADMD PROGRAM
MYTD     TRANSACTION
TDB2     TRANSACTION
DBMYTDB2 DB2ENTRY
```

30.3 About the system configuration

You can follow along with each of the scenarios even if you do not install the application programs. However, if you do want to run the applications, you also need to have the appropriate system configuration.

This section briefly reviews the software that was installed on our system, and what you would need to run the applications on yours.

30.3.1 About the software prerequisites

The Trader application and the scenarios have been tested with the following software installed and configured:

- z/OS V1R5
- CICS Transaction Server V2.3 and CICS Transaction Server V3.1
- DB2 Universal Database V7.1 and DB2 Universal Database V8.1

The Application Development Tools for z/OS and OS/390:

- IBM Debug Tool Version 5, Release 1
- IBM Fault Analyzer for z/OS Version 5, Release 1
- IBM File Manager for z/OS Version 5, Release 1
- IBM Application Monitor Version 2, Release 1
- IBM Workload Simulator for z/OS and OS/390 Version 1, Release 1

It is possible that other levels of these software components may work, but the applications were tested with the levels listed here.

30.3.2 About the CICS configuration

A summary of the steps required to set up the CICS configuration follows:

- The CICS resource definitions are specified in *userid*.PDTOOLS.REDBOOK.JCL(PDPAK).
- An entry for the PDPAK group was added to the site's CICS definitions list, DFHCSD.
- After all of the batch compile jobs completed, the load modules for the CICS COBOL application programs, MYTRADMV, MYTRADMD, MYTRADMI, MYTRADS, MYTRADD and MYTRADI were in our CICS user load library, which is in the CICS DFHRPL.

30.3.3 About the DB2 configuration

A summary of the steps required to set up the DB2 configuration follows:

- The batch job *userid*.PDTOOLS.REDBOOK.JCL(BIND) creates a plan used by the CICS application.
- The plan name is MYTRADD, which is specified in the DB2 entry of the CICS definitions.
- The batch job *userid*.PDTOOLS.REDBOOK.JCL(GRANT) grants execution access to the plan MYTRADD.
- The batch job *userid*.PDTOOLS.REDBOOK.JCL(TABLES) defines the tables CUSTOMER_DETAILS and COMPANY_DETAILS.
- The batch job *userid*.PDTOOLS.REDBOOK.JCL(DATA) populates these tables.

30.3.4 Validate the installation

Follow the instructions in Chapter 31, “Guided tour of the Trader application” on page 817 to validate the installation. It presents the CICS/VSAM and batch/VSAM aspects of the application.

31

Guided tour of the Trader application

After you have established all of the components of the applications in your environment, you need to validate the installation.

The Trader application is used to maintain a stock portfolio held by an individual. This application enables you to:

- Obtain quotes
- Buy more shares of a company's stock
- Sell currently held shares of a company's stock

Note: This example was designed to demonstrate the capabilities of the Problem Determination and Deployment Tools. Therefore, a minimal amount of code was developed. This application does *not* represent real-world securities processing.

31.1 Getting started

This section tells you what must be running on the system to allow the applications to execute.

Before you can start the applications, the subsystems must be started. The subsystems required for Trader application are CICS and DB2.

31.2 Trader application in CICS

We present here the CICS/VSAM application. The panels and process are also valid for DB2 data.

31.2.1 Log on to the application

Perform the following steps to start the Trader application:

1. Verify that you have installed this application correctly.
2. Log on to your demonstration CICS region.
3. Enter the transaction `MYTD`.
4. Enter the user name `ERIC` and the password `xxxx` as shown in Figure 31-1.

```
                    Share Trading Demonstration                TRADER.T001

                    Share Trading Manager: Logon



              Enter your User Name:  ERIC


              Enter your Password:       _



--------------------------------------------------------------------------------
PF3=Exit                                                              PF12=Exit
                                                                        13/046
```

Figure 31-1 CICS Trader logon panel

Note: In the Trader application, navigation keys are displayed at the bottom of each screen. PF3 is used to go back to the previous screen (except on the Logon screen) and PF12 is used to terminate the application.

31.2.2 Running the Trader application

After the logon, the Company Selection panel is displayed, as shown Figure 31-2. Select a company you want to trade.

```
                    Share Trading Demonstration                    TRADER.T002

                 Share Trading Manager: Company Selection

                       1. Casey_Import_Export

                       2. Glass_and_Luget_plc

                       3. Headworth_Electrical

                       4. IBM



                Please select a company (1,2,3 or 4) :  _



--------------------------------------------------------------------------------
PF3=Return                                                            PF12=Exit
                                                                        17/060
```

Figure 31-2 CICS Trader list of companies

The Options panel is displayed, as shown in Figure 31-3.

```
                    Share Trading Demonstration                    TRADER.T003

                 Share Trading Manager: Options

                       1. New Real-Time Quote

                       2. Buy Shares

                       3. Sell Shares



                Please select an option (1,2 or 3):  1_



--------------------------------------------------------------------------------
PF3=Return                                                            PF12=Exit
                                                                        17/058
```

Figure 31-3 CICS Trader actions available for this company

On this panel, you select the trading option you want to perform:

- Obtain real-time quotes for a company.
- Buy additional shares of the company.

- Sell existing shares of the company.

Continue by selecting an option.

31.2.3 Obtain real-time quotes

Use the following steps to obtain real-time quotes:

1. Request a list of real-time quotes for the selected company by selecting option 1.
2. Share prices from the prior week are displayed including net present value as shown in Figure 31-4.

```
                          Share Trading Demonstration                TRADER.T004

                     Share Trading Manager: Real-Time Quote

      User Name:     ERIC

      Company Name:  Casey_Import_Export

      Share Values:                     Commission Cost:
         NOW:        00079.00              for Selling:        007
         1 week ago: 00059.00              for Buying:         010
         6 days ago: 00063.00
         5 days ago: 00065.00
         4 days ago: 00070.00
         3 days ago: 00072.00
         2 days ago: 00078.00           Number of Shares Held: 0005
         1 day ago:  00077.00           Value of Shares Held:  000000395.00



Request Completed OK
--------------------------------------------------------------------------------
PF3=Return                                                            PF12=Exit
                                                                        01/001
```

Figure 31-4 CICS Trader quotes

This screen displays the price of the company's shares over the past seven days, the number of shares held, and the value of those shares based on the current day's price. Data is retrieved from:

- The company's share price is read from the VSAM file COMPFILE (*userid*.PDTOOLS.REDBOOK.COMPFILE).
- The details of the user's portfolio, (such as the number of shares held), are read from the VSAM file CUSTFILE (*userid*.PDTOOLS.REDBOOK.CUSTFILE).

Press PF3 to return to the Options panel.

31.2.4 Buy shares

Use the following steps to buy shares:

1. Request to buy shares of the selected company by selecting option 2.
2. Enter the number of shares to purchase, as shown in Figure 31-5.

```
                    Share Trading Demonstration                    TRADER.T005

                    Share Trading Manager: Shares - Buy


                User Name:     ERIC

                Company Name:  Casey_Import_Export


                Number of Shares to Buy:  15_


--------------------------------------------------------------------------------
PF3=Return                                                            PF12=Exit
                                                                        13/046
```

Figure 31-5 CICS Trader buy shares

3. A confirmation message is issued.

   The Options panel is re-displayed with a message in the bottom, left-hand corner indicating the status of the transaction.

   `Request Completed OK`

   If the process is successful, the value of the number of shares held is updated in the CUSTFILE.

31.2.5 Sell shares

Use the following steps to sell shares:

1. Request to sell shares of the selected company by selecting option 3.
2. Enter the number of shares to sell as shown in Figure 31-6.

```
                    Share Trading Demonstration                    TRADER.T005

                 Share Trading Manager: Shares - Sell


            User Name:     ERIC

            Company Name:  Casey_Import_Export


            Number of Shares to Sell: 10_




--------------------------------------------------------------------------------
PF3=Return                                                          PF12=Exit
                                                                       13/046
```

Figure 31-6 CICS Trader sell shares

3. A confirmation message is issued.

   The Options panel is re-displayed with a message in the bottom, left-hand corner indicating the status of the transaction.

   `Request Completed OK`

   If the process is successful, the value of the number of shares held is updated in the CUSTFILE.

31.3 Running the Trader application in batch

Use the following steps to run the Trader application in batch:

1. Verify that you have installed this application correctly.
2. Create a transaction file that contains sample records to buy, sell, and list shares in one company.

You can use the one in *userid*.PDTOOLS.REDBOOK.SAMPLES(TRANFILE).

31.3.1 Run the batch job

Run the Trader application as follows:

- Submit the batch job TRADER, in *userid*.PDTOOLS.REDBOOK.JCL (Example 31-1).

Example 31-1 TRADER batch job

```
//TRADERBA JOB 1,RACFUSER,TIME=1440,NOTIFY=&SYSUID,REGION=4M,
//             CLASS=A,MSGCLASS=X,MSGLEVEL=(1,1)
//*****************************************************************
//GO       EXEC PGM=TRADERB,PARM=RPTOPTS(YES)
//STEPLIB  DD DISP=SHR,DSN=userid.PDTOOLS.REDBOOK.LOAD
//SYSPRINT DD SYSOUT=*
```

```
//SYSABEND DD SYSOUT=*
//COMPFILE DD DISP=SHR,DSN=userid.PDTOOLS.REDBOOK.COMPFILE
//CUSTFILE DD DISP=SHR,DSN=userid.PDTOOLS.REDBOOK.CUSTFILE
//TRANSACT DD DISP=SHR,DSN=userid.PDTOOLS.REDBOOK.SAMPLES(TRANFILE)
//REPOUT   DD SYSOUT=*
//TRANREP  DD SYSOUT=*
//*
```

- This job invokes the program, TRADERB, which reads the transaction file. The contents of the file dictate the actions of the program, as reported in Example 31-2.

Example 31-2 TRANFILE sample

```
********************************** Top of Data **********************************
12345RB_DEMO                       .IBM                BUY_SELL       00301
12345RB_DEMO                       .Casey_Import_Export BUY_SELL      00251
12345RB_DEMO                                           SHARE_VALUE
******************************** Bottom of Data *******************************
```

- Each record is validated against the company file. If a BUY or a SELL request is found, the appropriate program logic is invoked. The customer file is updated as a result.

31.3.2 Check the result

All reports are written to the JES spool.

The share trading activity from this batch job is reported as shown in Example 31-3 and Example 31-4.

Example 31-3 REPOUT

```
Casey_Import_Export
CUSTOMER : ERIC                                          11/16/2004
-----------------------------------------------------------------
     COMPANY                  SHARES    SHARE            TOTAL
                              HELD      VALUE            COST
-----------------------------------------------------------------
     Casey_Import_Export       102        79.00         8,058.00
     IBM                        70       163.00        11,410.00
     Veck_Transport             25        36.00           900.00
```

Example 31-4 TRANREP

```
-----------------------------------------------------------------------------
CUSTOMER                COMPANY                QTY REQ-TYP  STATUS
-----------------------------------------------------------------------------
ERIC                    IBM                    30  BUY      PROCESSED SUCCESFULLY
ERIC                    Casey_Import_Export    25  BUY      PROCESSED SUCCESFULLY
```

This completes the batch/VSAM application. The results are also valid for DB2.

32

Scenario 1: Invalid VSAM data generating an abend

In this chapter we describe the application components that exist in the CICS environment on our system and show how they are set up.

We force the application to abend, describe the steps needed to identify the cause of an abend in the application using Fault Analyzer and Debug Tool, and explain how to manipulate the data to correct the problem using File Manager.

32.1 Set up the components

Two types of components need to be established for this scenario:

- CICS components
- Program products:
  - Debug Tool
  - Fault Analyzer
  - File Manager

32.1.1 CICS components

The components used by the Trader application are listed in Table 32-1.

Table 32-1 CICS components of the Trader application for the VSAM scenario

Component	Details	Remarks
Programs	MYTRADMV MYTRADS	CICS COBOL programs
Tran ID	MYTD	Transaction associated with the program MYTRADMV
Mapset	NEWTRAD	BMS mapset containing all the maps used by the application
Files	CHABERT.TRADER.CUSFILE CHABERT.TRADER.COMPFILE	VSAM files used by the application
Copybooks	CUSTFILE COMPFILE	File definition for CUSTFILE and COMPFILE

32.1.2 Program products

To use the Problem Determination Tools with this scenario, make sure you have the following output or supporting files for each product:

Debug Tool

Ensure that Debug Tool is correctly installed in your CICS region.

Fault Analyzer

Ensure that Fault Analyzer is correctly installed in your CICS region.

- As an example, the transaction CFA on C22F returns the panel shown in Figure 32-1.

```
                    Fault Analyzer Control Transaction

Option: F=Force purge active task

_ Number of ACTIVE analysis tasks   0000
  Number of WAITING analysis tasks  0000

Options: I=Install U=Uninstall

                   Current Status/Error Message
_ XPCABND          Installed
_ XDUREQ           Installed
_ LE Exit          Installed
_ SDUMP Screening  Installed









PF1=Help  PF3=Exit   PF9=IVP   ENTER=Update
                                                              05/002
```

Figure 32-1 CFA Transaction

You must have a compiler listing or side file for the programs MYTRADMV and MYTRADS.

If you are not using the supplied JCL to compile these programs, make sure you specify the following compiler options:

```
LIST,SOURCE,XREF,MAP
```

File Manager

You need the following copybooks that contain the record structure of the VSAM files CHABERT.TRADER.CUSTFILE and CHABERT.TRADER.COMPFILE:

- Copybooks CUSTFILE and COMPFILE in CHABERT.TRADER.COPYLIB

Make sure you run the DEFVSAM1 batch job to load the VSAM files.

32.2 Tracking an abend in the application

To demonstrate the capabilities of the Problem Determination Tools, this section describes how to force the application to abend and then step through the process of fixing it. Debug Tool is used to follow the execution of the application; Fault Analyzer is used to identify the cause of the abend; File Manager is used to correct the error in a VSAM application file.

- Access the Trader application by selecting:
  - MYTD transaction
  - Userid = TSDEMO
  - password = itso
  - 4 = IBM
  - 1 = New Real-time quote

In this example, you attempt to obtain the real-time quotes of IBM for the customer, TSDEMO. After you select option 1 on the Options menu of the CICS Trader application and press Enter, the application abends. The CICS-issued transaction abend message is shown in Figure 32-2.

```
                    Share Trading Demonstration                    TRADER.T003

                    Share Trading Manager: Options


                     1. New Real-Time Quote

                     2. Buy Shares

                     3. Sell Shares




                Please select an option (1,2 or 3):  1



DFHAC2206 20:06:36 CICSC22F Transaction MYTD failed with abend ASRA. Updates
 to local recoverable resources backed out.
                                                                      01/001
```

Figure 32-2 Abend ASRA in MYTD

32.2.1 Viewing the abend in Fault Analyzer

As you can see in Figure 32-3, Fault ID F00017 contains the abending program MYTRADS with the abend code ASRA. Make an Interactive reanalysis by entering an **I** in front of the Fault ID and pressing Enter.

```
  File  Options  View  Services  Help
 ____________________________________________________________________________
 IBM Fault Analyzer - Fault Entry List                        Line 1 Col 1 80
 Command ===> ____________________________________________ Scroll ===> CSR

 Fault History File or View  : 'IDI.HIST.CICSC22F'

 {The following line commands are available: ? (Query), V (View real-time
 report), I (Interactive reanalysis), B (Batch reanalysis), D (Delete).}

    Fault_ID Program  Dup_Count Job/Tran User_ID  Sys/Job  Abend Date       Tim
 I_   F00017 MYTRADS          0 MYTD     CHABERT  CICSC22F ASRA  2004/11/25 20:
 __   F00009 MYTRADMD         0 TDB2     KDFORD   CICSC22F AEIS  2004/11/19 09:
 __   F00001 DFHZARQ          1 TRAR     CONOVER  CICSC22F AZI4  2004/11/09 13:

 *** Bottom of data.

                                                                      12/003
```

Figure 32-3 Interactive Reanalysis

After the Interactive reanalysis has finished, the Interactive Reanalysis Report is displayed as shown in Figure 32-4.

```
  File  View  Services  Help
 ____________________________________________________________________________
Interactive Reanalysis Report                                   Line 1 Col 1 80
Command ===> ____________________________________________ Scroll ===> CSR
TRANID: MYTD      CICS ABEND: ASRA              STLABF6   2004/11/25  20:06:32

Fault Summary:
Module MYTRADS, program MYTRADS, source line # 783 : CICS abend ASRA.

Select one of the following options and press Enter to access further fault
information:
_  1.  Synopsis
   2.  Event Summary
   3.  System-Wide Information
   4.  Abend Job Information
   5.  Options in Effect

{Fault Analyzer maximum storage allocated: 2.34 megabytes.}

*** Bottom of data.

                                                                      12/002
```

Figure 32-4 Interactive Reanalysis Report

Here you can see that the abend occurred at line 783. You will use this information when you involve Debug Tool. To look for more details, choose option **1** to see the Synopsis as shown in Figure 32-5.

```
  File  View  Services  Help
 ____________________________________________________________________________
Synopsis                                                        Line 13 Col 1 80
Command ===> ____________________________________________ Scroll ===> CSR
TRANID: MYTD      CICS ABEND: ASRA              STLABF6   2004/11/26  11:09:34
  Source
  Line #
  000802            MULTIPLY DECIMAL-SHARE-VALUE BY DEC-NO-SHARES
  000803              GIVING DECIMAL-SHARE-VALUE

The COBOL source code for data fields involved in the failure:

  Source
  Line #
  000115            03 DEC-NO-SHARES             PIC S9(7) COMP-3.
  000220                 07 DECIMAL-SHARE-VALUE PIC 9(11)V99.

Data field values at time of abend:

  DEC-NO-SHARES         = X'F0F2F0F0'  *** Cause of error ***
  DECIMAL-SHARE-VALUE   = 163.00

*** Bottom of data.

                                                                      04/015
```

Figure 32-5 Interactive Reanalysis Synopsis

The synopsis shows:

- The source line where the abend occurred
- The variables involved and their values at abend

You can see that DEC-NO-SHARES contains invalid data. To obtain more information about the variable, place the cursor on line 000115 as shown in Figure 32-6 and press Enter.

```
  File  View  Services  Help
 ------------------------------------------------------------------------------
Program MYTRADS Compiler Listing                                Line 181 Col 1 80
Command ===> ____________________________________________________ Scroll ===> CSR
TRANID: MYTD  _   CICS ABEND: ASRA                 STLABF6   2004/11/26  09:13:29
000106            03 UPDATE-BUY-SELL              PIC X(1).
000107            03 FILLER                       PIC X(95).
000108         01 CUSTOMER-IO-BUFFER.
000109            COPY CUSTFILE.
000110  *******01 CUSTOMER-IO-BUFFER.
000111            03 KEYREC.
000112               05 CUST-NM                    PIC X(60).
000113               05 KEYREC-DOT                 PIC X(1).
000114               05 COMP-NM                    PIC X(20).
000115            03 DEC-NO-SHARES                 PIC S9(7) COMP-3.
000116            03 BUY-FROM                      PIC X(8).
000117            03 BUY-FROM-NO                   PIC X(4).
000118            03 BUY-TO                        PIC X(8).
000119            03 BUY-TO-NO                     PIC X(4).
000120            03 SELL-FROM                     PIC X(8).
000121            03 SELL-FROM-NO                  PIC X(4).
000122            03 SELL-TO                       PIC X(8).
000123            03 SELL-TO-NO                    PIC X(4).
000124            03 ALARM-PERCENT                 PIC X(3).
                                                                        05/016
```

Figure 32-6 Complier listing

The resulting message is:

```
DEC-NO-SHARES is a part of CUSTOMER-IO-BUFFER.
```

So now you have the question:

- Is the bad data coming from the file that has been read?

  or

- Is it the program that put the wrong data in the field?

We discuss how to answer these questions in 32.2.2, “Using Debug Tool to analyze the problem” on page 834.

But before that, find the actual record in the dump.

How to find the record in the dump

From the Interactive Analysis Report select option **2** (Event Summary) and the panel shown in Figure 32-7 is returned.

```
  File  View  Services  Help
 ------------------------------------------------------------------------
Event Summary                                                Line 1 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR
TRANID: MYTD      CICS ABEND: ASRA              STLABF6   2004/11/26  09:13:29

{The following events are presented in chronological order.}

Event           Fail  Module   Program  EP
#  Type         Point Name     Name     Name     Event Location (*)     Loaded
 1 Call               DFHAPLI  DFHAPLI1 n/a      P+29D6                 CICSVS
 2 Call               CEEPLPKA n/a      CEECRINI E+B2A                  CEE.SC
 3 Call               CEEPLPKA n/a      CEECRINV E+68E                  CEE.SC
 4 Call               CEEEV005 n/a      IGZCEV5  E+65E                  CEE.SC
 5 Link               MYTRADMV MYTRADMV MYTRADMV L#1253 P+23D2          CHABER
 6 Call               DFHAPLI  DFHAPLI1 n/a      P+27B2                 CICSVS
 7 Call               CEEPLPKA n/a      CEECRINI E+B2A                  CEE.SC
 8 Call               CEEPLPKA n/a      CEECRINV E+68E                  CEE.SC
 9 Call               CEEEV005 n/a      IGZCEV5  E+65E                  CEE.SC
10 Abend ASRA   ***** MYTRADS  MYTRADS  MYTRADS  L#802 P+1D16           CHABER


(*) One or more of the following abbreviations might appear in the "Event
    Location" column:
                                                                        04/015
```

Figure 32-7 Event Summary

Event number 10 is your point of failure; select it and press Enter to view the panel shown in Figure 32-8.

```
  File  View  Services  Help
 ------------------------------------------------------------------------
Event 10 of 10: Abend ASRA *** Point of Failure ***          Line 1 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR
TRANID: MYTD      CICS ABEND: ASRA              STLABF6   2004/11/26  09:13:29

Previous Event Details


CICS Abend Code . . . . . . : ASRA
Program Interruption Code . : 0007 (Data Exception)
  A decimal digit or sign was invalid.

NOTE: The following source code information for program MYTRADS was obtained
      from a compiler listing or side-file that might not match the current
      load module.
      Program MYTRADS was compiled on 2004/11/25 at 22:20:29. The program
      listing for MYTRADS was created on 2004/11/25 at 22:23:47.

COBOL Source Code:
  Source
  Line #
  000802            MULTIPLY DECIMAL-SHARE-VALUE BY DEC-NO-SHARES
  000803              GIVING DECIMAL-SHARE-VALUE
                                                                        04/015
```

Figure 32-8 Point of failure

Type `M` on the command line and press PF8 to scroll max down to get the link to the associated data areas, where you can see the Working-Storage Section with its values, as shown in Figure 32-9.

```
  File  View  Services  Help
 ------------------------------------------------------------------------------
Event 10 of 10: Abend ASRA *** Point of Failure ***          Line 102 Col 1 80
Command ===> ____________________________________________________ Scroll ===> CSR
TRANID: MYTD      CICS ABEND: ASRA              STLABF6   2004/11/26  09:13:29
  Continue Receiving Data . : No                      (EIBRECV)
  Attach Header Data in RU. : No                      (EIBATT)
  RU Indicates End-of-Chain : No                      (EIBEOC)
  User Data Contains FMH. . : No                      (EIBFMH)
  Data is Complete. . . . . : No                      (EIBCOMPL)
  Signal Received . . . . . : No                      (EIBSIG)
  CONFIRM Request Received. : No                      (EIBCONF)
  Error Code Received . . . : No                      (EIBERRCD)
  SYNCPOINT ROLLBACK Req'd. : No                      (EIBSYNRB)
  No Data Sent. . . . . . . : No                      (EIBNODAT)
  Rollback. . . . . . . . . : No                      (EIBRLDBK)

Associated Messages

CEE3207S Job-specific text not available

Associated Storage Areas

*** Bottom of data.
                                                                        22/025
```

Figure 32-9 Point of failure: End of report

Place the cursor on the highlighted area and press Enter. The panel shown in Figure 32-10 is displayed.

```
  File  View  Services  Help
 ------------------------------------------------------------------------------
Associated Storage Areas                                     Line 1 Col 1 80
Command ===> ____________________________________________________ Scroll ===> CSR
TRANID: MYTD      CICS ABEND: ASRA              STLABF6   2004/11/26  09:13:29

Task Global Table (TGT) at address 12EA2AA0 for length 376


WORKING-STORAGE SECTION
 Off Hex Value                            EBCDIC Value       Source (Starting a
BLW=0000 at address 12EA2CE0
                                                              01 WRITEQ-WORDS.
   0 F0F97AF1 F37AF2F7                    *09:13:27        *     03 TIME-TRACE
   8 40D4E8E3 D9C1C4E2 7A                 * MYTRADS:       *     03 PROGRAM-TRA
  11 C2E4C9D3 C4C9D5C7 40D9C5E3 E4D9D540 *BUILDING RETURN *     03 COMMENT-FIE
  21 C3D6D4D4 C1D9C5C1 40404040 40404040 *COMMAREA        *
  31 40404040 40404040 40404040 40404040 *                *
  41 4040                                *                *
  48 E4E2C5D9 40E3E2C4 C5D4D640 C3D6D4D7 *USER TSDEMO COMP*  01 WORK-FIELD PIC
  58 C1D5E840 C9C2D440 40404040 40404040 *ANY IBM         *
  68 40404040 40404040 40404040 40404040 *                *
  78 4040                                *                *
  80 00000000 00000000 00000000 00000000 *................*  01 DEBUG-WORDS PI
                                                                        11/043
```

Figure 32-10 Associated Storage Areas

Scroll down using PF8 or issue the **Find** command to get to the area where the current record is located in the Working-Storage Section as shown in Figure 32-11.

```
  File  View  Services  Help
 ------------------------------------------------------------------------------
Associated Storage Areas                                      Line 107 Col 42 121
Command ===> ____________________________________________________ Scroll ===> CSR
TRANID: MYTD      CICS ABEND: ASRA              STLABF6   2004/11/26  11:09:34

*                 *
                     01 CUSTOMER-IO-BUFFER.
                        03 KEYREC.
*TSDEMO           *        05 CUST-NM                PIC X(60).
*                 *

*                 *
*.                *        05 KEYREC-DOT             PIC X(1).
*IBM              *        05 COMP-NM                PIC X(20).
*                 *
*0200             *     03 DEC-NO-SHARES             PIC S9(7) COMP-3.
*                 *     03 BUY-FROM                  PIC X(8).
*                 *     03 BUY-FROM-NO               PIC X(4).
*00110.00         *     03 BUY-TO                    PIC X(8).
*0100             *     03 BUY-TO-NO                 PIC X(4).
*00180.00         *     03 SELL-FROM                 PIC X(8).
*0200             *     03 SELL-FROM-NO              PIC X(4).
*00170.00         *     03 SELL-TO                   PIC X(8).
                                                                        06/029
```

Figure 32-11 Associated Storage Areas - CUSTOMER-IO-BUFFER (EBCDIC)

As you can see, the record key is:

- CUST-NM = TSDEMO
- KEYREC-DOT =.
- COMP-NM = IBM

You also can see the field in error DEC-NO-SHARES, which has a value of 0200. *Remember that this is shown in EBCDIC*. By scrolling to the right with PF10 you will see the HEX value of the DEC-NO-SHARES as shown in Figure 32-12.

```
  File  View  Services  Help
 ------------------------------------------------------------------------------
Associated Storage Areas                                       Line 107 Col 6 85
Command ===> ____________________________________________________ Scroll ===> CSR
TRANID: MYTD      CICS ABEND: ASRA              STLABF6   2004/11/26  11:09:34
    Lines 3B5-3F4 same as above
40404040 40404040 40404040 404040   *                *
                                                        01 CUSTOMER-IO-BUFFER.
                                                           03 KEYREC.
E3E2C4C5 D4D64040 40404040 40404040 *TSDEMO          *        05 CUST-NM
40404040 40404040 40404040 40404040 *                *
    Lines 428-437 same as above
40404040 40404040 40404040          *                *
4B                                  *.               *        05 KEYREC-DOT
C9C2D440 40404040 40404040 40404040 *IBM             *        05 COMP-NM
40404040                            *                *
F0F2F0F0                            *0200            *     03 DEC-NO-SHARES
40404040 40404040                   *                *     03 BUY-FROM
40404040                            *                *     03 BUY-FROM-NO
F0F0F1F1 F04BF0F0                   *00110.00        *     03 BUY-TO
F0F1F0F0                            *0100            *     03 BUY-TO-NO
F0F0F1F8 F04BF0F0                   *00180.00        *     03 SELL-FROM
F0F2F0F0                            *0200            *     03 SELL-FROM-NO
F0F0F1F7 F04BF0F0                   *00170.00        *     03 SELL-TO
                                                                        07/002
```

Figure 32-12 Associated Storage Areas - CUSTOMER-IO-BUFFER (Hexa)

In HEX the value is F0F2F0F0 and that is an invalid value for a COMP-3 field. This is the reason for the abend.

What we know about the abend so far

Here is what we know:

- The abending program line number is 806.
- Fields involved in abend are DEC-NO-SHARES and DECIMAL-SHARE-VALUE.
- Field in error DEC-NO-SHARES with a value of F0F2F0F0.
- The record that contains the field in error.

Here is what we do not know:

- Is the bad data coming from the file that has been read?

  or

- Is it the program that puts the wrong data in the field?

32.2.2 Using Debug Tool to analyze the problem

The first step for debugging the abending program is to define it to Debug Tool using the CICS transaction DTCN. As shown in Figure 32-13, define the program MYTRADS and the userid CHABERT to Debug Tool, so it is just when the userid CHABERT runs the program MYTRADS that Debug Tool will be involved. All other parameters on the screen are set at default by Debug Tool. After entering the values, press PF4 to update the debug profile as shown in Figure 32-13.

```
DTCN                Debug Tool CICS Control - Primary Menu            CICSC22F

Select the combination of resources to debug (see Help for more information)

 Terminal Id      ==> 0001
 Transaction Id   ==>
 Program Id(s)    ==> MYTRADS  ==>          ==>          ==>
                  ==>          ==>          ==>          ==>
 User Id          ==> CHABERT
 NetName          ==>

Select type and ID of debug display device

 Session Type      ==> MFI                 MFI, TCP
 Port Number       ==>                     TCP Port
 Display Id        ==> 0001
 Connection Type   ==> SINGLE              Single, Multiple

Generated String:  TEST(ALL,'*',PROMPT,'MFI%0001:CHABERT.DEBUG.PREF(TRAD1)')

Repository String: TEST(ALL,'*',PROMPT,'MFI%0001:CHABERT.DEBUG.PREF(TRAD1)')


PF1=HELP 2=GHELP 3=EXIT 4=SAVE 6=DELETE 7=SHOW 9=OPTIONS
                                                                      05/023
```

Figure 32-13 Debug Tool CICS Control: Primary Menu

Access the Trader application by making the following selections: transaction `MYTD`; userid `TSDEMO`; password `itso`; option **4** IBM; and option **1** New Real-Time Quote, as shown in Figure 32-14.

```
                         Share Trading Demonstration                TRADER.T003

                         Share Trading Manager: Options


                          1. New Real-Time Quote

                          2. Buy Shares

                          3. Sell Shares




                   Please select an option (1,2 or 3):  1_



--------------------------------------------------------------------------------
PF3=Return                                                              PF12=Exit
                                                                          17/058
```

Figure 32-14 Share Trading Manager: Options

Debug Tool will now be activated for program MYTRADS as shown in Figure 32-15.

What you want to see is the value of the variable DEC-NO-SHARES:

- Before the record has been read
- After the record is has been read

To do this, define a pause at 543 as shown in Figure 32-13 on page 834. This is the pause before reading the record.

```
COBOL    LOCATION: MYTRADS ENTRY
Command ===> AT 543_                                          Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 2
****************************** TOP OF MONITOR ********************************
0001  1 EQA1252I ********** AUTOMONITOR **********
0002    EQA1945I There are no variables in the statement to display.
****************************** BOTTOM OF MONITOR *****************************



SOURCE: MYTRADS --1----+----2----+----3----+----4----+----5--- LINE: 29 OF 1197
      29         PROGRAM-ID. MYTRADS.                                         .
      30         ENVIRONMENT DIVISION.                                        .
      31         CONFIGURATION SECTION.                                       .
      32         SOURCE-COMPUTER. IBM-PC.                                     .
      33         OBJECT-COMPUTER. IBM-PC.                                     .
      34        ***************************************************************.
      35         DATA DIVISION.                                               .
      36         WORKING-STORAGE SECTION.                                     .
      37        * PUT IN FOR WRITEQ                                           .
LOG 0----+----1----+----2----+----3----+----4----+----5----+---- LINE: 94 OF 94
PF  1:PK13-24    2:KEEP       3:QUIT       4:CURRLINE   5:FIND       6:SETBREAK
PF  7:UP         8:DOWN       9:GO 1      10:ZOOM WIN  11:ZOOM LOG  12:GO
                                                                        02/021
```

Figure 32-15 COBOL Location

To list the value of DEC-NO-SHARES in the monitor window give the command:

```
MONITOR LIST DEC-NO-SHARES
```

The results are shown in Figure 32-16.

```
COBOL    LOCATION: MYTRADS :> 320.1
Command ===> _                                                Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 1
0001  1 DEC-NO-SHARES   X'00000000'
***************************** BOTTOM OF MONITOR ******************************


SOURCE: MYTRADS --1----+----2----+----3----+----4----+----5-- LINE: 318 OF 1197
     318        ***************************************************************.
     319         MAINLINE SECTION.                                             .
     320             MOVE 'Entry' TO COMMENT-FIELD                             0
     321             PERFORM WRITEQ-TS                                         0
     322             MOVE DFHCOMMAREA TO COMMAREA-BUFFER                       0
     323             MOVE USER-TRACE-MSG TO COMMENT-FIELD                      0
     324             PERFORM WRITEQ-TS                                         0
     325             EVALUATE REQUEST-TYPE                                     0
     326                WHEN SHARE-VALUE-REQ                                   .
LOG 0----+----1----+----2----+----3----+----4----+----5----+-- LINE: 101 OF 105
0101 EQA1046I The STATEMENT MYTRADS ::> MYTRADS :> 543.1 breakpoint is
0102          replaced.
0103  MONITOR
0104    LIST DEC-NO-SHARES ;
PF  1:PK13-24    2:KEEP       3:QUIT       4:CURRLINE   5:FIND       6:SETBREAK
PF  7:UP         8:DOWN       9:GO 1      10:ZOOM WIN  11:ZOOM LOG  12:GO
                                                                     02/016
```

Figure 32-16 COBOL Monitor List

Press GO. (In our example this is PF12 because of our preference file. The default key is PF9). This instructs Debug Tool to execute the program, which will now execute to line 543 where we set the pause. This is shown in Figure 32-17. The value of DEC-NO-SHARES is shown in the monitor window. As you can see, the value is X'00000000'. This is the value before the record has been read.

```
COBOL    LOCATION: MYTRADS :> 543.1
Command ===> _                                                Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 1
0001  1 DEC-NO-SHARES   X'00000000'
***************************** BOTTOM OF MONITOR ******************************


SOURCE: MYTRADS --1----+----2----+----3----+----4----+----5-- LINE: 541 OF 1197
     541        ***************************************************************.
     542         READ-CUSTFILE SECTION.                                        .
     543             MOVE 'READING RECORD FROM CUSTOMER FILE' TO COMMENT-FIELD 0
     544             PERFORM WRITEQ-TS                                         0
     545        * Build record key                                             .
     546             MOVE USERID TO CUST-NM  OF CUSTOMER-IO-BUFFER             0
     547             MOVE '.' TO KEYREC-DOT                                    0
     548             MOVE COMPANY-NAME TO COMP-NM OF CUSTOMER-IO-BUFFER        0
     549        *EXEC CICS READ                                                .
LOG 0----+----1----+----2----+----3----+----4----+----5----+-- LINE: 103 OF 106
0103  MONITOR
0104    LIST DEC-NO-SHARES ;
0105  SET AUTOMONITOR OFF ;
0106  GO ;
PF  1:PK13-24    2:KEEP       3:QUIT       4:CURRLINE   5:FIND       6:SETBREAK
PF  7:UP         8:DOWN       9:GO 1      10:ZOOM WIN  11:ZOOM LOG  12:GO
                                                                     02/015
```

Figure 32-17 DEC-NO-SHARES value at line 543

Set the second pause to get the value of DEC-NO-RECORDS after the record has been read. This is done by giving the command **AT 563**, as shown in Figure 32-18.

```
COBOL    LOCATION: MYTRADS :> 543.1
Command ===> AT 563_                                          Scroll ===> CSR
SOURCE: MYTRADS --1----+----2----+----3----+----4----+----5-- LINE: 549 OF 1197
     549        *EXEC CICS READ                                              .
     550        *          FILE('CUSTFILE')                                  .
     551        *          INTO(CUSTOMER-IO-BUFFER)                          .
     552        *          LENGTH(LENGTH OF CUSTOMER-IO-BUFFER)              .
     553        *          RIDFLD(KEYREC)                                    .
     554        *          NOHANDLE                                          .
     555        *          END-EXEC                                          .
     556             Move LENGTH OF CUSTOMER-IO-BUFFER to dfhb0020           0
     557             Call 'DFHEI1' using by content x'0602f0002700008000f0f0f4 0
     558        -    '404040' by content 'CUSTFILE' by reference             .
     559             CUSTOMER-IO-BUFFER by reference dfhb0020 by reference KEY .
     560             end-call                                                .
     561                                                                     .
     562                                                                     .
     563             MOVE 'READ' TO CICS-FUNCTION                            0
     564             PERFORM TRACE-CICS-ERROR                                0
     565             EVALUATE EIBRESP                                        0
     566        * DFHRESP(NORMAL) = 0                     INSERTED BY TRANSL .
     567                 WHEN 0                                              .
PF  1:PK13-24    2:KEEP       3:QUIT       4:CURRLINE   5:FIND       6:SETBREAK
PF  7:UP         8:DOWN       9:GO 1      10:ZOOM WIN  11:ZOOM LOG  12:GO
                                                                      02/021
```

Figure 32-18 Pause at line 563

Press GO again; the program will execute to this line and you will see the value of DEC-NO-SHARES after the record has been read, as shown in Figure 32-19. The value is X'F0F2F0F0' which is the value you saw in the dump.

In the next section we describe how to use File Manager to fix the incorrect value.

```
COBOL    LOCATION: MYTRADS :> 563.1
Command ===> _                                                Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 1
0001  1 DEC-NO-SHARES   X'F0F2F0F0'
***************************** BOTTOM OF MONITOR ******************************


SOURCE: MYTRADS --1----+----2----+----3----+----4----+----5-- LINE: 561 OF 1197
     561                                                                     .
     562                                                                     .
     563             MOVE 'READ' TO CICS-FUNCTION                            0
     564             PERFORM TRACE-CICS-ERROR                                0
     565             EVALUATE EIBRESP                                        0
     566        * DFHRESP(NORMAL) = 0                     INSERTED BY TRANSL .
     567                 WHEN 0                                              .
     568                      MOVE CLEAN-RETURN TO RETURN-VALUE              0
     569        * DFHRESP(NOTFND) = 13                    INSERTED BY TRANSL .
LOG 0----+----1----+----2----+----3----+----4----+----5----+-- LINE: 108 OF 111
0108 EQA1278I  SCROLL UP
0109 EQA1806E The command element AT is invalid.
0110  AT 563 ;
0111  GO ;
PF  1:PK13-24    2:KEEP       3:QUIT       4:CURRLINE   5:FIND       6:SETBREAK
PF  7:UP         8:DOWN       9:GO 1      10:ZOOM WIN  11:ZOOM LOG  12:GO
                                                                      02/015
```

Figure 32-19 DEC-NO-SHARES value at line 563

32.2.3 Using File Manager to correct a problem with data

File Manager is used to correct the invalid data in the application file.

> **Note:** You must close the CUSTFILE in the CICS region before you attempt to edit it. If you do not, File Manager will display the following error message when you edit the file:
>
> `VSAM OPEN RC X'08', Error Code X'A8`
>
> Call the CEMT transaction `CEMT Set File (CUSTFILE) CLO ENA` and press Enter.

Access File Manager in your ISPF session. Select option **2**, Edit, and enter the VSAM data set name, copybook data set name, and starting position `TSDEMO` as shown in Figure 32-20. Press Enter.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 File Manager                    Edit Entry Panel
 Command ===> _______________________________________________________________

 Input Partitioned, Sequential or VSAM Data Set:
    Data set name  . TRADER.CUSTFILE                          ____________
    Member . . . . . ________      Blank or pattern for member list
    Volume serial  . ______        If not cataloged
    Start position . TSDEMO                              +
    Record limit . . ________

 Copybook or Template:
    Data set name  . TRADER.COPYLIB                           ____________
    Member . . . . . CUSTFILE __   Blank or pattern for member list

 Processing Options:
  Copybook/template   Start position type   Enter "/" to select option
  1  1. Above         _  1. Key             _  Edit template
     2. Previous         2. RBA             /  Set top to Start position
     3. None             3. Slot            _  Create audit trail
     4. Create dynamic   4. Record number

                                                                     04/015
```

Figure 32-20 File Manager - Edit Entry Panel

The panel shown in Figure 32-21 appears, displaying records containing TSDEMO as the value of the CUST-NM.

```
 Process   Options   Help
File Manager Edit CHABERT.TRADER.CUSTFILE
Command ===> _____________________________________________________ Scroll CSR
       Type KSDS     Refresh on save N__                           Format TABL
       CUST-NM                                                      KEYREC-DOT
       #3                                                           #4
       AN 1:60                                                      AN 61:1
       <---+----1----+----2----+----3----+----4----+----5----+----> -
000000 ****  Key used to set start point  ****
000001 TSDEMO                                                       .
000002 TSDEMO                                                       .
000003 TSDEMO                                                       .
000004 TSDEMO                                                       .
000005 ****  End of data  ****
                                                                 04/015
```

Figure 32-21 File Manager: CUSTFILE (Edit 1)

Press PF11 to scroll to the right, finding the invalid data. File Manager displays the invalid data as a string of highlighted asterisks as shown in Figure 32-22.

```
 Process   Options   Help
File Manager Edit CHABERT.TRADER.CUSTFILE
Command ===> _____________________________________________________ Scroll CSR
       Type KSDS     Refresh on save N__                           Format TABL
       COMP-NM              DEC-NO-SHARES BUY-FROM BUY-FROM-NO BUY-TO   BUY-TO
       #5                             #6 #7       #8          #9       #10
       AN 62:20                    PD 82:4 AN 86:8  AN 94:4     AN 98:8  AN 106
       <---+----1----+---->       <---+--> <---+--> <-->        <---+--> <-->
000000 ****  Key used to set start point  ****
000001 Casey_Import_Export        ******** 00080.00 0100        00050.00 0150
000002 IBM                        ********                      00110.00 0100
000003 ShareSelect                ********
000004 SportSelect                ********
000005 ****  End of data  ****
                                                                 04/015
```

Figure 32-22 File Manager: CUSTFILE (Edit 2)

Place the cursor on the record containing the invalid data (remember that the record had the key TSDEMO.IBM) and press PF2 to display the record in SNGL format as shown in Figure 32-23. (You could also change the record in Figure 32-22. The last few steps are just to show the action being performed in SNGL mode.)

```
 Process   Options   Help
File Manager Edit CHABERT.TRADER.CUSTFILE
Command ===> _____________________________________________________ Scroll CSR
       Type KSDS                                                Zoom Format SNGL
                                  Top Line is 6    of 16    in Record 1
Field                            Data
  3 COMP-NM                      Casey_Import_Export
 2 DEC-NO-SHARES                 ********
 2 BUY-FROM                      00080.00
 2 BUY-FROM-NO                   0100
 2 BUY-TO                        00050.00
 2 BUY-TO-NO                     0150
 2 SELL-FROM                     00098.00
 2 SELL-FROM-NO                  0200
 2 SELL-TO                       00045.00
 2 SELL-TO-NO                    0100
 2 ALARM-PERCENT                 016
****  End of record  ****
                                                                    08/034
```

Figure 32-23 File Manager: CUSTFILE (Edit 3)

Type the correct value in the field DEC-NO-SHARES and save the record as shown in Figure 32-24.

```
 Process   Options   Help
File Manager Edit CHABERT.TRADER.CUSTFILE
Command ===> _____________________________________________________ Scroll CSR
       Type KSDS                                                Zoom Format SNGL
                                  Top Line is 6    of 16    in Record 1
Field                            Data
  3 COMP-NM                      Casey_Import_Export
 2 DEC-NO-SHARES                     2369
 2 BUY-FROM                      00080.00
 2 BUY-FROM-NO                   0100
 2 BUY-TO                        00050.00
 2 BUY-TO-NO                     0150
 2 SELL-FROM                     00098.00
 2 SELL-FROM-NO                  0200
 2 SELL-TO                       00045.00
 2 SELL-TO-NO                    0100
 2 ALARM-PERCENT                 016
****  End of record  ****
                                                                    10/034
```

Figure 32-24 File Manager: CUSTFILE (Edit 4)

Note: Once any data in the record is changed, all of the fields associated with the record are highlighted.

32.2.4 Running the application after the fix

You have finished correcting the invalid data in the CUSTFILE. Now access the Trader application in CICS, and obtain a real-time quote in IBM for customer TSDEMO.

32.3 Summary of Scenario 1

In this chapter we described the various components that make up the CICS environment in our system and how they are set up.

We reviewed the processing performed by the CICS Trader application.

We detailed a process that:

- Used Debug Tool to follow the execution of the application
- Used Fault Analyzer to identify the cause of an abend in the application
- Described File Manager's capability to identify and correct the data that caused the problem

33

Scenario 2: Using Debug Tool

In this chapter we describe the application components that exist in the batch environment in our system and show how they are set up.

We explain the processing that is performed in the batch Trader application.

The application is forced to produce incorrect output, and then we describe the steps needed to identify the logic error in the application using Debug Tool in batch mode. We also define how to step through the program to isolate and correct the problem using Debug Tool in foreground mode.

33.1 Set up the components

Two types of components need to be established for this scenario:

- Batch components
- Program products
  - Debug Tool

33.1.1 Batch components

Components used by the Trader application are listed in Table 33-1.

Table 33-1 Components of the Trader application for the batch scenario

Component	Details	Remarks
Program	TRADERB	Batch COBOL program
JCL	TRADER	JCL to run the batch application
Files	GRACINE.TRADER.CUSTFILE GRACINE.TRADER.COMPFILE GRACINE.TRADER.TRANFILE	VSAM files and sequential transaction file used by the application
Copybooks	CUSTFILE COMPFILE TRANFILE	File definition for Customer file, company file and transaction file.

33.1.2 Program products

To use the Problem Determination Tools with this scenario, make sure you have the following output or supporting files for the following product:

Debug Tool

You must have a compiler listing or side file for the program TRADERB.

If you are not using the supplied JCL to compile these programs, make sure you specify the following compiler options:

```
LIST, XREF,MAP,RENT,TEST
```

If you prefer to use a side file instead of a compiler listing, include the SEPARATE suboption of the TEST compiler option. Recall that the side file required by Debug Tool is different from the one required by Fault Analyzer. Make sure you run the DEFVSAM1 batch job to load the VSAM files.

33.2 Walkthrough of the batch Trader application

The batch Trader application is used to maintain stock portfolios held by individuals. You execute a batch job that processes a day's worth of trading activity, which:

- Lists portfolios and their value
- Buys shares of a company's stock
- Sells shares of a company's stock

Note: This example was designed to demonstrate the capabilities of the Problem Determination Tools. Therefore, a minimal amount of code was developed. This application does *not* represent real-world securities processing.

33.2.1 The Trader batch job

The JCL to run the batch Trader application is shown in Figure 33-1.

```
  Menu  Utilities  Compilers  Help
 ---------------------------------------------------------------------------
 BROWSE    GRACINE.TRADER.JCL(TRADERB) - 01.15        Line 00000000 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
******************************** Top of Data ********************************
//BATCHTRA JOB ,CLASS=A,NOTIFY=&SYSUID,MSGCLASS=H,MSGLEVEL=(1,1)
//*******************************************************************
//GOTRAD   EXEC PGM=TRADERB
//STEPLIB  DD DISP=SHR,DSN=GRACINE.TRADER.LOAD
//         DD DISP=SHR,DSN=ADTOOLS.DEBUG.TOOL51.SEQAMOD
//SYSPRINT DD SYSOUT=*
//SYSABEND DD SYSOUT=*
//COMPFILE DD DISP=SHR,DSN=GRACINE.TRADER.COMPFILE
//CUSTFILE DD DISP=SHR,DSN=GRACINE.TRADER.CUSTFILE
//TRANSACT DD DISP=SHR,DSN=GRACINE.TRADER.SAMPLES(TRANFILE)
//REPOUT   DD SYSOUT=*
//TRANREP  DD SYSOUT=*
//*
****************************** Bottom of Data ******************************
```

Figure 33-1 JCL to run the batch Trader application

The job invokes program TRADERB, which reads a member of a PDS file with DDname of TRANSACT, to obtain the day's transactions. The program processes each of the records in this file. At the same time, the program reads the company file (COMPFILE) and reads and updates the customer file (CUSTFILE).

After the program processes the input file, it generates two output reports: REPOUT, which contains a list of all customer portfolios and TRANREP, which contains a detailed list of the transaction activity and processing status.

The transaction file

The transaction file is an 80-byte, sequential file that is input to the Trader application a shown in Figure 33-2. It can contain three types of requests:

- List shares
- Buy shares
- Sell shares

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------
EDIT       GRACINE.TRADER.SAMPLES(TRANFILE) - 01.09        Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
****** ***************************** Top of Data ******************************
000001 12345GARY                          .IBM                BUY_SELL
000002 12345GARY                          .IBM                BUY_SELL
000003 12345GARY                          .Casey_Import_Export BUY_SELL
000004 12345GARY                                              SHARE_VALUE
****** **************************** Bottom of Data ****************************
```

Figure 33-2 Batch trader application transaction file (1 of 2)

Figure 33-3 shows the second part of the screen display; you can press the F11 key to move it to the right of the screen.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------
EDIT       GRACINE.TRADER.SAMPLES(TRANFILE) - 01.09        Columns 00009 00080
Command ===> _____________________________________________ Scroll ===> CSR
****** ***************************** Top of Data ******************************
000001 Y                          .IBM                BUY_SELL       00301
000002 Y                          .IBM                BUY_SELL       00162
000003 Y                          .Casey_Import_Export BUY_SELL       00251
000004 Y                                              SHARE_VALUE
****** **************************** Bottom of Data ****************************
```

Figure 33-3 Batch trader application transaction file (2 of 2)

The record layout for the transaction file is as listed in Table 33-2.

Table 33-2 Transaction file record layout

Column	Description	Field Name
1-5	Account Number	TR-ACCOUNT-NUMBER
6-35	Customer Name	TR-CUSTOMER-NAME
36	Dot	FILLER
37-51	Company Name	TR-COMPANY-NAME
52-71	Request Type	TR-REQUEST-TYPE
71-75	Number of shares (buy or sell)	TR-NO-OF-SHARES
76	Transaction Type (buy or sell)	TR-SUBTYPE
77-80	Blank	FILLER

After the program TRADERB reads each record, it examines the TR-REQUEST-TYPE field to determine the type of processing to perform.

33.2.2 Listing shares

If the field TR-REQUEST-TYPE contains the value SHARE_VALUE, the program uses the value in the TR-CUSTOMER-NAME field to print a report that lists the shares held by that customer in each company the customer owns as shown in Figure 33-4.

```
  Display  Filter  View  Print  Options  Help
 ------------------------------------------------------------------------------
 SDSF OUTPUT DISPLAY BATCHTRA JOB03311  DSID   103 LINE 0       COLUMNS 02- 81
 COMMAND INPUT ===> _                                          SCROLL ===> CSR
********************************* TOP OF DATA ***********************************
CUSTOMER : GARY                                                 11/26/2004
-----------------------------------------------------------------
     COMPANY                  SHARES    SHARE            TOTAL
                              HELD      VALUE            COST
-----------------------------------------------------------------
     Casey_Import_Export        79        79.00         6,241.00
     IBM                       524       163.00        85,412.00
******************************** BOTTOM OF DATA *********************************

  F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=IFIND     F6=BOOK
  F7=UP        F8=DOWN      F9=SWAP      F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Figure 33-4 Batch trader application list shares report

33.2.3 Buying and selling shares

If field TR-REQUEST-TYPE contains the value BUY_SELL, and field TR-SUB-TYPE contains a value of 1, the program processes a request to buy the number of shares in TR-NO-OF-SHARES.

If field TR-REQUEST-TYPE contains the value BUY_SELL, and field TR-SUB-TYPE contains a value of 2, the program processes a request to sell the number of shares in TR-NO-OF-SHARES.

After the process completes successfully, the program updates the Customer file, GRACINE.TRADER.CUSTFILE.

The program also produces a transaction report, as shown in Figure 33-5. This report lists the transaction file input request and the status of the processing. The STATUS column in the report lists how the request was processed. If the processing is successful, the message PROCESSED is printed, otherwise the message *ERROR* is printed.

```
  Display  Filter  View  Print  Options  Help
 ------------------------------------------------------------------------------
 SDSF OUTPUT DISPLAY BATCHTRA JOB03311  DSID   104 LINE 0       COLUMNS 02- 81
 COMMAND INPUT ===> _                                          SCROLL ===> CSR
********************************* TOP OF DATA ***********************************
------------------------------------------------------------------------------
CUSTOMER                          COMPANY                 QTY  REQ-TYP  STATUS
------------------------------------------------------------------------------
GARY                              IBM                      30  BUY      PROCESSI
GARY                              IBM                      16  SELL     PROCESSI
GARY                              Casey_Import_Export      25  BUY      PROCESSI
******************************** BOTTOM OF DATA *********************************
```

Figure 33-5 Batch Trader application Transaction report listing BUY / SELL shares (1 of 2)

The second part of the screen display is shown in Figure 33-6; press the F11 key to scroll to the right of the screen.

```
  Display  Filter  View  Print  Options  Help
 ------------------------------------------------------------------------------
 SDSF OUTPUT DISPLAY BATCHTRA JOB03311  DSID   104 LINE 0       COLUMNS 55- 134
 COMMAND INPUT ===> _                                         SCROLL ===> CSR
********************************* TOP OF DATA ********************************
---------------------------
     QTY  REQ-TYP  STATUS
---------------------------
      30  BUY      PROCESSED SUCCESFULLY
      16  SELL     PROCESSED SUCCESFULLY
t     25  BUY      PROCESSED SUCCESFULLY
******************************** BOTTOM OF DATA ******************************
```

Figure 33-6 Batch Trader application transaction report listing BUY / SELL shares (2 of 2)

To demonstrate the capabilities of the Problem Determination Tools, we next describe how to force the application to encounter an error. Consider that your business user, Joe, tells you about a problem with the output contained in one of the reports. You step through the process of fixing it, and use Debug Tool (in batch and foreground mode) to first identify and then to isolate a problem in the application program logic.

In this example, you have a Transaction file that contains the day's trading activity for the customer, RB_DEMO:

- Buy 30 shares of IBM
- Sell 16 shares of IBM
- Buy 25 shares of Casey Import/Export.
- List the shares held by Gary.

This activity is represented by the records shown in Figure 33-7.

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------------
 EDIT       GRACINE.TRADER.SAMPLES(TRANFILE) - 01.09        Columns 00009 00080
 Command ===> _                                              Scroll ===> CSR
 ****** **************************** Top of Data ******************************
 000001 Y                          .IBM                 BUY_SELL       00301
 000002 Y                          .IBM                 BUY_SELL       00162
 000003 Y                          .Casey_Import_Export BUY_SELL       00251
 000004 Y                                               SHARE_VALUE
 ****** **************************** Bottom of Data ***************************
```

Figure 33-7 Transaction record for batch scenario

You submit the batch job, TRADER. The TRADERB application program reads the input from the Transaction file and processes the requests. The results of the transaction processing is printed as a report, as shown in Figure 33-8.

```
  Display  Filter  View  Print  Options  Help
 ------------------------------------------------------------------------------
 SDSF OUTPUT DISPLAY BATCHTRA JOB03311  DSID   104 LINE 0       COLUMNS 02- 81
 COMMAND INPUT ===> _                                         SCROLL ===> CSR
********************************* TOP OF DATA ***********************************
------------------------------------------------------------------------------
CUSTOMER                            COMPANY                 QTY  REQ-TYP  STATUS
------------------------------------------------------------------------------
GARY                                IBM                      30  BUY      PROCESSI
GARY                                IBM                      16  SELL     PROCESSI
GARY                                Casey_Import_Export      25  BUY      PROCESSI
******************************** BOTTOM OF DATA *********************************
```

Figure 33-8 TRANREP report from batch job TRADERB (1 of 2)

Use the PF11 key to scroll to the right of the screen for more information as shown in Figure 33-9.

```
  Display  Filter  View  Print  Options  Help
 ------------------------------------------------------------------------------
 SDSF OUTPUT DISPLAY BATCHTRA JOB03311  DSID   104 LINE 0       COLUMNS 55- 134
 COMMAND INPUT ===> _                                         SCROLL ===> CSR
********************************* TOP OF DATA ***********************************
----------------------------
     QTY  REQ-TYP  STATUS
----------------------------
      30  BUY      PROCESSED SUCCESFULLY
      16  SELL     PROCESSED SUCCESFULLY
t     25  BUY      PROCESSED SUCCESFULLY
******************************** BOTTOM OF DATA *********************************
```

Figure 33-9 TRANREP report from batch job TRADERB (2 of 2)

Your business user, Joe, who reviews these reports on a daily basis, tells you there is an error. He shows you the report from September 30th. It only lists the shares held by the customer Gary in the company Casey_Import_Export, which doesn't reconcile with the account.

You check the Transaction Report in Figure 33-8 and sure enough, it shows the buy/sell requests for IBM and Casey_Import_Export were processed successfully. To make sure, you access the CICS Trader application to review Gary's account. The shares for both of these companies are listed.

You can see there is a problem printing all of the shares held by a customer. You know from experience it has something to do with the program logic, because the buy requests have been processed successfully, and two new records have been written to the customer file for Gary.

You decide to investigate further and use Debug Tool.

33.2.4 Using Debug Tool in batch mode to try to find the error

You are going to use Debug Tool to show you the flow of the program so that you can find out where the program is experiencing the problem. Do this by listing the paragraphs that are performed when the job executes.

To do this, create a commands file for Debug Tool commands, and instruct Debug Tool to use this file at the start of the debug session.

Setting up the commands file

For this example, create a commands file. This can be any fixed-block, 80-byte sequential file, or a member of a partitioned data set (PDS). Figure 33-10 contains the commands to list the paragraphs that are performed when the program executes.

```
  Menu  Utilities  Compilers  Help
 ------------------------------------------------------------------------
 BROWSE    GRACINE.TRADER.CMDS(CMD01) - 01.00        Line 00000000 Col 001 080
 Command ===> _____________________________________________ Scroll ===> CSR
******************************** Top of Data ********************************
AT GLOBAL LABEL PERFORM;
    LIST LINES %LINE;
    GO;
   END-PERFORM;
   GO;
   QUIT;
****************************** Bottom of Data ******************************
```

Figure 33-10 Debug Tool commands to list paragraph names

This routine requests a listing of the line number and name of each paragraph (label) in the program.

Running Debug Tool in batch mode

You can also create a batch job to invoke Debug Tool to debug your program. The fastest way to do this is to modify the Trader batch job. Then use the JCL shown in Figure 33-11.

Include the TEST runtime option and point to your Commands file. The output from the Commands file will be directed to the JES spool (although it could also go to a sequential file).

```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ------------------------------------------------------------------------
 EDIT       GRACINE.TRADER.JCL(TRADERD1) - 01.02            Columns 00001 00072
 Command ===> _____________________________________________ Scroll ===> CSR
 ****** ***************************** Top of Data ******************************
 000001 //TRADERD1 JOB ,CLASS=A,NOTIFY=&SYSUID,MSGCLASS=H,MSGLEVEL=(1,1),
 000002 //         REGION=0M
 000003 //GO       EXEC PGM=TRADERB,
 000004 //         PARM='/TEST(,INSPIN,,)'
 000005 //STEPLIB  DD DISP=SHR,DSN=GRACINE.TRADER.LOAD
 000006 //         DD DISP=SHR,DSN=ADTOOLS.DEBUG.TOOL51.SEQAMOD
 000007 //SYSPRINT DD SYSOUT=*
 000008 //SYSABEND DD SYSOUT=*
 000009 //COMPFILE DD DISP=SHR,DSN=GRACINE.TRADER.COMPFILE
 000010 //CUSTFILE DD DISP=SHR,DSN=GRACINE.TRADER.CUSTFILE
 000011 //TRANSACT DD DISP=SHR,DSN=GRACINE.TRADER.SAMPLES(TRANFILE)
 000012 //REPOUT DD SYSOUT=*
 000013 //TRANREP DD SYSOUT=*
 000014 //INSPIN DD DISP=SHR,DSN=GRACINE.TRADER.CMDS(CMD01)
 000015 //INSPLOG DD SYSOUT=*,DCB=(LRECL=72,RECFM=FB)
 000016 //*
 ****** ***************************** Bottom of Data ***************************
  F1=Help      F2=Split     F3=Exit      F5=Rfind     F6=Rchange   F7=Up
  F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 33-11 Batch job to run Debug Tool for quick problem identification

Make the following changes to the JCL:

- Add a comma after the program name.

- Include a parameter that specifies overrides to runtime options, and include TEST and your commands file.
- Add the load library for Debug Tool to the STEPLIB concatenation (if it is in LINKLIST).
- Add the DD statement INSPIN and use your commands file.
- Add the DD statement INSPLOG and use the JES spool for the log file.

Submit this job. After the batch job completes, review the output of the log file.

Contents of the log file

Debug Tool lists each of the line numbers and paragraph names in the log file, as shown in Figure 33-12 and Figure 33-13.

```
  Display  Filter  View  Print  Options  Help
 ------------------------------------------------------------------------------
 SDSF OUTPUT DISPLAY TRADERD1 JOB03323  DSID   105 LINE 0       COLUMNS 02- 81
 COMMAND INPUT ===> _                                         SCROLL ===> CSR
********************************* TOP OF DATA ***********************************
      * IBM Debug Tool Version 5 Release 1 Mod 0
      * 11/26/2004  2:14:22 PM
      * 5655-M18 and 5655-M19: (C) Copyright IBM Corp. 1992, 2004
      * EQA1872E An error occurred while opening file: INSPPREF. The
      *          file may not exist, or is not accessible.
      * An error was found at line 1 in the current input file.
      * The partially parsed command is:
      *  TRIGGER GLOBAL
      * The command element LABEL is invalid.
        LIST LINES %LINE ;
      * Statement   is not valid.
        GO ;
      * You were prompted because the CEE067  condition was raised in
      * your program.
      * CEE067 is a severity or class 1 condition.
      * The operating system has generated the following message:
      *    CEE0199W The termination of a thread was signaled due to a
  F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=IFIND     F6=BOOK
  F7=UP        F8=DOWN      F9=SWAP      F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Figure 33-12 Log file listing paragraphs performed during program execution (Part 1 of 2)

Scroll down using the PF8 key to display the rest of the log file as shown in Figure 33-13.

```
  Display  Filter  View  Print  Options  Help
--------------------------------------------------------------------------------
SDSF OUTPUT DISPLAY TRADERD1 JOB03323  DSID   105 LINE 18      COLUMNS 02- 81
COMMAND INPUT ===> _                                          SCROLL ===> CSR
     * STOP statement.
     * The current location is TRADERB ::> TRADERB :> 346.1.
     * An error was found at line 4 in the current input file.
     * The command element END-PERFORM is invalid.
       GO ;
******************************** BOTTOM OF DATA ********************************
 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=IFIND     F6=BOOK
 F7=UP        F8=DOWN      F9=SWAP      F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Figure 33-13 Log file listing paragraphs performed during program execution (Part 2 of 2)

Review the program's processing along with the log file

Review what TRADERB is designed to do to try and isolate the problem.

Recall that the customer file has one record for every company in which the customer holds shares.

When a transaction to list shares is processed, the program starts to read the customer file. It reads the records one at a time and prints the details, until the record of a different customer is read.

Review the Transaction file and see the two transactions. You realize that it does not matter if Gary had no shares in IBM and Casey_Import_Export before the Trader batch job executed, because two records were written to the customer file when the program processed these records. One was for IBM and another was for Casey_Import_Export.

Recognize that when TRADERB processes the record in the transaction file to list the shares held by Gary, the paragraph READ-CUSTFILE-NEXT should be executed at least four times (one read past the current Customer record).

Look carefully at the Log file, which shows that READ-CUSTFILE-NEXT is only executed twice. This shows you there is a problem with the logic in the section of the program that reads the Customer file.

33.2.5 Using Debug Tool in DTSU to pinpoint the solution

Here we discuss a solution using Debug Tool.

Setting up a DTSU session

For this example, run the TRADERB application from DTSU.

Start your DTSU session using the following steps:

1. Select **DTU** from ISPF option (Debug Tool Utilities) or enter the command **`exec 'your_hlq.seqaexec(eqastart)'`** from ISPF option 6.
2. Select option **2** (Manage and Use Debug Tool Setup Files).
3. In Figure 33-14, enter a member name, like TRADERB. (This implies that you have used DTSU before, and your control file was allocated before). Do not insert a slash just before initialize New setup file for DB2 (/), since this is not a DB2 application.

```
------------------- Debug Tool Foreground - Edit Setup File ------------------
Command ===> ____________________________________________________________________


Setup File Library:
   Project . . . gracine
   Group . . . . trader  . . . ________ . . . ________ . . . ________
   Type  . . . . DTSF
   Member  . . . traderb        (Blank or pattern for member selection list)
                                (or existing or new member name)

Other Data Set Name:
   Data Set Name . . . ____________________________________________________
   Volume Serial . . . ______    (If not cataloged)


_ Initialize New setup file for DB2 (/)




 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 33-14 DTSU edit setup, file 1

4. On the panel shown in Figure 33-15:
   – Enter the name of the load module to be tested (TRADERB).
   – 1 for COBOL, to define the parameter format.
   – / to modify parameters.

   Press Enter.

```
EDIT - Edit Setup File 'GRACINE.TRADER.DTSF(TRADERB)'          Row 1 to 1 of 1
Command ===> ___________________________________________ Scroll ===> crs

Modify information and use the  Run command to execute,
        or the  Submit command to Batch.
     Press HELP for a list of all available commands.

Load Module Name traderb__
Choose the format of your parameter string:
1 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs    - Run-time Options  / Program Arguments

/ Enter / to modify parameters ____________________________________________

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)           DISP
___ ________ ___ _ ***************** Top of Data ********************* _____
******************************** Bottom of data ********************************



 F1=Help      F3=Exit      F4=Run       F7=Backward  F8=Forward  F10=Submit
F12=Cancel
```

Figure 33-15 DTSU edit setup, File 2

5. The next panel displays the default runtime options to start the Debug Tool in a TSO environment. Since you will use a commands and preferences file, you will also define an INSPCMD DD name (INSPREF is the default DD name for the preferences file). Note that the session type has the choice of MFI or Remote Debugger, and the VTAM LU name is blank to use the TSO session. In this sampling, the remote debugger can be used but MFI will be preselected. The following figures show two different ways of setting up the parameters:
   - Figure 33-16 and Figure 33-17 show the parm setting for the Remote Debugger.
   - Figure 33-18 and Figure 33-19 show the parm setting for the MFI.

```
--------------- Debug Tool Foreground - Modify Parameter String --------------
Command ===> ____________________________________________________________________
                                                                   More:     +
Modify Test options, other run-time options, and Program arguments

Select Test Options:

Test Option      ==> TEST___________  Test/Notest
Test Level       ==> ALL____________  All/Error/None
Commands File                         *, DDname, or Data Set Name
                 ==> *________________________________________________________
Prompt Level     ==> PROMPT_________  Prompt/NoPrompt
Preference File                       *, DDname, or Data Set Name
                 ==> *________________________________________________________

Select (/) a session type and provide parameters:

 _ Full-screen mode
   Terminal LU     ==> ____________   blank or MFI VTAM Terminal LU

 / Remote debug mode
   Connection type ==> SINGLE___      SINGLE/MULTIPLE socket
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 33-16 Remote Debugger Parm String (Part 1 of 2)

```
--------------- Debug Tool Foreground - Modify Parameter String --------------
Command ===> ____________________________________________________________________
                                                                   More:   -

 _ Full-screen mode
   Terminal LU     ==> ____________   blank or MFI VTAM Terminal LU

 / Remote debug mode
   Connection type ==> SINGLE___      SINGLE/MULTIPLE socket
   Address         ==> 9.28.205.51____________________________________________
   Port            ==> 8001_______

Other run-time options: ____________________________________________________
____________________________________________________________________________
____________________________________________________________________________

Program arguments: _________________________________________________________
____________________________________________________________________________
____________________________________________________________________________



 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 33-17 Remote Debugger Parm String (Part 2 of 2)

```
--------------- Debug Tool Foreground - Modify Parameter String --------------
Command ===> _____________________________________________________________
                                                                   More:     +
Modify Test options, other run-time options, and Program arguments

Select Test Options:

Test Option      ==> TEST            Test/Notest
Test Level       ==> ALL             All/Error/None
Commands File                        *, DDname, or Data Set Name
                 ==> *
Prompt Level     ==> PROMPT          Prompt/NoPrompt
Preference File                      *, DDname, or Data Set Name
                 ==> *

Select (/) a session type and provide parameters:

 / Full-screen mode
   Terminal LU      ==> ____________ blank or MFI VTAM Terminal LU

 _ Remote debug mode
   Connection type ==> SINGLE       SINGLE/MULTIPLE socket
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
```

Figure 33-18 FMI Parm Set-up Selection (Part 1 of 2)

```
--------------- Debug Tool Foreground - Modify Parameter String --------------
Command ===> _____________________________________________________________
                                                                   More:   -

 / Full-screen mode
   Terminal LU      ==> ____________ blank or MFI VTAM Terminal LU

 _ Remote debug mode
   Connection type ==> SINGLE       SINGLE/MULTIPLE socket
   Address         ==> 9.28.205.51
   Port            ==> 8001

Other run-time options: ____________________________________________________
____________________________________________________________________________
____________________________________________________________________________

Program arguments: _________________________________________________________
____________________________________________________________________________
____________________________________________________________________________

 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 33-19 FMI Parm Set-up Selection (Part 2 of 2)

6. If the Remote Debugger is selected, Figure 33-20 shows the launch and main menu.

```
--------------- Debug Tool Foreground - Modify Parameter String --------------
Command ===> ____________________________________________________________________
                                                                   More:     +
Modify Test options, other run-time options, and Program arguments

Select Test Options:

Test Option      ==> TEST___________  Test/Notest
Test Level       ==> ALL____________  All/Error/None
Commands File                         *, DDname, or Data Set Name
                 ==> *________________________________________________________
Prompt Level     ==> PROMPT_________  Prompt/NoPrompt
Preference File                       *, DDname, or Data Set Name
                 ==> INSPPREF_________________________________________________

Select (/) a session type and provide parameters:

 _ Full-screen mode
   Terminal LU      ==> _____________ blank or MFI VTAM Terminal LU

 / Remote debug mode
   Connection type ==> SINGLE__       SINGLE/MULTIPLE socket
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 33-20 DTSU options selection panel (Part 1 of 2)

7. Press PF8 to scroll down to view the additional information as shown in Figure 33-21.

```
--------------- Debug Tool Foreground - Modify Parameter String --------------
Command ===> ____________________________________________________________________
                                                                   More:   -

 _ Full-screen mode
   Terminal LU      ==> _____________ blank or MFI VTAM Terminal LU

 / Remote debug mode
   Connection type ==> SINGLE__       SINGLE/MULTIPLE socket
   Address         ==> 9.30.60.129____________________________________________
   Port            ==> 8001___________

Other run-time options: ____________________________________________________
____________________________________________________________________________
____________________________________________________________________________

Program arguments: _________________________________________________________
____________________________________________________________________________
____________________________________________________________________________

 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 33-21 DTSU options selection panel (Part 2 of 2)

8. Press PF3 to exit. The DTSU Edit Setup File panel has been updated with the Debug Tool options as shown in Figure 33-22.

```
EDIT - Edit Setup File 'GRACINE.TRADER.DTSF(TRADERB)'          Row 1 to 1 of 1
Command ===> _____________________________________________ Scroll ===> CSR

Modify information and use the  Run command to execute,
        or the  Submit command to submit to Batch.
     Press HELP for a list of all available commands.

Load Module Name TRADERB
Choose the format of your parameter string:
1 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs    - Run-time Options  / Program Arguments

_ Enter / to modify parameters '/TEST(ALL,'*',PROMPT,'TCPIP&9.30.60.129%8001:*'
) '
Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)            DISP
___ ________ ___ _ ***************** Top of Data ******************** _____
******************************** Bottom of data ********************************


 F1=Help      F3=Exit      F4=Run       F7=Backward  F8=Forward  F10=Submit
F12=Cancel
```

Figure 33-22 DTSU edit setup: File 3

9. You have not yet defined the resources you need to run this program. Since you already have a JCL to start TRADERB, you need to generate the file allocations. Enter **copy** in the command line. The panel in Figure 33-23 is displayed.

```
EDIT - Edit Setup File 'GRACINE.TRADER.DTSF(TRADERB)'          Row 1 to 1 of 1
Command ===> COPY_________________________________________ Scroll ===> CSR

Modify information and use the  Run command to execute,
        or the  Submit command to submit to Batch.
     Press HELP for a list of all available commands.

Load Module Name TRADERB
Choose the format of your parameter string:
1 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs    - Run-time Options  / Program Arguments

_ Enter / to modify parameters '/TEST(ALL,'*',PROMPT,'TCPIP&9.30.60.129%8001:*
) '
Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)            DISP
___ ________ ___ _ ***************** Top of Data ******************** _____
******************************** Bottom of data ********************************


 F1=Help      F3=Exit      F4=Run       F7=Backward  F8=Forward  F10=Submit
F12=Cancel
```

Figure 33-23 Copy from existing JCL

10. Enter the JCL data set name and press Enter. On the next panel, select the JCL member as shown in Figure 33-24 and Figure 33-25.

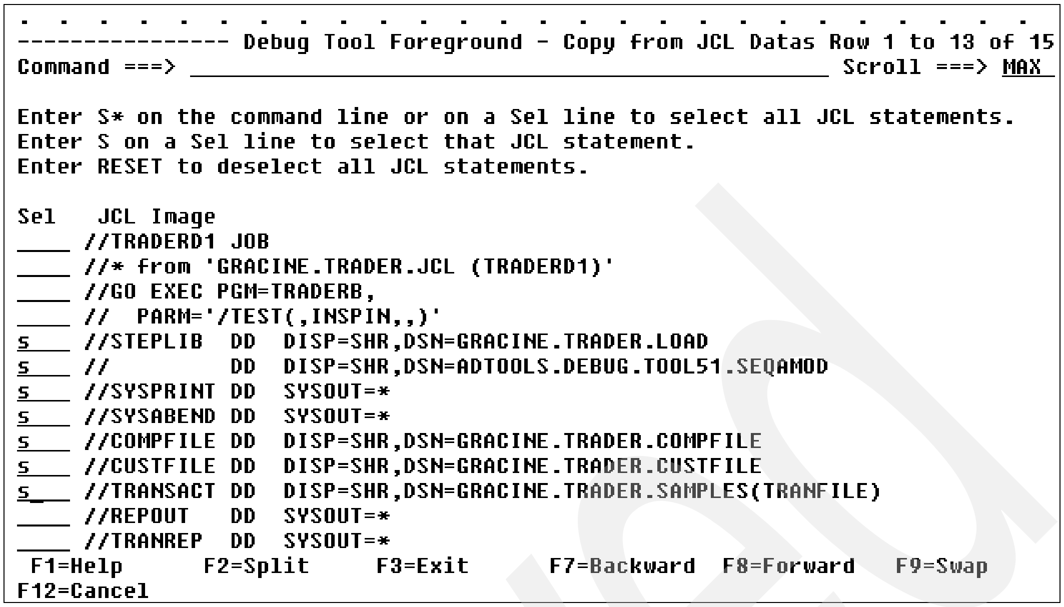
```
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
------------- Debug Tool Foreground - Copy from Setup File or JCL ------------
Command ===> _____________________________________________________________

Select data to copy into 'GRACINE.TRADER.DTSF(TRADERB)'

Setup File or JCL Library:
   Project . . . GRACINE
   Group . . . . TRADER  . . . ________ . . . ________ . . . ________
   Type  . . . . JCL
   Member  . . . ________        (Blank or pattern for member selection list)
                                 (or existing or new member name)

Other Data Set Name:
   Data Set Name . . . ______________________________________________________
   Volume Serial . . . ______    (If not cataloged)


Note: When you copy from another setup file the entire contents are copied.
      When copying from JCL you can select the information you want to copy.


 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 33-24 JCL member selection list (Part 1 of 2)

```
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
  Menu  Functions  Confirm  Utilities  Help
 ____________________________________________________________________________
LIBRARY          GRACINE.TRADER.JCL                     Row 00046 of 00060
Command ===> ____________________________________________ Scroll ===> PAGE
   Name     Prompt          Size    Created          Changed            ID
_ GRANT                      18   1999/08/13   2004/11/04 16:54:42    CHABERT
_ IDISAMP1                   51   2001/06/11   2002/10/17 14:55:40    DAVINR1
_ IDISAMP2                   20   2001/06/11   2002/10/17 14:55:51    DAVINR1
_ IMSCOBBA                   77   2002/10/03   2003/11/19 15:25:00    ERIC
_ IMSCXCOB                   99   2002/10/07   2004/09/01 12:16:06    ERIC
_ IMSGO                      51   2002/10/03   2002/10/17 10:11:51    DAVINR1
_ PDTCOPY                    27   2001/07/18   2001/07/31 21:58:25    DAVIN7
_ RUNSTAT                    15   2004/10/20   2004/10/20 15:39:41    CHABERT
_ TABLES                     48   1999/08/09   2004/11/04 16:32:35    CHABERT
_ TRADERB                    13   2001/06/21   2004/11/26 09:33:23    GRACINE
_ TRADERBS                   20   2001/06/21   2004/11/22 19:02:00    CHABER2
_ TRADERD                    23   2002/09/25   2004/11/22 19:02:05    CHABER2
s TRADERD1 _                 16   2004/11/26   2004/11/26 16:55:38    GRACINE
_ TRADERI                    17   2002/10/07   2004/11/22 19:02:09    CHABER2
_ TRANFILE                    3   2004/11/21   2004/11/26 10:04:33    GRACINE
  **End**
 F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
F10=Left   F11=Right  F12=Cancel
```

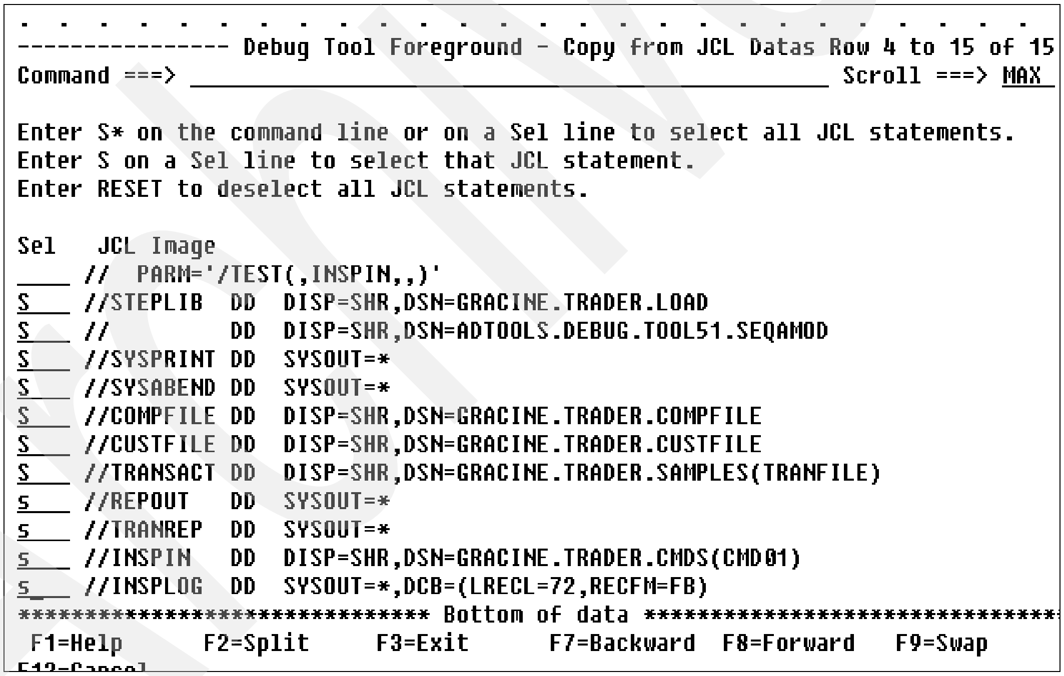
Figure 33-25 JCL member selection list (Part 2 of 2)

11. Figure 33-26 and Figure 33-27 display the selected JCL. You can either select all the DD statements needed, one by one, or you can enter `S*` in the command line and then deselect all DD statements you do not need.



```
 .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
---------------- Debug Tool Foreground - Copy from JCL Datas Row 1 to 13 of 15
Command ===> ________________________________________________ Scroll ===> MAX

Enter S* on the command line or on a Sel line to select all JCL statements.
Enter S on a Sel line to select that JCL statement.
Enter RESET to deselect all JCL statements.

Sel   JCL Image
____ //TRADERD1 JOB
____ //* from 'GRACINE.TRADER.JCL (TRADERD1)'
____ //GO EXEC PGM=TRADERB,
____ //  PARM='/TEST(,INSPIN,,)'
s___ //STEPLIB  DD  DISP=SHR,DSN=GRACINE.TRADER.LOAD
s___ //         DD  DISP=SHR,DSN=ADTOOLS.DEBUG.TOOL51.SEQAMOD
s___ //SYSPRINT DD  SYSOUT=*
s___ //SYSABEND DD  SYSOUT=*
s___ //COMPFILE DD  DISP=SHR,DSN=GRACINE.TRADER.COMPFILE
s___ //CUSTFILE DD  DISP=SHR,DSN=GRACINE.TRADER.CUSTFILE
s___ //TRANSACT DD  DISP=SHR,DSN=GRACINE.TRADER.SAMPLES(TRANFILE)
____ //REPOUT   DD  SYSOUT=*
____ //TRANREP  DD  SYSOUT=*
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 33-26   JCL selection panel (Part 1 of 2)



```
 .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
---------------- Debug Tool Foreground - Copy from JCL Datas Row 4 to 15 of 15
Command ===> ________________________________________________ Scroll ===> MAX

Enter S* on the command line or on a Sel line to select all JCL statements.
Enter S on a Sel line to select that JCL statement.
Enter RESET to deselect all JCL statements.

Sel   JCL Image
____ //  PARM='/TEST(,INSPIN,,)'
S___ //STEPLIB  DD  DISP=SHR,DSN=GRACINE.TRADER.LOAD
S___ //         DD  DISP=SHR,DSN=ADTOOLS.DEBUG.TOOL51.SEQAMOD
S___ //SYSPRINT DD  SYSOUT=*
S___ //SYSABEND DD  SYSOUT=*
S___ //COMPFILE DD  DISP=SHR,DSN=GRACINE.TRADER.COMPFILE
S___ //CUSTFILE DD  DISP=SHR,DSN=GRACINE.TRADER.CUSTFILE
S___ //TRANSACT DD  DISP=SHR,DSN=GRACINE.TRADER.SAMPLES(TRANFILE)
s___ //REPOUT   DD  SYSOUT=*
s___ //TRANREP  DD  SYSOUT=*
s___ //INSPIN   DD  DISP=SHR,DSN=GRACINE.TRADER.CMDS(CMD01)
s___ //INSPLOG  DD  SYSOUT=*,DCB=(LRECL=72,RECFM=FB)
******************************* Bottom of data *******************************
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 33-27   JCL selection panel (Part 2 of 2)

Note: Never select a SYSUDUMP, SYSABEND, or SYSMDUMP DD card as ‘SYSOUT=* ’ will route all dump output to your 3270 TSO session. Either do not select at all, or define a data set for it.

12. Enter `Exit` or press PF3. The Edit Setup File panel has now been completed with your selected DD statements as shown in Figure 33-28.

```
EDIT - Edit Setup File 'GRACINE.TRADER.DTSF(TRADERB)'          Row 1 to 4 of 12
Command ===> _____________________________________________ Scroll ===> CSR

Modify information and use the  Run command to execute,
        or the  Submit command to submit to Batch.
     Press HELP for a list of all available commands.

Load Module Name TRADERB
Choose the format of your parameter string:
1 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs    - Run-time Options  / Program Arguments

_ Enter / to modify parameters '/TEST(ALL,'*',PROMPT,'TCPIP&9.30.60.129%8001:*
) '

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)            DISP
___ ________ ___ _ ***************** Top of Data ******************** _____
___ COMPFILE 1__ _ 'GRACINE.TRADER.COMPFILE'                         SHR __
___ CUSTFILE 1__ _ 'GRACINE.TRADER.CUSTFILE'                         SHR __
___ INSPIN   1__ _ 'GRACINE.TRADER.CMDS(CMD01)'                      SHR __
 F1=Help      F3=Exit      F4=Run       F7=Backward  F8=Forward  F10=Submit
F12=Cancel
```

Figure 33-28 Completed edit setup file panel

13. Use the PF8 key to scroll down the panel to find all the chosen and selected DD statements.

14. You can browse (PF8) through the DD information. You can edit (E) any of the files and you can also add some new DD statements (either R for repeat, or I for insert). This is a good time to insert an INSPLOG, INSPREF, and INSPCMD DD statement. We added DD information as shown in the next three figures, Figure 33-29 to Figure 33-31.

```
 File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help

EDIT       GRACINE.TRADER.JCL($$INFO) - 01.00              Columns 00001 00072
Command ===> ___________________________________________ Scroll ===> CSR
****** ***************************** Top of Data ******************************
000001       SEQ  DD  INFORMATION (DSN/SYSIN/SYSOUT/DUMMY)
000002 CMD      DD     DSN  NAME              DISP
000003 INSPCMD  1  'GRACINE.TRADER.INSPCMD'   SHR
000004 INSPLOG  1  'GRACINE.TRADER.INSPLOG'   SHR
000005 INSPREF  1  'GRACINE.TRADER.INSPREF'   SHR
000006 INSPSAFE 1  'GRACINE.TRADER.INSPSAFE'  SHR
****** ***************************** Bottom of Data ***************************
```

Figure 33-29 Revised DD information (Part 1 of 3)

```
 Menu  Utilities  Compilers  Help

BROWSE    GRACINE.TRADER.INSPREF(PREF01) - 01.01     Line 00000000 Col 001 080
Command ===> ___________________________________________ Scroll ===> CSR
********************************* Top of Data **********************************
SET ECHO OFF ;
SET KEYS ON 24 ;
PLAYBACK ENABLE;
SET AUTOMONITOR ON;
******************************** Bottom of Data ********************************
```

Figure 33-30 Revised DD information (Part 2 of 3)

```
File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help

EDIT       GRACINE.TRADER.CMDS(CMD02) - 01.00              Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
****** ***************************** Top of Data ******************************
000001 SET SUFFIX ON;
000002 SET FREQUENCY ON;
000003 AT TERMINATION LIST FREQUENCY * ;
****** **************************** Bottom of Data ****************************
```

Figure 33-31 Revised DD information (Part 3 of 3)

15. Enter RUN or PF4. This will start your program under control of the Debug Tool.

Debug Tool starts, with the first line of the program shown in the Source window, as shown in Figure 33-32 with the Remote Debugger.

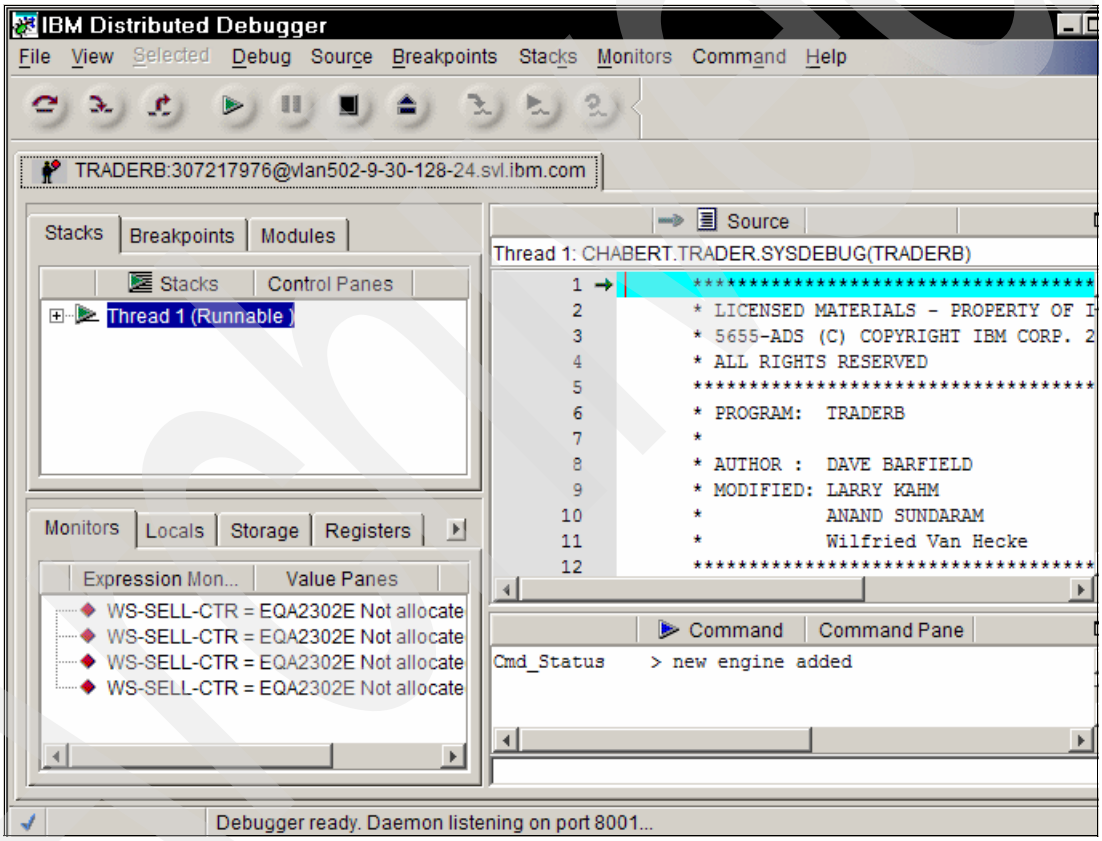


Figure 33-32 Remote Debugger being used after start of program TRADERB

Using MFI for the scenario

Perform the following steps to use MFI for this scenario:

1. Launch MFI with DTU interactively; Figure 33-33 is displayed.

```
EDIT - Edit Setup File 'GRACINE.TRADER.DTSF(TRADERB)'          Execution RC 0
Command ===> _____________________________________________ Scroll ===> CSR

Modify information and use the  Run command to execute,
        or the  Submit command to submit to Batch.
     Press HELP for a list of all available commands.

Load Module Name TRADERB
Choose the format of your parameter string:
1 1 LE COBOL Default   - Program Arguments / Run-time Options
  2 Other LE Languages - Run-time Options  / Program Arguments
  3 Non-LE Programs    - Run-time Options  / Program Arguments

_ Enter / to modify parameters '/TEST(ALL,'*',PROMPT,'MFI:*')  '

Cmd DD Name  Seq C DD Information (DSN/Sysin/Sysout/Dummy)            DISP
___ ________ ___ _ ****************** Top of Data ********************
___ COMPFILE 1__ _ 'GRACINE.TRADER.COMPFILE'                          SHR
___ CUSTFILE 1__ _ 'GRACINE.TRADER.CUSTFILE'                          SHR
___ INSPIN   1__ _ 'GRACINE.TRADER.CMDS(CMD02)'                       SHR
___ INSPLOG  1__ _ *                                                  SHR
 F1=Help      F3=Exit      F4=Run       F7=Backward  F8=Forward  F10=Submit
F12=Cancel
```

Figure 33-33 Press PF4 will launch with MFI

2. Once Debug Tool is started with MFI, Figure 33-34 is displayed.

```
      * IBM Debug Tool Version 5 Release 1 Mod 0
      * 11/28/2004 11:31:31 AM
      * 5655-M18 and 5655-M19: (C) Copyright IBM Corp. 1992, 2004
*** _
```

Figure 33-34 First MFI screen showing LOG entry

3. Press Enter again and Figure 33-35 is displayed.

```
COBOL    LOCATION: TRADERB initialization
Command ===> _                                          Scroll ===> PAGE
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 0 OF
******************************** TOP OF MONITOR ********************************
****************************** BOTTOM OF MONITOR *******************************

SOURCE: TRADERB --1----+----2----+----3----+----4----+----5----+ LINE: 1 OF 81
      1         ************************************************************
      2         * LICENSED MATERIALS - PROPERTY OF IBM
      3         * 5655-ADS (C) COPYRIGHT IBM CORP. 2004
      4         * ALL RIGHTS RESERVED
      5         ************************************************************
      6         * PROGRAM:  TRADERB
LOG 0----+----1----+----2----+----3----+----4----+----5----+----6- LINE: 1 OF
********************************* TOP OF LOG ***********************************
0001 IBM Debug Tool Version 5 Release 1 Mod 0
0002 11/28/2004 11:51:40 AM
0003 5655-M18 and 5655-M19: (C) Copyright IBM Corp. 1992, 2004
PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEAR
PF  7:UP         8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEVE
```

Figure 33-35 Debug Tool with MFI at start of program TRADERB

For now, the MFI will be used to determine that the problem occurs when reading the customer file; a decision is made to set a breakpoint at the START command when issued on the Customer file.

4. Find the first occurrence of the string START-CUSTFILE and set the breakpoint at START-CUSTFILE by typing in quotes "START-CUSTFILE" on the command line as shown in Figure 33-36 and Figure 33-37.

```
COBOL    LOCATION: TRADERB initialization
Command ===> "START-CUSTFILE"                           Scroll ===> PAGE
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 0 OF
******************************** TOP OF MONITOR ********************************
****************************** BOTTOM OF MONITOR *******************************

SOURCE: TRADERB --1----+----2----+----3----+----4----+----5----+ LINE: 1 OF 81
      1         ************************************************************
      2         * LICENSED MATERIALS - PROPERTY OF IBM
      3         * 5655-ADS (C) COPYRIGHT IBM CORP. 2004
      4         * ALL RIGHTS RESERVED
      5         ************************************************************
      6         * PROGRAM:  TRADERB
LOG 0----+----1----+----2----+----3----+----4----+----5----+----6- LINE: 1 OF
********************************* TOP OF LOG ***********************************
0001 IBM Debug Tool Version 5 Release 1 Mod 0
0002 11/28/2004 11:51:40 AM
0003 5655-M18 and 5655-M19: (C) Copyright IBM Corp. 1992, 2004
PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEAR
PF  7:UP         8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEVE
```

Figure 33-36 Press PF5 will set-up the breakpoint

```
COBOL    LOCATION: TRADERB initialization
Command ===>                                                  Scroll ===> PAGE
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 0 OF
******************************** TOP OF MONITOR ********************************
****************************** BOTTOM OF MONITOR *******************************



SOURCE: TRADERB --1----+----2----+----3----+----4----+----5--- LINE: 706 OF 81
     706            PERFORM START-CUSTFILE.
     707            MOVE ZERO TO WS-SHARE-VALUE WS-SHARE-VALUE-GR.
     708            IF RETURN-VALUE = CLEAN-RETURN
     709               PERFORM READ-CUSTFILE-NEXT
     710        *      MOVE KEYREC OF CUSTOMER-IO-BUFFER TO WS-CUST-KEY
     711               MOVE CUST-NM OF CUSTOMER-IO-BUFFER TO WS-CUST-NM
LOG 0----+----1----+----2----+----3----+----4----+----5----+----6- LINE: 1 OF
********************************* TOP OF LOG ***********************************
0001 IBM Debug Tool Version 5 Release 1 Mod 0
0002 11/28/2004 11:51:40 AM
0003 5655-M18 and 5655-M19: (C) Copyright IBM Corp. 1992, 2004
PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEAR
PF  7:UP         8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEVE
```

Figure 33-37 Debug Tool with the breakpoint set at START-CUSTFILE

5. Press F6. This will set the breakpoint on the appropriate line number.

> **Note:** If you already know the line number, you can set the breakpoint by entering the command explicitly on the Command line AT 706

6. Press F9 to issue the GO command.

   The program executes and stops at line 706, PERFORM START-CUSTFILE, before it is executed. Figure 33-38 shows the log of activity.

```
        GO ;
TRADERB STARTED: CURRENT-DATE 20041128
TRAN-FILE-OPEN-> 00
BUY RECORDS        =          2
SELL RECORDS       =          1
UPDATE RECORDS     =          0
      * You were prompted because the CEE067  condition was raised in
      * your program.
      * CEE067 is a severity or class 1 condition.
      * The operating system has generated the following message:
      *    CEE0199W The termination of a thread was signaled due to a
      * STOP statement.
      * The current location is TRADERB ::> TRADERB :> 346.1.
*** _
```

Figure 33-38 Log is displayed upon the activity

7. Press Enter and Figure 33-39 is displayed.

```
COBOL    LOCATION: TRADERB initialization
Command ===>_                                                  Scroll ===> PAGE
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF
******************************** TOP OF MONITOR ********************************
0001  1 KEYREC
0002        The environment is not yet fully initialized.
0003    WS-CUST-KEY
0004        The environment is not yet fully initialized.
****************************** BOTTOM OF MONITOR *******************************

SOURCE: TRADERB --1----+----2----+----3----+----4----+----5--- LINE: 706 OF 81
     706            PERFORM START-CUSTFILE.
     707            MOVE ZERO TO WS-SHARE-VALUE WS-SHARE-VALUE-GR.
     708            IF RETURN-VALUE = CLEAN-RETURN
     709               PERFORM READ-CUSTFILE-NEXT
     710        *      MOVE KEYREC OF CUSTOMER-IO-BUFFER TO WS-CUST-KEY
     711               MOVE CUST-NM OF CUSTOMER-IO-BUFFER TO WS-CUST-NM
LOG 0----+----1----+----2----+----3----+----4----+----5----+----6- LINE: 2 OF
0002 11/28/2004 12:27:09 PM
0003 5655-M18 and 5655-M19: (C) Copyright IBM Corp. 1992, 2004
0004  MONITOR
0005    LIST ( KEYREC, WS-CUST-KEY ) ;
PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEAR
PF  7:UP         8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEVE
```

Figure 33-39 Monitor window displaying values variables

8. Review the code within the current paragraph and note the following:

   a. Line 709 contains the instruction, PERFORM READ-CUSTFILE-NEXT.

   b. There is a MOVE statement that uses two variables, KEYREC and WS-CUST-KEY.

   You want to see what happens to the values in these fields (KEYREC and WS-CUST-KEY) when the customer file is read.

9. Enter the following command on the command line:

   `MONITOR LIST (KEYREC, WS-CUST-KEY);`

   The values of these variables are displayed in the Monitor window as shown in Figure 33-40.

```
COBOL    LOCATION: TRADERB ENTRY
Command ===>_                                                  Scroll ===> PAGE
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF
0001  1 02 TRADERB:>KEYREC
0002       03 TRADERB:>CUST-NM    X'0000000000000000000000000000000000000000
0003       0000000000000000000000000000000000000000000000000000000000000000
0004       0000000000000'
0005       03 TRADERB:>KEYREC-DOT    X'00'
0006       03 TRADERB:>COMP-NM    X'00000000000000000000000000000000000000'
0007    WS-CUST-KEY    X'0000000000000000000000000000000000000000000000000
SOURCE: TRADERB --1----+----2----+----3----+----4----+----5--- LINE: 23 OF 81
      23        ************************************************************
      24         IDENTIFICATION DIVISION.
      25         PROGRAM-ID. TRADERB.
      26         ENVIRONMENT DIVISION.
      27         CONFIGURATION SECTION.
      28         SOURCE-COMPUTER. IBM-370.
LOG 0----+----1----+----2----+----3----+----4----+----5----+----6- LINE: 3 OF
0003 5655-M18 and 5655-M19: (C) Copyright IBM Corp. 1992, 2004
0004  MONITOR
0005    LIST ( KEYREC, WS-CUST-KEY ) ;
0006  STEP ;
PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEA
PF  7:UP         8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEV
```

Figure 33-40 Monitor window displaying values variables

10. Press PF2 to step through the program one statement at a time.

    As shown in Figure 33-40, the program successfully executed the PERFORM START-CUSTFILE statement, and the value in the KEYREC field is Gary.

    Note: The highlighted line in the Source window is the line that will be executed next.

11. Press PF2 until you reach line 718. While you do, pay attention to the value of the variables in the Monitor window.

    As you can see in Figure 33-41, line 718, PERFORM CALCULATE-SHARE-VALUE is performed until the values in the variables KEYREC and WS-CUST-KEY are not equal.

```
COBOL    LOCATION: TRADERB :> 718.1
Command ===> _                                          Scroll ===> PAG
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF
0001  1 02 TRADERB:>KEYREC
0002       03 TRADERB:>CUST-NM   'GARY
0003                            '
0004       03 TRADERB:>KEYREC-DOT   '.'
0005       03 TRADERB:>COMP-NM   'Casey_Import_Export '
0006    WS-CUST-KEY   X'00000000000000000000000000000000000000000000000000
0007    00000000000000000000000000000000000000000000000000000000000000000
SOURCE: TRADERB --1----+----2----+----3----+----4----+----5--- LINE: 716 OF 81
     716                GO TO GENERATE-CUSTOMER-REPORT-EXIT
     717              END-IF.
     718              PERFORM CALCULATE-SHARE-VALUE
     719                UNTIL CUST-NM OF CUSTOMER-IO-BUFFER NOT EQUAL
     720                      WS-CUST-NM.
     721        *       UNTIL KEYREC OF CUSTOMER-IO-BUFFER NOT EQUAL
LOG 0----+----1----+----2----+----3----+----4----+----5----+-- LINE: 255 OF 25
0255  STEP ;
0256  STEP ;
0257  STEP ;
0258  STEP ;
PF  1:?        2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEA
PF  7:UP       8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEV
```

Figure 33-41 Monitor values of variables in Trader application- screen 2

    At this point, you can see the values of both the variables are still equal. The value of the field CUST-NM of KEYREC is Gary and the value of the field, COMP-NM of KEYREC, is Casey_Import_Export. Control is transferred to the CALCULATE-SHARE-VALUE paragraph and the record details are printed.

12. Continue to press PF2 until the next READ statement.

13. Check the values of these variables after the READ statement.

    The values in the variables are different, as shown in Figure 33-42, and the READ process for customer Gary is terminated.

```
COBOL    LOCATION: TRADERB :> 757.1
Command ===> _                                                  Scroll ===> PAG
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF
0001  1 02 TRADERB:>KEYREC
0002       03 TRADERB:>CUST-NM    'GARY                                      
0003                             '
0004       03 TRADERB:>KEYREC-DOT    '.'
0005       03 TRADERB:>COMP-NM    'Casey_Import_Export '
0006    WS-CUST-KEY   'GARY                                                  
0007          .Casey_Import_Export '
SOURCE: TRADERB --1----+----2----+----3----+----4----+----5--- LINE: 753 OF 81
     753                END-EVALUATE.
     754          START-CUSTFILE-EXIT.
     755              EXIT.
     756          READ-CUSTFILE-NEXT.
     757              READ IO-CUSTOMER-FILE NEXT INTO CUSTOMER-IO-BUFFER.
     758                EVALUATE WS-CUST-FILE-STATUS
LOG 0----+----1----+----2----+----3----+----4----+----5----+-- LINE: 271 OF 27
0271  STEP ;
0272  STEP ;
0273  STEP ;
0274  STEP ;
PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEA
PF  7:UP         8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEV
```

Figure 33-42 Variable values after READ statement

You can see that the record is for GARY because the field CUST-NM of KEYREC has that value.

But the key value, KEYREC, is different from WS-CUST-KEY because the field COMP-NM of KEYREC has a new value, IBM, and the variable WS-CUST-KEY still has the old value.

Because these values are different, control is not transferred to the CALCULATE-SHARE-VALUE paragraph, as shown in Figure 33-43 and the READ process for this customer is terminated.

```
COBOL    LOCATION: TRADERB :> 725.1
Command ===> _                                                  Scroll ===> PAGE
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF
0001  1 02 TRADERB:>KEYREC
0002       03 TRADERB:>CUST-NM    'TSDEMO                                    
0003                             '
0004       03 TRADERB:>KEYREC-DOT    '.'
0005       03 TRADERB:>COMP-NM    'Casey_Import_Export '
0006    WS-CUST-KEY   'GARY                                                  
0007          .IBM                '
SOURCE: TRADERB --1----+----2----+----3----+----4----+----5--- LINE: 723 OF 81
     723          *    PERFORM WRITE-TOTALS.
     724          GENERATE-CUSTOMER-REPORT-EXIT.
     725              EXIT.
     726          CALCULATE-SHARE-VALUE.
     727          *    IF KEYREC OF CUSTOMER-IO-BUFFER = WS-CUST-KEY AND
     728               IF CUST-NM OF CUSTOMER-IO-BUFFER = WS-CUST-NM AND
LOG 0----+----1----+----2----+----3----+----4----+----5----+-- LINE: 310 OF 31
0310  STEP ;
0311  STEP ;
0312  STEP ;
0313  STEP ;
PF  1:?          2:STEP       3:QUIT       4:LIST       5:FIND       6:AT/CLEA
PF  7:UP         8:DOWN       9:GO        10:ZOOM      11:ZOOM LOG  12:RETRIEV
```

Figure 33-43 Control not transferred

14. Enter the `QUIT ABEND` command on the command line to end the session.

You found that saving the value of the previously read key value of the Customer record and checking it with the key value immediately after the next read is causing the problem.

Because the customer has one record for every company in which he holds shares, the program logic must be changed to check only the CUST-NM of KEYREC. Saving the CUST-NM field of KEYREC and checking it just after a READ NEXT should solve the problem.

The changes you make to the program are (the corrected “after lines” were commented out):

Before (line 710)

```
MOVE KEYREC OF CUSTOMER-IO-BUFFER TO WS-CUST-KEY
```

After (line 711)

```
MOVE CUST-NM OF CUSTOMER-IO-BUFFER TO WS-CUST-NM
```

Before (line 721, 722)

```
PERFORM CALCULATE-SHARE-VALUE
   UNTIL KEYREC OF CUSTOMER-IO-BUFFER NOT EQUAL
         WS-CUST-KEY.
```

After (line 719, 720)

```
PERFORM CALCULATE-SHARE-VALUE
   UNTIL CUST-NM OF CUSTOMER-IO-BUFFER NOT EQUAL
         WS-CUST-NM.
```

Before (line 736)

```
MOVE CUST-NM OF CUSTOMER-IO-BUFFER TO WS-CUST-NM
```

After (line 735)

```
MOVE KEYREC OF CUSTOMER-IO-BUFFER TO WS-CUST-KEY
```

33.2.6 Executing the batch application after the fix

Recompile the program after you finish correcting the program logic in TRADERB.

Submit the batch job, TRADERB.

33.2.7 Summary of scenario 2

In this chapter we described the various components that make up the batch environment in our system and how they are set up.

We reviewed these topics:

1. The processing performed by the batch Trader application
2. A process that used Debug Tool in batch mode to identify a possible problem in the application
3. Debug Tool’s capability in foreground mode to pinpoint an error to allow it to be corrected

34

Is the error in DB2 data or program logic?

In this chapter we describe the application components that exist in the CICS and DB2 environments on our system and explain how they are set up.

The application is forced to encounter an error. We describe the steps needed to identify the cause of the problem in the application using Debug Tool, and show how to manipulate the data to correct the problem using File Manager DB2.

34.1 Set up the components

Two types of components need to be established for this scenario:

- CICS and DB2 components
- Program products:
  - Debug Tool
  - File Manager DB2

34.1.1 CICS and DB2 components

The components used by the Trader application are listed in Table 34-1.

Table 34-1 CICS components of the Trader application for the DB2 scenario

Component	Details	Remark
Programs	MYTRADMD MYTRADD	CICS DB2 COBOL programs
Tran ID	TDB2	CICS transaction associated with the program
Mapset	NEWTRAD	BMS mapset containing all the maps used by the application.
Tables	CUSTOMER_DETAILS COMPANHY_DETAILS	DB2 tables used by the application

34.1.2 Program products

To use the Problem Determination Tools with this scenario, make sure you have the following output or supporting files for each product.

Debug Tool

You must have a compiler listing or side file for the programs MYTRADMD and MYTRADD.

If you are not using the supplied JCL to compile these programs, make sure you specify the following compiler options:

```
LIST,XREF,RENT,TEST
```

If you prefer to use a side file instead of a compiler listing, include the SEPARATE sub-option of the TEST compiler option. Recall the side file required by Debug Tool is the same as the one required by Fault Analyzer.

File Manager

You will use the templates for the DB2 tables CUSTOMER_DETAILS and COMPANY_DETAILS dynamically created at first edit of these tables.

Make sure you run the TABLES batch job to create the DB2 tables, and then run the DATA batch job to load the DB2 tables.

34.2 Tracking a problem in the application

To demonstrate the capabilities of the Problem Determination Tools, we show you how to force the application to encounter an error and step you through the process of fixing it. Debug Tool is used to follow the execution of the application, and identify and isolate the problem. File Manager is used to correct an error in a DB2 database.

34.2.1 Recreating the error

In this example, you invoke the Trader application and perform the following actions for the customer, RB_DEMO, using IBM as the company in which shares are traded:

1. Launch the Trader application (TDB2 transaction).
2. Select IBM as the company you want to trade.
3. Select option **1** New Real-Time Quote on the Options panel to display the number of shares available. This amount may differ from one run of the transaction to another depending on the operations done on the data. The result may be something like that shown in Figure 34-1.

```
                         Share Trading Demonstration                TRADER.T004

                    Share Trading Manager: Real-Time Quote

      User Name:     RB_DEMO

      Company Name:  IBM

      Share Values:                     Commission Cost:
         NOW:            109.00            for Selling:         010
         1 week ago:     111.00            for Buying:          005
         6 days ago:     112.00
         5 days ago:     111.00
         4 days ago:     109.00
         3 days ago:     107.00
         2 days ago:     107.00         Number of Shares Held: 0075
         1 day ago:      106.00         Value of Shares Held:        8175.0



Request Completed OK
-------------------------------------------------------------------------------
PF3=Return                                                          PF12=Exit
                                                                       01/001
```

Figure 34-1 Number of shares available

4. Press PF3 Return and select option **3** Sell Shares on the Options panel.
5. On the Shares - Sell panel, shown in Figure 34-2, sell more shares (100) than the number held (75).

```
                    Share Trading Demonstration                    TRADER.T005

                 Share Trading Manager: Shares - Sell


            User Name:     RB_DEMO

            Company Name:  IBM


            Number of Shares to Sell: 100_




--------------------------------------------------------------------------------
PF3=Return                                                            PF12=Exit
                                                                        13/047
```

Figure 34-2 Recreating a problem in the Trader application

The Options panel is re-displayed with the message:

```
Request Completed OK
```

6. Select option **1** New Real-Time Quote on the Options panel to obtain real-time quotes and a listing of the shares held as shown Figure 34-3.

```
                    Share Trading Demonstration                    TRADER.T004

                 Share Trading Manager: Real-Time Quote

      User Name:     RB_DEMO

      Company Name:  IBM

      Share Values:                    Commission Cost:
         NOW:          109.00            for Selling:        010
         1 week ago:   111.00            for Buying:         005
         6 days ago:   112.00
         5 days ago:   111.00
         4 days ago:   109.00
         3 days ago:   107.00
         2 days ago:   107.00         Number of Shares Held: 0025
         1 day ago:    106.00         Value of Shares Held:        2725.0



Request Completed OK
--------------------------------------------------------------------------------
PF3=Return                                                            PF12=Exit
                                                                        01/001
```

Figure 34-3 Recreating a problem in the Trader application

Note: The share details would be listed. The Number of Shares Held field would have a value of **25** and it should be **-25** (75 - 100 = **-25**)

7. Press PF3 to return to the Options panel.
8. Select option **2** Buy Shares and buy 25 shares.

The Options panel is re-displayed with the message:

```
Request Completed OK.
```

9. Once again, select option **1** New Real-Time Quote and review the number of shares held. Figure 34-4 shows this value as 0 (-25 + 25 = 0).

```
                          Share Trading Demonstration                TRADER.T004

                     Share Trading Manager: Real-Time Quote

       User Name:     RB_DEMO

       Company Name:  IBM

       Share Values:                     Commission Cost:
          NOW:          109.00             for Selling:        010
          1 week ago:   111.00             for Buying:         005
          6 days ago:   112.00
          5 days ago:   111.00
          4 days ago:   109.00
          3 days ago:   107.00
          2 days ago:   107.00          Number of Shares Held: 0000
          1 day ago:    106.00          Value of Shares Held:  000000000000



 Information: You hold no shares in this company !
 ------------------------------------------------------------------------------
 PF3=Return                                                          PF12=Exit
                                                                        01/001
```

Figure 34-4 Recreating a problem in the Trader application (part 3)

Clearly, there is a serious problem with this series of transactions.

We believe the problem is with the data in the table, CUSTOMER_DETAILS, or in the program that reads the table.

We decide to look first at the specific customer record in the database to see if that will help you understand more about the problem.

Tip: Sell 5 more shares before going to the next step in the analysis process.

34.2.2 Viewing the data in File Manager/DB2

The following steps allow you to view data:

1. To determine which DB2 was used, from CICS we issue the command:

   ```
   CEMT I DB2CONNECTION
   ```

   The panel shown in Figure 34-5 is returned.

```
 I DB2CONN _
 STATUS:  RESULTS - OVERTYPE TO MODIFY
   Accountrec( Uow )              Planexitname( DSNCUEXT )
   Authid(          )             Priority( High )
   Authtype( Userid )             Purgecyclem( 00 )
   Comauthid(         )           Purgecycles( 30 )
   Comauthtype( Cuserid )         Resyncmember( Resync )
   Comthreadlim( 0001 )           Signid( CHABERT  )
   Comthreads(0000)               Security(              )
   Connecterror( Sqlcode )        Standbymode( Reconnect )
   Connectst( Connected )         Statsqueue( CDB2 )
   Db2groupid( D80G )             Tcblimit( 0012 )
   Db2id( D82G )                  Tcbs(0000)
   Db2release(0810)               Threaderror(N906d)
   Drollback(Rollback)            Threadlimit( 0003 )
   Msgqueue1( CDB2 )              Threads(0000)
   Msgqueue2(      )              Threadwait( Twait )
   Msgqueue3(      )
   Nontermrel( Release )
   Plan(          )

                                                  SYSID=C22F APPLID=CICSC22F
  RESPONSE: NORMAL                          TIME:  14.51.18  DATE: 11.26.04
PF 1 HELP       3 END       5 VAR        7 SBH 8 SFH 9 MSG 10 SB 11 SF
                                                                    01/013
```

Figure 34-5 CICS DB2Connection

2. Access File Manager/DB2 in your ISPF session.

   If your system contains only one active DB2 subsystem, File Manager/DB2 automatically connects to that subsystem.

   However, since we are working in an environment that contains more than one active DB2 subsystem, we must select a DB2 subsystem before File Manager/DB2 can connect to it.

   Overtype the ID of the DB2 subsystem currently shown in the DB2 SSID field with the ID of the active DB2 subsystem you want, and press Enter, or delete the contents of this field and select from the list displayed when you press Enter. The File Manager/DB2; DB2 Subsystem Selection menu is displayed, as shown in Figure 34-6. Because our CICS DB2Connection uses the DB2 data sharing group, we select the DB2 data sharing group as shown in the figure.

```
 Process   Help
 ____________________________________________________________________________
 FM/DB2 (D80G)            DB2 Subsystem Selection                 Row 1 of 6
 Command ===> ____________________________________________________ Scroll PAGE

 Sel SSID Status    Description                                   Prefix
 ----------------------------------------------------------------------------
  _  DB1E ACTIVE    DB2 VERSION 7 LPAR F6                         -DB1E
  _  D82G ACTIVE    DB2 VERSION 8 LPAR F6 - GROUP D80G            -D82G
  S _D80G GROUP     DB2 DATA SHARING GROUP
     D61F INACTIVE                                                -D61F
     D71F INACTIVE                                                -D71F
     D81G INACTIVE  DB2 VERSION 8 LPAR F7 - GROUP D80G            -D81G
     ****  End of data  ****

                                                                    10/005
```

Figure 34-6 File Manager/DB2 - Data Sharing Group

3. Press Enter, select option **1** Browse, and press Enter. The DB2 Browse panel shown in Figure 34-7 is displayed.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------------
FM/DB2 (D80G)                      DB2 Browse
Command ===> _____________________________________________________________________

Specify the DB2 Object:
   Location  . . . . . _________________        Database . . ________ (optional)
   Owner . . . . . . . CHABERT          +       Table space  ________ (optional)
   Name  . . . . . . . CUSTOMER DETAILS                                        +

   Row count . . . . . 100        Number of rows to browse

Template:
   Data set name . . . ______________________________________________
   Member  . . . . . . ________

Processing Options:
   Template usage                        Enter "/", "A"lways to select option
   3  1. Above                           _  Edit options
      2. Previous                       _  Edit template
      3. Generate from table            _  Re-edit template
      4. Generate/Replace

                                                                        04/015
```

Figure 34-7 File Manager/DB2 - Browse panel

4. Specify the following information:
   a. The table Owner: user ID used at installation time
   b. The table name
   c. Select **3**, Generate from table in the Processing Options field.

The Table Browse panel is displayed in Figure 34-8.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------------
FM/DB2 (D80G)                      Table Browse                    9 rows fetched
Command ===> ____________________________________________________ Scroll PAGE
TABLE CHABERT.CUSTOMER_DETAILS                                    Format TABL
CUSTOMER                  COMPANY                 NO_SHARES
#1                        #2                            #3
CHARACTER(25)             CHARACTER(20)            INTEGER
PU--+----1----+----2----> PU--+----1----+----> <---+----1>
****  Top of data  ****
ANAND                     Headworth_Electrical          25
ANAND                     IBM                            5
RB_DEMO                   Glass_and_Luget_plc           45
RB_DEMO                   Headworth_Electrical          55
RB_DEMO                   IBM                           -5
LARRY                     Casey_Import_Export          100
LARRY                     ShareSelect                   65
E                         Casey_Import_Export            4
E                         IBM                            3
****  End of data  ****

                                                                        04/015
```

Figure 34-8 File Manager-DB2 - Table Browse Panel

5. The record containing the string RB_DEMO is displayed as the first (and only) page in the panel. Locate the record that has a value of IBM in the COMPANY column.

You can see in Figure 34-9 (using the SiNGLe display format) that the value in the NO_SHARES column is -5. This is incorrect data in the application. No negative value is allowed.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------------
FM/DB2 (D80G)                      Table Browse
Command ===> _____________________________________________________ Scroll PAGE
TABLE CHABERT.CUSTOMER_DETAILS                                     Format SNGL
                                        Top Line is 1    of 3     in Row 5
Key     Column      Data
PU      CUSTOMER    RB_DEMO
PU      COMPANY     IBM
        NO_SHARES            -5
****  End of record  ****
                                                                     04/015
```

Figure 34-9 File Manager-DB2 - Table Browse Record

At this point, we believe the problem is due to faulty logic in the program that updates the CUSTOMER_DETAILS table.

We can review the compiler listing to get an overview of the program, and to see where the table is processed.

We decide to debug the program with Debug Tool.

34.2.3 Using Debug Tool to identify the logic problem

Set up and use a debug session for the TDB2 transaction in your CICS region by performing the following steps:

1. Enter transaction ID `DTCN` to update your profile, as shown in Figure 34-10.

```
DTCN                Debug Tool CICS Control - Primary Menu              CICSC22F

Select the combination of resources to debug (see Help for more information)

 Terminal Id      ==> 0007
 Transaction Id   ==>
 Program Id(s)    ==>          ==>          ==>          ==>
                  ==>          ==>          ==>          ==>
 User Id          ==> CHABERT
 NetName          ==>

Select type and ID of debug display device

 Session Type      ==> MFI                MFI, TCP
 Port Number       ==>                    TCP Port
 Display Id        ==> 0007
 Connection Type   ==> SINGLE             Single, Multiple

Generated String:  TEST(ALL,'*',PROMPT,'MFI%0007:CHABERT.DEBUG.PREF(TRAD1)')

Repository String: TEST(ALL,'*',PROMPT,'MFI%0007:CHABERT.DEBUG.PREF(TRAD1)')


PF1=HELP 2=GHELP 3=EXIT 4=SAVE 6=DELETE 7=SHOW 9=OPTIONS
                                                                      05/023
```

Figure 34-10 CICS - DTCN transaction profile

2. Press PF4 to Save.
3. PF3 to Exit from this panel.
4. Enter the transaction ID `TDB2`.

   The debug session is started, as shown in Figure 34-11.

```
COBOL    LOCATION: MYTRADMD ENTRY
Command ===>                                                  Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 2
****************************** TOP OF MONITOR ******************************
0001  1 EQA1252I ********** AUTOMONITOR **********
0002    EQA1945I There are no variables in the statement to display.
***************************** BOTTOM OF MONITOR ****************************
SOURCE: MYTRADMD -1----+----2----+----3----+----4----+----5---- LINE: 7 OF 1587
       7          PROGRAM-ID. MYTRADMD.
       8
       9          ENVIRONMENT DIVISION.
      10
      11          DATA DIVISION.
      12          WORKING-STORAGE SECTION.
      13
      14          01  TASK-DATA.
      15              02 DATA1                      PIC S9(8) COMP.
LOG 0----+----1----+----2----+----3----+----4----+----5----+---- LINE: 92 OF 94
0092 EQA1239I The program is currently entering block MYTRADMD ::> MYTRADMD.
0093  SET PF24 "RETRIEVE" = IMMEDIATE RETRIEVE ;
0094 EQA1239I The program is currently entering block MYTRADMD ::> MYTRADMD.
****************************** BOTTOM OF LOG *******************************
PF  1:PK13-24    2:KEEP       3:QUIT       4:CURRLINE   5:FIND       6:SETBREAK
PF  7:UP         8:DOWN       9:GO 1      10:ZOOM WIN  11:ZOOM LOG  12:GO
                                                                      19/015
```

Figure 34-11 Debug session starting for program MYTRADMD

5. Issue the following commands on the command line to stop the program's execution when the program MYTRADD is invoked:

```
AT APPEARANCE MYTRADD
AT ENTRY MYTRADMD::>MYTRAD;
```

Note: You can also use the one line syntax:

```
AT APPEARANCE MYTRADD perform AT ENTRY MYTRADMD::>MYTRAD; end-perform;
```

6. Run the program using PF12 (Note that PF9 is the default key).
7. Press PF12 repeatedly and enter the appropriate values until the Shares - Buy screen is displayed.
8. Select option **2** Buy Shares and press Enter.
9. In the Shares - Buy screen, enter 5 in the Number of Shares to Buy field, and press Enter.
10. Press PF12 to continue program execution.

    The program stops when the program MYTRADD is invoked.
11. Issue the following command to monitor the value of the NO-SHARES field (the host variable for the column NO_SHARES in the CUSTOMER_DETAILS table):

    ```
    MONITOR LIST NO-SHARES;
    ```

Note: Two other variables are interesting and may also be monitored:

```
MONITOR LIST NO-OF-SHARES-DEC
MONITOR LIST %HEX (SHARES-OVERFLOW)
```

This last one must keep a value of X'00'.

The value of these variables is displayed in the Monitor window, as shown in Figure 34-12.

```
COBOL    LOCATION: MYTRADD :> 691.1
Command ===> _                                                Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 3
****************************** TOP OF MONITOR ******************************
0001  1 NO-SHARES   +0000000000
0002  2 NO-OF-SHARES-DEC   0000
0003  3 %HEX ( SHARES-OVERFLOW )   X'00'
SOURCE: MYTRADD --1----+----2----+----3----+----4----+----5-- LINE: 691 OF 1432
     691            Call 'DFHEI1' using by content x'0e08000007000001000f0f0f6 .
     692       -    '404040' end-call                                         .
     693                                                                      .
     694            .                                                         .
     695        MAINLINE-EXIT.                                                .
     696            EXIT.                                                     .
     697       ************************************************************   .
     698        BUY-SELL SECTION.                                             .
     699            EVALUATE UPDATE-BUY-SELL                                  .
LOG 0----+----1----+----2----+----3----+----4----+----5----+-- LINE: 110 OF 113
0110  MONITOR
0111    LIST NO-OF-SHARES-DEC ;
0112  MONITOR
0113    LIST %HEX ( SHARES-OVERFLOW ) ;
PF 13:PK 1-12   14:CLR KEEP  15:QUIT      16:LIST CSR  17:FIND      18:AT/CLEAR
PF 19:TOP       20:BOTTOM    21:GO 1      22:ZOOM WIN  23:ZOOM LOG  24:RETRIEVE
                                                                     02/015
```

Figure 34-12 Monitoring the values

12. Press PF9 (the default key is PF2) to step through the program one line at a time. As you do, keep monitoring the value of NO-SHARES in the Monitor window.

You see that the value in NO-SHARES is -5, as shown in Figure 34-13, after the record in the CUSTOMER_DETAILS table is read in the READ-CUSTOMER-TABLE paragraph.

```
COBOL    LOCATION: MYTRADD :> 867.1
Command ===> _                                                  Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 3
****************************** TOP OF MONITOR ********************************
0001  1 NO-SHARES  -0000000005
0002  2 NO-OF-SHARES-DEC  0005
0003  3 %HEX ( SHARES-OVERFLOW )   X'00'
SOURCE: MYTRADD --1----+----2----+----3----+----4----+----5-- LINE: 863 OF 1432
     863        *****                          AND   COMPANY = :COMPANY-CUST  .
     864        *****END-EXEC.                                                .
     865             PERFORM SQL-INITIAL UNTIL SQL-INIT-DONE                  1
     866             CALL 'DSNHLI' USING SQL-PLIST3.                          1
     867             IF SQLCODE = 0                                           0
     868                MOVE CLEAN-RETURN       TO RETURN-VALUE               0
     869             ELSE                                                     .
     870              IF SQLCODE = 100                                        0
     871                MOVE CUSTOMER-NOT-FOUND TO RETURN-VALUE               0
LOG 0----+----1----+----2----+----3----+----4----+----5----+---- LINE: 96 OF 99
0096  STEP ;
0097  STEP ;
0098  STEP ;
0099  STEP ;
PF  1:PK13-24    2:KEEP       3:QUIT       4:CURRLINE   5:FIND       6:SETBREAK
PF  7:UP         8:DOWN       9:GO 1      10:ZOOM WIN  11:ZOOM LOG  12:GO
                                                                      02/015
```

Figure 34-13 Monitoring the value in NO-SHARES after table read

13. Press PF9 to check the program flow before the program updates the Customer table.

    The number of shares to be bought is added to the existing value in NO-SHARES in the CALCULATE-SHARES-BOUGHT paragraph. This is done before the table is updated in the UPDATE-CUSTOMER-TABLE paragraph as shown in Figure 34-14.

```
COBOL    LOCATION: MYTRADD :> 783.1
Command ===>                                                    Scroll ===> CSR
SOURCE: MYTRADD --1----+----2----+----3----+----4----+----5-- LINE: 766 OF 1432
     766         BUY-SELL-BUY-FUNCTION-EXIT.                                  .
     767             EXIT.                                                    0
     768        ************************************************************.
     769         CALCULATE-SHARES-BOUGHT SECTION.                            .
     770        * Move new number of shares into i/p Commarea and            .
     771        * customer file write commarea for update                    .
     772             ADD NO-OF-SHARES-DEC TO NO-SHARES.                       1
     773                                                                      .
     774             EVALUATE SHARES-OVERFLOW                                 1
     775               WHEN 0                                                 .
     776                 MOVE SHARES-NORMAL TO NO-OF-SHARES-DEC               1
     777                 MOVE SHARES-NORMAL TO DEC-NO-SHARES                  1
     778               WHEN OTHER                                             .
     779                 MOVE INVALID-BUY TO RETURN-VALUE                     0
     780                 MOVE TOO-MANY-MSG TO COMMENT-FIELD                   0
     781             END-EVALUATE.                                            .
     782         CALCULATE-SHARES-BOUGHT-EXIT.                                .
     783             EXIT.                                                    0
     784        ************************************************************.
PF  1:PK13-24    2:KEEP       3:QUIT       4:CURRLINE   5:FIND       6:SETBREAK
PF  7:UP         8:DOWN       9:GO 1      10:ZOOM WIN  11:ZOOM LOG  12:GO
                                                                      22/019
```

Figure 34-14 CALCULATE -SHARES-BOUGHT section.

The value of NO-SHARES is now 0, as shown in Figure 34-15.

```
COBOL    LOCATION: MYTRADD :> 783.1
Command ===>                                                  Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 3
****************************** TOP OF MONITOR ********************************
0001  1 NO-SHARES  +0000000000
0002  2 NO-OF-SHARES-DEC  X'000000F0'
0003  3 %HEX ( SHARES-OVERFLOW )   X'00'
SOURCE:MYTRADD --1----+----2----+----3----+----4----+----5-- LINE: 781 OF 1432
     781            END-EVALUATE.                                          .
     782         CALCULATE-SHARES-BOUGHT-EXIT.                             .
     783            EXIT.                                                  0
     784         ************************************************************.
     785          CALCULATE-SHARES-SOLD SECTION.                           .
     786         * Move new number of shares into i/p Commarea and          .
     787         * customer file write commarea for update                  .
     788         ************************************************************.
     789         * Commented out to generate an abend in the TDB2 transaction .
LOG 0----+----1----+----2----+----3----+----4----+----5----+--- LINE: 98 OF 111
0098  STEP ;
0099  STEP ;
0100  STEP ;
0101  STEP ;
PF  1:PK13-24    2:KEEP       3:QUIT       4:CURRLINE   5:FIND       6:SETBREAK
PF  7:UP         8:DOWN       9:GO 1      10:ZOOM WIN  11:ZOOM LOG  12:GO
                                                                     22/036
```

Figure 34-15 Monitoring the value in NO-SHARES now in error

Conclusion 1: The buy process actually *zeros* the value; therefore, the display shows zero number of shares.

We continue the debugging session to review the Sell processing portion of the program.

14. Press PF12. The Options panel is displayed.
15. Select option **3** Sell Shares and press Enter.

    Enter 5 in the Number of Shares to Sell field.
16. Press PF9 to step through the program one line at a time. Continue to watch the value of NO-SHARES in the Monitor window.

    You can see that the value of NO-SHARES after the READ-CUSTOMER-TABLE paragraph is executed is 0, as shown in Figure 34-16.

```
COBOL    LOCATION: MYTRADD :> 814.1
Command ===>                                                  Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 3
****************************** TOP OF MONITOR ********************************
0001  1 NO-SHARES  +0000000000
0002  2 NO-OF-SHARES-DEC  0005
0003  3 %HEX ( SHARES-OVERFLOW )   X'00'
SOURCE:MYTRADD --1----+----2----+----3----+----4----+----5-- LINE: 812 OF 1432
     812         * Check whether we have any shares to sell                .
     813            PERFORM READ-CUSTOMER-TABLE THRU READ-CUSTOMER-EXIT.   1
     814            EVALUATE RETURN-VALUE                                  0
     815              WHEN CLEAN-RETURN                                    .
     816                     PERFORM CALCULATE-SHARES-SOLD                 0
     817                     PERFORM UPDATE-CUSTOMER-TABLE                 0
     818                     PERFORM BUILD-RESP-COMMAREA                   0
     819         *          END-IF                                         .
     820              WHEN CUSTOMER-NOT-FOUND                              .
LOG 0----+----1----+----2----+----3----+----4----+----5----+--- LINE: 96 OF 103
0096  STEP ;
0097  STEP ;
0098  STEP ;
0099  STEP ;
PF  1:PK13-24    2:KEEP       3:QUIT       4:CURRLINE   5:FIND       6:SETBREAK
PF  7:UP         8:DOWN       9:GO 1      10:ZOOM WIN  11:ZOOM LOG  12:GO
                                                                     22/017
```

Figure 34-16 Monitoring the value in NO-SHARES after table read

17. Press PF9 to continue executing the program.

You can see the value of NO-SHARES is a negative value (-5) as shown in Figure 34-17, after the SUBSTRACT statement in the CALCULATE-SHARES-SOLD section.

```
COBOL    LOCATION: MYTRADD :> 798.1
Command ===>                                                  Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 3
****************************** TOP OF MONITOR ********************************
0001  1 NO-SHARES  -0000000005
0002  2 NO-OF-SHARES-DEC  0005
0003  3 %HEX ( SHARES-OVERFLOW )   X'00'
SOURCE: MYTRADD --1----+----2----+----3----+----4----+----5-- LINE: 794 OF 1432
     794        *       MOVE TOO-MANY-SHARES-MSG TO COMMENT-FIELD
     795        *    ELSE
     796                SUBTRACT NO-OF-SHARES-DEC FROM NO-SHARES
     797                GIVING NO-SHARES.
     798                MOVE NO-SHARES TO NO-OF-SHARES-DEC.
     799        *    END-IF.
     800        ************************************************************
     801        * End of commented out
     802        ************************************************************
LOG 0----+----1----+----2----+----3----+----4----+----5----+-- LINE: 103 OF 106
0103    LIST %HEX ( SHARES-OVERFLOW ) ;
0104  STEP ;
0105  STEP ;
0106  STEP ;
PF  1:PK13-24    2:KEEP       3:QUIT       4:CURRLINE   5:FIND       6:SETBREAK
PF  7:UP         8:DOWN       9:GO 1      10:ZOOM WIN  11:ZOOM LOG  12:GO
                                                                     22/017
```

Figure 34-17 Monitoring the value in NO-SHARES after calculation

Conclusion 2: It is clear that the problem is program logic. There is no validation of the number of shares held by a customer before the sell is processed.

Let's figure out how to correct this problem. We need to add a validation routine to the program that encapsulates the following logic:

If the shares held by the customer is less than the number of shares to be sold, then the transaction is not performed and a warning message is issued. This stops negative values from appearing in the database.

Example 34-1 shows the updated code in the CALCULATE-SHARES-SOLD paragraph.

Example 34-1 Coding changes in MYTRADD to correct the error

```
IF NO-OF-SHARES-DEC IS GREATER THAN NO-SHARES
 THEN
   MOVE INvaLID-SALE TO RETURN-vaLUE
   MOVE TOO-MANY-SHARES-MSG TO COMMENT-FIELD
ELSE
   SUBTRACT NO-OF-SHARES-DEC FROM NO-SHARES
   GIVING NO-SHARES
   MOVE NO-SHARES TO NO-OF-SHARES-DEC
END-IF.
```

Attention: You may encounter a bad return when buying shares. Monitor the variable SHARES-OVERFLOW and make sure its value is x'00' as shown in Figure 34-18. To change its value, enter the command:

```
MOVE X'00' TO SHARES-OVERFLOW
```

```
COBOL    LOCATION: MYTRADD :> 868.1
Command ===>                                                  Scroll ===> CSR
MONITOR --+----1----+----2----+----3----+----4----+----5----+----6 LINE: 1 OF 3
******************************** TOP OF MONITOR ********************************
0001  1 NO-SHARES  +0000000005
0002  2 NO-OF-SHARES-DEC  0010
0003  3 %HEX ( SHARES-OVERFLOW )   X'00'
SOURCE:MYTRADD --1----+----2----+----3----+----4----+----5-- LINE: 864 OF 1432
     864      _ *****END-EXEC.                                              .
     865            PERFORM SQL-INITIAL UNTIL SQL-INIT-DONE                 1
     866            CALL 'DSNHLI' USING SQL-PLIST3.                         1
     867            IF SQLCODE = 0                                          1
     868               MOVE CLEAN-RETURN       TO RETURN-VALUE              0
     869            ELSE                                                    .
     870             IF SQLCODE = 100                                       0
     871               MOVE CUSTOMER-NOT-FOUND TO RETURN-VALUE              0
     872             ELSE                                                   .
LOG 0----+----1----+----2----+----3----+----4----+----5----+--- LINE: 97 OF 100
0097  STEP ;
0098  STEP ;
0099  STEP ;
0100  STEP ;
PF  1:PK13-24    2:KEEP       3:QUIT       4:CURRLINE   5:FIND       6:SETBREAK
PF  7:UP         8:DOWN       9:GO 1      10:ZOOM WIN  11:ZOOM LOG  12:GO
                                                                     09/015
```

Figure 34-18 Check the SHARES-OVERFLOW variable

Remember: To enable the corrected programs for CICS, first you have to remove the old ones. Follow these instructions:

TSO

Run the setup jobs again to generate new copies of these programs. Remember you also have to rebind the DB2 DBRM.

CICS

Make these copies available. Enter the command:

```
cemt set program(MYTRAD*) new
```

> **Tip:** To ease the work, create a second set of programs, plan, transaction, and DB2 entry, and present either the erroneous program or the correct one.

34.2.4 Using File Manager/DB2 to correct the data

We decide to use File Manager/DB2 to correct the invalid data in the NO_SHARES column in the CUSTOMER_DETAILS table to rectify the problem in the database. Use these steps to do this:

1. Access File Manager/DB2 in your ISPF session.
2. Select option **2** Edit and press Enter.
3. The DB2 Edit panel is displayed, as shown in Figure 34-19.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------------
FM/DB2 (D80G)                        DB2 Edit
Command ===> _____________________________________________________________________

Specify the DB2 Object:
   Location  . . . . . _________________        Database . . ________ (optional)
   Owner . . . . . . . CHABERT          +       Table space  ________ (optional)
   Name  . . . . . . . CUSTOMER DETAILS                                        +

   Row count . . . . . 100        Number of rows to edit

Template:
   Data set name . . . ____________________________________________
   Member  . . . . . . ________

Processing Options:
   Template usage                      Enter "/", "A"lways to select option
   3  1. Above                         _  Edit options
      2. Previous                      _  Edit template
      3. Generate from table           _  Re-edit template
      4. Generate/Replace

                                                                        04/015
```

Figure 34-19 File Manager-DB2 - Table Edit Entry Panel

4. Specify the following information:

   a. The table owner: This is the user ID used at installation time.
   b. The table name.
   c. Select **3** in the Processing Options field.

   The Table Edit panel is displayed, as shown in Figure 34-20.

```
 Process   Options   Utilities   Help
 ------------------------------------------------------------------------------
FM/DB2 (D80G)                        Table Edit                     9 rows fetched
Command ===> _____________________________________________________ Scroll PAGE
TABLE CHABERT.CUSTOMER_DETAILS                                     Format TABL
       CUSTOMER                  COMPANY                  NO_SHARES
       #1                        #2                              #3
       CHARACTER(25)             CHARACTER(20)              INTEGER
       PU--+----1----+----2----> PU--+----1----+----> <---+----1>
000000 ****  Top of data  ****
000001 ANAND                     Headworth_Electrical           25
000002 ANAND                     IBM                             5
000003 RB_DEMO                   Glass_and_Luget_plc            45
000004 RB_DEMO                   Headworth_Electrical           55
000005 RB_DEMO                   IBM                            -5
000006 LARRY                     Casey_Import_Export           100
000007 LARRY                     ShareSelect                    65
000008 E                         Casey_Import_Export             4
000009 E                         IBM                             3
000010 ****  End of data  ****

                                                                        04/015
```

Figure 34-20 File Manager-DB2 - Table Edit Panel

5. Change the value in the NO_SHARES column to 0 for customer RB_DEMO's holdings in IBM.

6. Enter `SAVE` on the command line.

   The change is saved and the message, `Commit issued` is displayed, as shown in Figure 34-21.

```
 Process   Options   Utilities   Help
FM/DB2 (D80G)                      Table Edit                      Commit issued
Command ===> _____________________________________________________ Scroll PAGE
TABLE CHABERT.CUSTOMER_DETAILS                                     Format TABL
       CUSTOMER                  COMPANY                 NO_SHARES
       #1                        #2                            #3
       CHARACTER(25)             CHARACTER(20)             INTEGER
       PU--+----1----+----2----> PU--+----1----+----> <---+----1>
000000 ****  Top of data  ****
000001 ANAND                     Headworth_Electrical         25
000002 ANAND                     IBM                           5
000003 RB_DEMO                   Glass_and_Luget_plc          45
000004 RB_DEMO                   Headworth_Electrical         55
000005 RB_DEMO                   IBM                           0
000006 LARRY                     Casey_Import_Export         100
000007 LARRY                     ShareSelect                  65
000008 E                         Casey_Import_Export           4
000009 E                         IBM                           3
000010 ****  End of data  ****
                                                                 04/015
```

Figure 34-21 File Manager-DB2 - Table Edit Entry Panel with corrected data saved

7. You can check the value you entered by switching to hexadecimal presentation: Enter the command `HEX ON` and press Enter. The data is displayed as shown in Figure 34-22.

```
 Process   Options   Utilities   Help
FM/DB2 (D80G)                      Table Edit
Command ===> _____________________________________________________ Scroll PAGE
TABLE CHABERT.CUSTOMER_DETAILS                                     Format TABL
       CUSTOMER                  COMPANY                 NO_SHARES
       #1                        #2                            #3
       CHARACTER(25)             CHARACTER(20)             INTEGER
       PU--+----1----+----2----> PU--+----1----+----> <---+----1>
000004 RB_DEMO                   Headworth_Electrical         55
       DC6CCDD444444444444444444 C888A99A86C988A98889       0003
       92D4546000000000000000000 851466938D5353399313        0007

000005 RB_DEMO                   IBM                           0
       DC6CCDD444444444444444444 CCD4444444444444444        0000
       92D4546000000000000000000 9240000000000000000        0000

000006 LARRY                     Casey_Import_Export         100
       DCDDE4444444444444444444 C8A8A6C9999A6CA999A4        0006
       3199800000000000000000000 31258D947693D5776930        0004

000007 LARRY                     ShareSelect                  65
       DCDDE4444444444444444444 E8898E8988A444444444        0004
       3199800000000000000000000 28195253533000000000        0001
                                                                 10/014
```

Figure 34-22 Hexadecimal edition

34.3 Summary of Scenario 3

In this chapter we described the various components that make up the CICS and DB2 environments in our system and explained how they are set up.

We reviewed these topics:

1. The processing performed by the CICS DB2 Trader application

2. A process that used Debug Tool, running under CICS, to identify a problem with the logic in the application
3. File Manager/DB2 capability to correct the data that resulted from the problem

35

Using Fault Analyzer and File Manager

In this chapter, we describe the application components that exist in the CICS environment on our system and how they are set up.

We explain the processing that is performed in the CICS Trader application. We force the application to abend and describe, in detail, the steps needed to identify the cause of an abend in the application using Fault Analyzer. We then describe how to manipulate the data to correct the problem using File Manager.

35.1 Set up the components

Two types of components need to be established for this scenario:

- CICS components
- Program products
  - Fault Analyzer
  - File Manager

35.1.1 CICS components

Components used by the Trader application are listed in Table 35-1.

Table 35-1 CICS components of the Trader application scenario

Components	Details	Remarks
Programs	MYTRADMV MYTRADS	CICS COBOL PROGRAM
TranID	MYTD	Transaction associated with the program MYTRADMV
MapSet	NEWTRAD	BMS mapset containing all the maps used by the application
Files	GRACINE.PDPAK.CUSTFILE GRACINE.PDPAK.COMPFILE	VSAM files used by the application
Copybooks	CUSTFILE COMPFILE	File definition for CUSTFILE & COMPFILE

35.1.2 Program products

To use the Problem Determination Tools with this scenario, make sure you have the following output or supporting files for each product.

Fault Analyzer

Ensure Fault Analyzer is correctly installed in your CICS region.

You must have a compiler listing or side file for the programs MYTRADMV and MYTRADS.

If you are not using the supplied batch jobs to compile these programs, make sure you specify the following compiler options:

```
LIST,SOURCE,XREF,MAP
```

File Manager

You need the copybooks that contain the record structure of the VSAM files DEMOS.PDPAK.CUSTFILE and DEMOS.PDPAK.COMPFILE:

Copybooks CUSTFILE and COMPFILE

Make sure you run the DEFVSAM1 batch job to load the VSAM files.

35.2 Walkthrough of the CICS Trader application

The CICS Trader application is used to maintain a stock portfolio held by an individual. This application enables you to:

- Obtain quotes
- Buy shares of a company's stock
- Sell currently held shares of a company's stock

Note: This example was designed to demonstrate the capabilities of the Problem Determination Tools. Therefore, a minimal amount of code was developed. This application does *not* represent real-world securities processing.

Before you start the application, access CICSC001, or your own CICS application region.

Enter the transaction ID MYTD.

The Logon screen of the application is displayed in Figure 35-1. A username and a password are required to access the application.

```
                    Share Trading Demonstration                    TRADER.T001

                     Share Trading Manager: Logon



              Enter your User Name:  RB_DEMO


              Enter your Password:          _



--------------------------------------------------------------------------------
PF3=Exit                                                              PF12=Exit
```

Figure 35-1 Trader application Logon screen

Note: In the Trader application, navigation keys are displayed at the bottom of each screen. PF3 is used to go back to the previous screen (except on the Logon screen) and PF12 is used to terminate the application.

35.2.1 Log on to the application

Log on to the application with a username and password. In this example, enter the username RB_DEMO and the password ITSO.

After you press Enter, the Company Selection screen is displayed, as shown in Figure 35-2. This screen lists the companies you can trade.

```
                    Share Trading Demonstration                    TRADER.T002

                 Share Trading Manager: Company Selection

                        1. Casey_Import_Export

                        2. Glass_and_Luget_plc

                        3. Headworth_Electrical

                        4. IBM


                Please select a company (1,2,3 or 4) :  4


 ------------------------------------------------------------------------------
 PF3=Return                                                          PF12=Exit
```

Figure 35-2 Trader application Company Selection screen

You must select a company to continue with the application. Select 4, IBM, and press Enter. The Options screen is displayed, as shown in Figure 35-3.

```
                    Share Trading Demonstration                    TRADER.T003

                 Share Trading Manager: Options

                        1. New Real-Time Quote

                        2. Buy Shares

                        3. Sell Shares


                Please select an option (1,2 or 3):  1_


 ------------------------------------------------------------------------------
PF3=Return                                                           PF12=Exit
```

Figure 35-3 Trader application Options screen

On this screen, select the trading option you want to perform:

- Obtain real-time quotes for a company.
- Buy additional shares of the company.
- Sell existing shares of the company.

In the following sections we demonstrate each option, in turn.

35.2.2 Obtaining quotes

Select Option **1**, New Real-Time Quote, and press Enter. The Real-Time Quote screen is displayed, as shown in Figure 35-4.

```
_                        Share Trading Demonstration                    TRADER.T004

                     Share Trading Manager: Real-Time Quote

       User Name:     RB_DEMO

       Company Name:  IBM

       Share Values:                      Commission Cost:
          NOW:        00113.00               for Selling:        015
          1 week ago: 00107.00               for Buying:         010
          6 days ago: 00106.00
          5 days ago: 00109.00
          4 days ago: 00111.00
          3 days ago: 00110.00
          2 days ago: 00112.00            Number of Shares Held: 0100
          1 day ago:  00113.00            Value of Shares Held:  000011300.00



 Request Completed OK
 ---------------------------------------------------------------------------
 PF3=Return                                                          PF12=Exit
```

Figure 35-4 Trader application Real-Time Quote screen

This screen displays the price of the company's share over the past seven days, the number of shares held, and the value of those shares based on the current day's price:

- The company's share price is read from the VSAM file DEMOS.PDPAK.COMPFILE (COMPFILE).
- The details of the user's portfolio, (for example, the number of shares held), are read from the VSAM file DEMOS.PDPAK.CUSTFILE (CUSTFILE).

Press PF3 to return to the Options screen.

35.2.3 Buying shares

Select Option **2**, Buy Shares, and press Enter. The Shares Buy screen is displayed, as shown in Figure 35-5.

```
                    Share Trading Demonstration                    TRADER.T005

                    Share Trading Manager: Shares - Buy


                User Name:     RB_DEMO

                Company Name:  IBM


                Number of Shares to Buy:  100_



--------------------------------------------------------------------------------
PF3=Return                                                            PF12=Exit
```

Figure 35-5 Trader application Shares Buy screen

Enter the number of shares you want to buy and press Enter.

The Options screen is re-displayed with a message in the lower, left-hand corner of the screen indicating the status of the transaction. If the process is successful, the value of the number of shares held is updated in the CUSTFILE.

35.2.4 Selling shares

Select Option **3**, Sell Shares, and press Enter. The Shares Sell screen is displayed, as shown in Figure 35-6.

```
                    Share Trading Demonstration                    TRADER.T005

                    Share Trading Manager: Shares - Sell


                User Name:     RB_DEMO

                Company Name:  IBM


                Number of Shares to Sell: 50_



--------------------------------------------------------------------------------
PF3=Return                                                            PF12=Exit
```

Figure 35-6 Trader application Shares Sell screen

Enter the number of shares you want to sell and press Enter.

The Options screen is re-displayed with a message in the lower, left-hand corner of the screen indicating the status of the transaction. If the process is successful, the value of the number of shares held is updated in the CUSTFILE.

35.3 Tracking an abend in the application

To demonstrate the capabilities of the Problem Determination Tools, we force the application to abend and step you through the process of fixing it. Fault Analyzer is used to identify the cause of the abend. File Manager is used to correct an error in a VSAM application file.

Note: We wanted to perform all of the steps in this example and obtain the screen shots at the same time. In reality, the process of writing this section took several days. As a result, there are discrepancies with the dates and times in some of the figures. These are not deliberate errors, and they are not meant to mislead you.

In this example, you attempt to obtain the real-time quotes of IBM for the customer, RB_DEMO. After you select Option 1 on the Options menu of CICS Trader application and press Enter, the application abends. The CICS-issued transaction abend message is shown in Figure 35-7.

```
_                    Share Trading Demonstration                    TRADER.T003

                    Share Trading Manager: Options

                     1. New Real-Time Quote

                     2. Buy Shares

                     3. Sell Shares



                Please select an option (1,2 or 3):  1



 DFHAC2206 11:31:10 A06C001 Transaction MYTD failed with abend ASRA. Updates to
  local recoverable resources backed out.
```

Figure 35-7 Trader application abend on the Options screen

35.3.1 Viewing the abend in Fault Analyzer

Use Fault Analyzer to conduct the analysis of the abend by performing the following steps:

1. Access Fault Analyzer in your ISPF session.

   The fault history file is displayed, as shown in Figure 35-8.

```
  File  Options  View  Services  Help
 ------------------------------------------------------------------------
 IBM Fault Analyzer - Fault Entry List                        Line 1 Col 1 80
 Command ===> _____________________________________________ Scroll ===> CSR

 Fault History File or View  : (CICSHIST) CICS HISTORY FILES FOR USER GRACINE

 {The following line commands are available: ? (Query), V (View real-time
 report), I (Interactive reanalysis), B (Batch reanalysis), D (Delete).}

    Fault_ID History_File_DSN                                System   Job/Tran Abe
 __   F00031 IDI.HIST.CICSC22F                               STLABF6  TDB2     403
 __   F00030 IDI.HIST.CICSC22F                               STLABF6  TDB2     403
 __   F00029 IDI.HIST.CICSC22F                               STLABF6  MYTD     403
 __   F00028 IDI.HIST.CICSC22F                               STLABF6  MYTD     403
 __   F00027 IDI.HIST.CICSC22F                               STLABF6  MYTD     403
 __   F00026 IDI.HIST.CICSC22F                               STLABF6  MYTD     403
 __   F00025 IDI.HIST.CICSC22F                               STLABF6  CEOT     ATN
 __   F00024 IDI.HIST.CICSC22F                               STLABF6  MYTD     ASR
 __   F00002 IDI.HIST.CICSC23G                               STLABF7  CEMT     ATN
 __   F00009 IDI.HIST.CICSC22F                               STLABF6  TDB2     AEI
 __   F00002 IDI.HIST.CICSC31G                               STLABF7  CEDA     ATN
  F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
  F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 35-8 Fault Analyzer fault history file

Tip: If you need to switch to a different fault history file, do the following:

a. Select Options → Change Fault History File Options.

b. Move your cursor to the appropriate file on the list and press Enter.

c. Press PF3 to display the fault history records.

Each abend is assigned a fault ID when it is recorded in the fault history file.

You can identify an abend by knowing the transaction ID and the date and time at which it occurred. In this example, the fault ID is F00028. An ASRA is listed in the Abend column.

2. Enter `V` in the line command area next to the fault ID to view the details of this abend.

   The real-time analysis synopsis report is displayed, as shown in Figure 35-9. It is generated at the time of the abend.

```
  File  View  Services  Help
Real-Time Report                                                Line 1 Col 1 80
Command ===> _____________________________________________ Scroll ===> CSR
- Collapse all / + Expand all
*****************************************************************************
* IBM Fault Analyzer for z/OS V5R1M0 (UQ94262 2004/10/25)
*
* (C) Copyright IBM Corp. 2000, 2004.  All rights reserved.
*****************************************************************************

TRANID: MYTD      CICS ABEND: 4038                STLABF6   2004/11/26  12:03:


- <H1> I B M   F A U L T   A N A L Y Z E R   S Y N O P S I S


A CICS abend 4038 occurred in an undetermined module.


- <H1> I B M   F A U L T   A N A L Y Z E R   E V E N T   S U M M A R Y

 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right
```

Figure 35-9 Fault Analyzer real-time analysis report synopsis

3. Look more closely at the report in Figure 35-9. You can see:
   a. Program MYTRADS experienced the abend
   b. The detail of the abend; in this example it is a data exception
   c. A short explanation of the abend
   d. An attempt to identify the instruction that caused the abend

The source statement cannot be identified because the compiler listing (or side file) for the program MYTRADS was not available to Fault Analyzer when the program abended.

35.3.2 Initiating interactive reanalysis for the abend

You must re-analyze the dump to identify the source statement in the program MYTRADS that caused the abend. This can be done in one of two ways: interactive reanalysis or batch reanalysis.

Interactive reanalysis

The abend is re-analyzed and the reports are displayed online in the ISPF session.

Batch reanalysis

The abend is re-analyzed in a batch job and the reports can be viewed in the spool or can be directed to a data set. This does not tie up your ISPF session.

In this example, you decide to perform an interactive reanalysis.

1. Enter **I** in the line command area next to the fault ID. Press Enter.

   The Interactive Options panel is displayed, as shown in Figure 35-10.

```
  File  Options  View  Services  Help
 ┌──────────────────────── Interactive Reanalysis Options ────────────────────────┐
I │                                                                                │ 0
C │ Press PF3 to save options or PF12 to cancel.                                   │
  │                                                                                │
F │ General Options:                                                               │
  │  Options line for interactive reanalysis:                                      │
{ │  LA(ENU)                                                                       │
r │  Redisplay this panel before each reanalysis Y  (Y/N)                          │
  │  Display panel to alter allocated data sets  Y  (Y/N)                          │
  │  Prompt before opening a SYSMDUMP  . . . . . Y  (Y/N)                          │ e
  │                                                                                │ 3
  │ Reanalysis Options Data Set Control:                                           │ 3
  │  Options data set name 'GRACINE.SIDIOPTS'                                     │ 3
i │  Options member name   OPTS1___                                                │ 3
  │  Use this data set during reanalysis . . . . Y  (Y/N)                          │ 3
  │  Edit the options data set before reanalysis Y  (Y/N)                          │ 3
  │                                                                                │ N
  │  F1=Help    F3=Exit   F12=Cancel                                               │ R
  └────────────────────────────────────────────────────────────────────────────────┘ N
     F00009 IDI.HIST.CICSC22F                          STLABF6  TDB2     AEI
     F00002 IDI.HIST.CICSC31G                          STLABF7  CEDA     ATN
 F1=Help      F3=Exit      F4=MatchCSR  F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right    F12=MatchALL
```

Figure 35-10 Initiate interactive reanalysis for fault

2. Enter the name of the Options Data Set that contains the name of your compiler listing or side file. (Recall, the compiler listing or side file is required by Fault Analyzer to identify the source line instruction in the program.)

   The summary panel of the Interactive Analysis report, shown in Figure 35-11, is displayed after reanalysis is complete.

```
  File  View  Services  Help
 ──────────────────────────────────────────────────────────────────────────────
Interactive Reanalysis Report                                   Line 1 Col 1 80
Command ===> ____________________________________________ Scroll ===> CSR
TRANID: MYTD      CICS ABEND: 4038            STLABF6   2004/11/26  12:03:13

Fault Summary:
CICS abend 4038.

Select one of the following options and press Enter to access further fault
information:
_  1.  Synopsis
   2.  Event Summary
   3.  System-Wide Information
   4.  Abend Job Information
   5.  Options in Effect

{Fault Analyzer maximum storage allocated: 2.15 megabytes.}

*** Bottom of data.


 F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
 F8=Down     F10=Left     F11=Right
```

Figure 35-11 Summary page Interactive Analysis report

3. Select **1** to view the Synopsis section.

   The Synopsis section starts with the same description that is shown in the real-time analysis report that you obtained using the **v** line command.

4. Scroll to the bottom of this section to view the statement that caused the error.

The values of the variables at the time of the abend are also displayed, as shown in Figure 35-12.

```
      ------ ------
      000801 n/a                MULTIPLY DECIMAL-SHARE-VALUE BY DEC-NO-SHARES
      000802 n/a                  GIVING DECIMAL-SHARE-VALUE

    The COBOL source code for data fields participating in the failure:

      Source List
      Line # Stmt #
      ------ ------
      000111 n/a          *   03 CONVERT1.
      000112 n/a          *      05 NO-SHARES-CUS          PIC X(4).
      000113 n/a          *   03 CONVERT2 REDEFINES CONVERT1.
      000114 n/a                 05 DEC-NO-SHARES          PIC S9(7) COMP-3.
      000219 n/a                    07 DECIMAL-SHARE-VALUE PIC 9(11)V99.

    Data field values at time of abend:

      DEC-NO-SHARES         = X'F0F1F0F0' *** Cause of error ***
      DECIMAL-SHARE-VALUE   = 0000000011300
```

Figure 35-12 Interactive Analysis Synopsis section displaying cause of error

5. You determine the cause of the error by looking at the values of the variables.

   In this example, the problem is caused by invalid data. It is clearly indicated that the field DEC-NO-SHARES has an invalid value. This field is defined in WORKING-STORAGE as a packed decimal field, yet it contains numeric data in an invalid format.

6. Press PF3 to exit from this panel and return to the Summary panel.

7. Select **2**, Point-of-failure.

   The Point of Failure section starts with a summary similar to the Synopsis section, but without the textual description.

8. Scroll down Figure 35-13 until you see Associated Storage Areas.

```
  File  View  Services  Help
 ____________________________________________________________________________
 Event 5 of 6: EXEC CICS *** Point of Failure ***            Line 56 Col 1 80
 Command ===> ____________________________________________ Scroll ===> CSR
 TRANID: TDB2      CICS ABEND: AEIS              STLABF6   2004/11/19  09:10:05
   R7:  001800D0 (Module MYTRADMD program MYTRADMD LINKAGE SECTION BLL=0001 +
                 X'0', symbol DFHEIBLK, source line # 599 )
   R8:  13DB517C (Module MYTRADMD program MYTRADMD + X'13C')
   R9:  12E9A840 (56006592 bytes of storage addressable)
   R10: 12E9AB60 (Module MYTRADMD program MYTRADMD WORKING-STORAGE SECTION
                 BLW=0000 + X'0', symbol TASK-DATA, source line # 14 )
   R11: 13DB693C (Module MYTRADMD program MYTRADMD + X'18FC', source line # 906
                 )
   R12: 12E988A8 (56014680 bytes of storage addressable)
   R13: 12E99850 (56010672 bytes of storage addressable)
   R14: 93DB725C (Module MYTRADMD program MYTRADMD + X'221C', source line #
                 1141 )
   R15: 00000000 (2048 bytes of storage addressable)

 Associated Control Blocks

 User EXEC Interface Block (EIB) at Address 001800D0 :
  F1=Help      F3=Exit      F4=Dsect     F5=RptFind   F6=Actions   F7=Up
  F8=Down     F10=Left     F11=Right
```

Figure 35-13 Interactive Analysis Point of Failure section

9. Place the cursor over Associated Storage Areas and press Enter.

   This displays a listing of the WORKING-STORAGE section of the program:

– The data definitions are shown on the left-hand side of the listing.
– The data values, in character format, start on the right-hand side of the listing.
– The data values, in hexadecimal format, are on the far right-hand side of the listing.

10. Issue the following command to find the field in error:

```
Find 'DEC-NO-SHARE'
```

11. Scroll up until you find the level-01 group item that contains this field,
12. Split the ISPF screen and view this program's (MYTRADS) compiler listing to find out how this level-01 group item is loaded with data values.

> **Note:** If you did not retain a compiler listing as output from the batch compile, refer to the source code directly.

In this example, CUSTOMER-IO-BUFFER is loaded from a CICS READ INTO statement as shown in Figure 35-14.

```
  Menu  Utilities  Compilers  Help
 ------------------------------------------------------------------------------
BROWSE    CHABERT.TRADER.COBLIST(MYTRADS)          Line 00000636 Col 001 080
Command ===> ____________________________________________ Scroll ===> CSR
 000545                 * Build record key
 000546                      MOVE USERID TO CUST-NM  OF CUSTOMER-IO-BUFFER
 000547                      MOVE '.' TO KEYREC-DOT
 000548                      MOVE COMPANY-NAME TO COMP-NM OF CUSTOMER-IO-BUFFER
 000549                 *EXEC CICS READ
 000550                 *          FILE('CUSTFILE')
 000551                 *          INTO(CUSTOMER-IO-BUFFER)
 000552                 *          LENGTH(LENGTH OF CUSTOMER-IO-BUFFER)
 000553                 *          RIDFLD(KEYREC)
 000554                 *          NOHANDLE
 000555                 *          END-EXEC
 000556                      Move LENGTH OF CUSTOMER-IO-BUFFER to dfhb0020
 000557                      Call 'DFHEI1' using by content x'0602f0002700008000
 000558                 -    '404040' by content 'CUSTFILE' by reference
 000559                      CUSTOMER-IO-BUFFER by reference dfhb0020 by referen
 000560                      end-call
 000561
 000562
 F1=Help    F2=Split   F3=Exit    F5=Rfind   F7=Up      F8=Down    F9=Swap
F10=Left   F11=Right  F12=Cancel
```

Figure 35-14 Compiler listing showing CUSTOMER-IO-BUFFER being loaded

13. Swap back to the Fault Analyzer panel.
14. The key of the record in the VSAM file is displayed in the Associated Storage Areas panel. It contains the customer name and the company name.
15. Find the key field of the record, then locate the value in the data buffer. You may have to scroll to the right to see the entire value. In this example, the values are RB_DEMO and IBM. Now that you have determined the problem is with data in the CUSTFILE, you can correct the error with the help of File Manager.

35.3.3 Using File Manager to correct a problem with data

Use File Manager to correct the invalid data in the application file by performing the following steps:

1. Access File Manager in your ISPF session.
2. Go to Data Set Edit (Option **2**).

3. Enter the VSAM Data set name and the Copybook Data set name, as shown in Figure 35-15.

```
 Process   Options   Help
File Manager                    Edit Entry Panel
Command ===> ____________________________________________________________

Input Partitioned, Sequential or VSAM Data Set:
   Data set name . . . . . 'DEMOS.PDPAK.CUSTFILE'
   Member  . . . . . . . . ________     (Blank or pattern for member list)
   Volume serial . . . . . ______       (If not cataloged)

Copybook or Tenplate:
   Data set name . . . . . 'DAVIN7.WORK.SOURCE'
   Member  . . . . . . . . CUSTFILE

Processing Options:
   Copybook/tenplate usage            Enter "/" to select option
   1  1. Above                        _  Edit copybook or tenplate
      2. Previous
      3. None
```

Figure 35-15 File Manager Edit Entry panel

Note: You must disable and close the CUSTFILE in the CICS region before you attempt to edit it. If you do not, File Manager will display the following error message when you edit the file:

```
VSAM OPEN RC X'08', Error Code X'A8
```

4. There are two ways to locate the record in error:

   a. Issue the `FE` command to locate records containing fields with errors.

   Note: This will find the initialization record at the start of a file. To bypass this record, press PF5 (RFIND) repeatedly.

   Figure 35-16 depicts the panel that is displayed when this method is used.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 File Manager                    Data Set Browse
 Command ===> ____________________________________________________ Scroll CSR
 RBA 2992        Key 'RB_DEMO                        '            Format TABL
 VOLSER DAVS9A   Type KSDS   DSNAME DEMOS.PDPAK.CUSTFILE

                    NO-SHARES DEC-NO-SHARES BUY-FROM BUY-FROM-NO BUY-TO   BUY-T
  RBA        Len    #8                 #10 #11      #12         #13      #14
 000002992   136    ....              *****
 000003128   136    1000               1000
 000003264   136    1100               1100
 000003400   136    0000               0000
 000003536   136    0200               0200
 000003672   136    0400               0400
 000003808   136    0440               0440
 ****  End of data  ****
```

Figure 35-16 record located as a result of the FE command

b. Issue the `FIND` command for the record key. Figure 35-17 depicts the panel that is displayed when this method is used.

   In this example, the record with a key of RB_DEMO is the one with the error.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 File Manager                    Data Set Edit
 Command ===> ____________________________________________________ Scroll CSR
 DSNAME DEMOS.PDPAK.CUSTFILE                                      Format TABL
 VOLSER DAVS9A  Type KSDS    Refresh on save N__

        CUST-NM                                                    KEYREC-DOT
        #4                                                         #5
 000023 RB_DEMO                                                    .
 000024 TSDEMO                                                     .
 000025 TSDEMO                                                     .
 000026 TSDEMO                                                     .
 000027 TSDEMO                                                     .
 000028 TSDEMO                                                     .
 000029 TSDEMO                                                     .
 000030 ****  End of data  ****
```

Figure 35-17 RB_DEMO record located as a result of the FIND command

5. Press PF2 (ZOOM) to display the record in SNGL format, as shown in Figure 35-18.

   File Manager displays the invalid data as a string of highlighted asterisks.

   The data is displayed in character format, but to edit a packed decimal field, you must switch to hexadecimal mode.

```
 Process   Options   Help
File Manager                    Data Set Edit
Command ===> _____________________________________________ Scroll CSR

DSNAME DEMOS.PDPAK.CUSTFILE                                Format SNGL
VOLSER DAVS9A  Type KSDS          Top Field is 1    of 14    in Record 23
Ref Field name      Data
 #4 CUST-NM       : RB_DEMO

 #5 KEYREC-DOT    : .
 #6 COMP-NM       : IBM
 #7 DEC-NO-SHARES : ********
 #8 BUY-FROM      :
 #9 BUY-FROM-NO   :
#10 BUY-TO        :
#11 BUY-TO-NO     :
#12 SELL-FROM     :
#13 SELL-FROM-NO  :
#14 SELL-TO       :
#15 SELL-TO-NO    :
#16 ALARM-PERCENT :
****  End of record  ****
```

Figure 35-18 record in SNGL format edit

6. Enter `HEX ON` in the command line.

   The record is displayed in hexadecimal mode as shown in Figure 35-19.

```
 Process   Options   Help
File Manager                    Data Set Edit
Command ===> _____________________________________________ Scroll CSR

DSNAME DEMOS.PDPAK.CUSTFILE                                Format SNGL
VOLSER DAVS9A  Type KSDS          Top Field is 5    of 14    in Record 23
Ref Field name      Data
 #7 DEC-NO-SHARES : ********
                  : FFFF
                  : 0100

 #8 BUY-FROM      :
                  : 44444444
                  : 00000000

 #9 BUY-FROM-NO   :
                  : 4444
                  : 0000

#10 BUY-TO        :
                  : 44444444
                  : 00000000
```

Figure 35-19 File Manager displaying invalid data as highlighted asterisks

7. Scroll down to the field in the record that must be changed. In this example, it is DEC-NO-SHARES.

   The data in DEC-NO-SHARES is character value 100 (displayed as F0F1F0F0). However, the field is defined as packed decimal.

8. Correct the data so that it matches the characters shown in Figure 35-20.

```
 Process   Options   Help
 ------------------------------------------------------------------------------
 File Manager                    Data Set Edit
 Command ===> _____________________________________________________ Scroll CSR

 DSNAME DEMOS.PDPAK.CUSTFILE                                   Zoom Format SNGL
 VOLSER DAVS9A  Type KSDS          Top Field is 5    of 14    in Record 23
 Ref Field name       Data
  #7 DEC-NO-SHARES : _0000100
                   : 0010
                   : 000C

  #8 BUY-FROM      :
                   : 44444444
                   : 00000000

  #9 BUY-FROM-NO   :
                   : 4444
                   : 0000

 #10 BUY-TO        :
                   : 44444444
                   : 00000000
```

Figure 35-20 File Manager displaying corrected data as highlighted fields

a. Clear the field of asterisks (ERASE EOF).

b. Enter the correct value in the field (100).

c. Press Enter.

> **Note:** Once any data in the record is changed, all of the fields associated with the record are highlighted.

9. Enter `HEX OFF` in the command line to return the fields to character format.

10. Press PF3 to save the changes and to exit from the edit session.

35.3.4 Running the application after the fix

Once you have finished correcting the invalid data in the CUSTFILE, again access the Trader application in CICS.

> **Note:** Do not forget to enable and open the CUSTFILE in the CICS region.

Obtain a real-time quote in IBM for customer RB_DEMO.

This results in a successful execution, and the screen shown in Figure 35-21 is displayed.

```
                          Share Trading Demonstration                TRADER.T004

                     Share Trading Manager: Real-Time Quote

      User Name:     RB_DEMO

      Company Name:  IBM

      Share Values:                     Commission Cost:
         NOW:        00113.00              for Selling:        015
         1 week ago: 00107.00              for Buying:         010
         6 days ago: 00106.00
         5 days ago: 00109.00
         4 days ago: 00111.00
         3 days ago: 00110.00
         2 days ago: 00112.00           Number of Shares Held: 0100
         1 day ago:  00113.00           Value of Shares Held:  000011300.00



Request Completed OK
--------------------------------------------------------------------------------
PF3=Return                                                            PF12=Exit
```

Figure 35-21 Trader application: Real-time quote of user RB_DEMO

35.4 Summary of Scenario

In this chapter we described the various components that make up the CICS environment in our system and how they are set up.

We reviewed the processing performed by the CICS Trader application.

We detailed a process whereby Fault Analyzer was used to identify the cause of an abend in the application. We continued with a description of File Manager's capability to identify and correct the data that caused the problem.

36

Application Monitor and CICSplex configuration

In the scenarios described in this chapter, all the transaction are running on a CICSplex environment. These scenarios were designed to highlight features of Application Monitor in a brief, but effective manner. In this chapter, we use the TRADER application, but defined as remote (see chapter 31 for a complete description) and we cover:

- Application Monitor and Transaction Routing
  - System configuration
  - Application Monitor set-up
  - Application Monitor reports
- Application Monitor and Function Shipping
  - System configuration
  - Application Monitor set-up
  - Application Monitor reports
- Application Monitor and Dynamic Program Link
  - System configuration
  - Application Monitor set-up
  - Application Monitor reports
- Conclusions
  - Transaction Routing
  - Function Shipping
  - Dynamic Program Link

36.1 Application Monitor and Transaction Routing

In this scenario we use a Static Transaction Routing between two CICS running in the same LPAR.

When we enter a transaction code for a transaction that is in a remote system, a transaction is attached in the TOR that executes a CICS-supplied program known as the relay program. This program provides the communication mechanism between the terminal and the remote transaction.

Because CICS executes the relay program, the transaction is called a relay transaction.

When the relay transaction is attached, it acquires a session and sends a request to the remote system to cause the “real” user transaction to be started.

36.1.1 System configuration

We have a classic CICSplex configuration, where CICSC22F is our TOR (Terminal Owning Region) and CICSC31F is our AOR/DOR (Application Owning Region and Data Owning Region). The TRAR transaction is the same as a TRAD transaction but defined as remote.

Figure 36-1 illustrates our configuration.

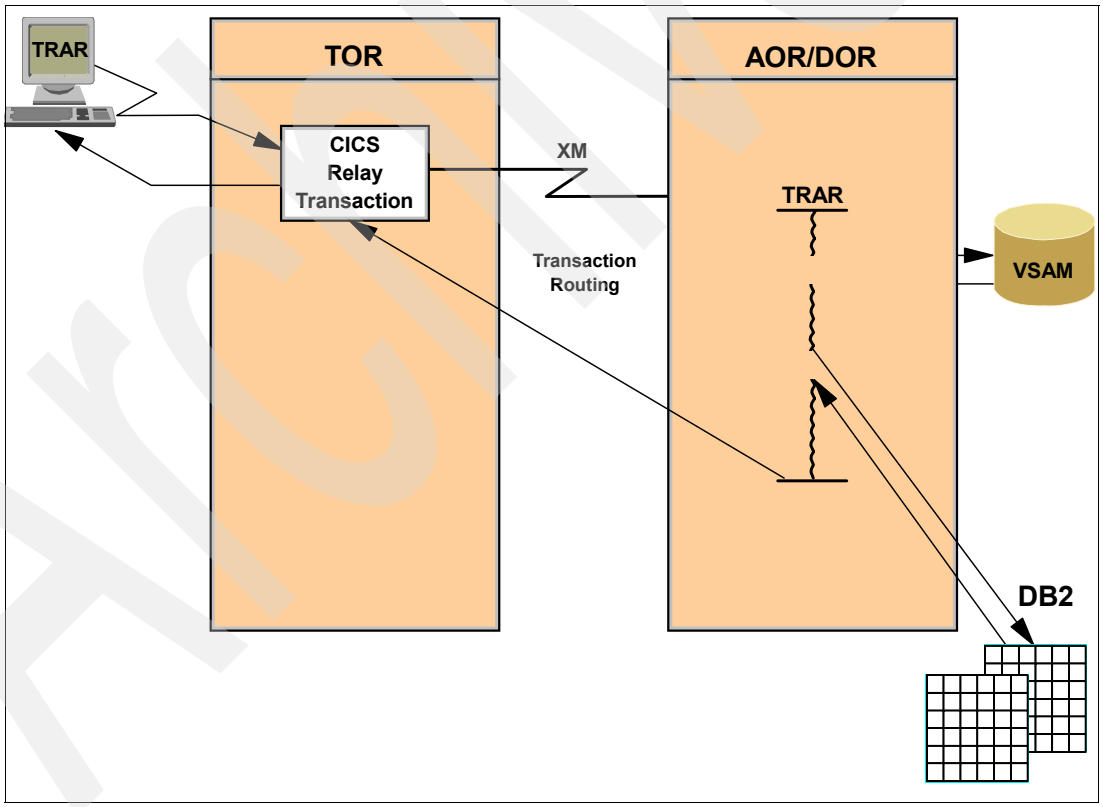


Figure 36-1 CICSplex: Transaction Routing configuration

36.1.2 Application Monitor set-up

In order to monitor our CICSplex configuration, we use the CICSPLEX Named Set. The following three figures, Figure 36-2 through Figure 36-4 illustrate how to create a Named Set,

using Application Monitor option **7.2** Create, Edit and Delete Named Sets; and how to Launch a Named Set, using Application Monitor's option **3** Analyze a Named Set of Jobs.

```
-------- IBM AM for z/OS - Create, Edit and Delete Named Sets  Row 1 to 2 of 2


Enter a line command to process a Named Set

  N - New (Create a new Named Set primed from the selected Named Set)
  E - Edit (Update the contents of the Named Set)
  D - Delete (Delete the Named Set)

C Name     Description                          Created             Count
_   ?      Blank Template for new Named Set     2004/10/27 14:28:39     0
_ CICSPLEX CICS TS22 (TOR) CICS TS 31 (AOR)     2004/10/22 17:07:38     2
****************************** Bottom of data ******************************









Command ===> _____________________________________________ Scroll ==> CSR
                                                                   24/015
```

Figure 36-2 AM: Create, Edit, and Delete Named Sets (Panel ID FBIP7200)

```
-------- IBM AM for z/OS - Create and Edit a Named Set ------- Row 1 to 3 of 8


Fill in the Named Set's details and select an option below

  Named Set Name ==> CICSPLEX
  Description    ==> CICS TS22 (TOR) CICS TS 31 (AOR)


  1 Save the Named Set and exit
  X Go back without saving (PF3)

Enter END (PF3) to go back without saving


Use line commands to define the jobs in the Named Set
A  - Add a new entry      E  - Edit an entry        D  - Delete an entry

C Job      Description
_ CICSC31F CICS TS 31 (AOR - FOR)
_ CICSC22F CICS TS 22 (TOR)
_

Option ===> _____________________________________________ Scroll ==> CSR
                                                                   24/014
```

Figure 36-3 AM: Create and Edit a Named Set (Panel ID FBIP72C0)

```
-------- IBM AM for z/OS - Specify a Named Set Entry -------------------------


Named Set CICSPLEX - CICS TS22 (TOR) CICS TS 31 (AOR)

Fill in the Named Job details below and then select an option

   Job Name    ==> CICSC22F
   Description ==> CICS TS 22 (TOR)
   Step Name   ==> *       Proc Step Name ==> *
   CICS Tran   ==> ___     or IMS Msg Id  ==> ___
   System      ==> STLABF6
   Enter / to select special types of analysis:
     Delays        /
     DB2 activity: /      DB2 Subsystem Id ==> DB1E
     Data Sets:    /


  1 Save the changes to this Named Set
  X Go back without saving changes (PF3)


Enter END (PF3) to exit without saving
Option ===> _____________________________________________________________
                                                                  24/014
```

Figure 36-4 AM: Specify a Named Set Entry (Panel Id FBIP72E0)

To monitor our CICSPLEX Named Set, we use Application Monitor's option **3** Analyze a Named Set of Jobs as shown in Figure 36-5 and Figure 36-6.

```
-------- IBM AM for z/OS - Specify a Named Set to Analyze ---- Row 1 to 1 of 1


Enter a line command to process a Named Set:
Go to the Administration screen to Create, Edit or Delete Named Sets

  L Launch (Initiate monitoring of all the Jobs in the Named Set)

C Name     Description                          Created             Count
_ CICSPLEX CICS TS22 (TOR) CICS TS 31 (AOR)     2004/10/22 17:07:38   2
****************************** Bottom of data ********************************









Command ===> ___________________________________________________ Scroll ==> CSR
                                                                  24/015
```

Figure 36-5 AM: Specify a Named Set to Analyze (Panel Id FBIP3000)

```
FBIP1A00 IBM AM for z/OS - Active Monitor Sessions ---------------------------
         2004/11/02 15:43:32 to 2004/11/02 15:43:35       Row 1 of 2

  Named Set: CICSPLEX
System   Jobname  Stepname Pgm  CICS IMS  DB2  Dly  Dsn  CPU%   Sample Duration
-------- -------- -------- Rpts Rpts Rpts Rpts Rpts Rpts ------  Count --------
STLABF6  CICSC31F CICSC31F    0 YES       YES  YES  YES    0.00      0 00:00:01
STLABF6  CICSC22F CICSC22F    0 YES       YES  YES  YES    0.00      0 00:00:01
****************************** BOTTOM OF DATA ******************************



Command ===> _____________________________________________ Scroll ==> CSR
                                                                    24/015
```

Figure 36-6 AM: Active Monitor Sessions (Panel Id FBIP1A00)

Because both of our CICS regions are active when we enter the **L** line command to initiate the real-time monitoring, Application Monitor shows the Active Monitor Sessions display shown in Figure 36-6.

Now we log on to CICSC22F and we start the remote TRAR transaction.

36.1.3 Application Monitors reports

We only show the main reports here, to point out what is specific to a Transaction Routing configuration.

Application Monitor reports for our TOR

Because we are using Transaction Routing, our TRAR transaction is handled by a CICS-supplied transaction called the relay transaction. Therefore, there are no special reports to display for the TOR region. CICS, DB2, and DSN reports have no information concerning our user application, as shown in Figure 36-7, Figure 36-8, and Figure 36-9.

```
FBIP1B00 IBM AM for z/OS - CICS Transactions ---------------------------------
         2004/11/03 10:43:48 to 2004/11/03 11:25:21       Row 1 of 2
History Data Set: CHABERT.AM.CICSC22F.D2004308.T1043474
   CICS Job: CICSC22F     CICS APPLID: CICSC22F          System: STLABF6

Tran Program   Count  Busy%
---- -------- ------ ------
CEX2 DFHD2EX2      1  99.99 ==============>
CEMT DFHEMTP       4   0.00
****************************** BOTTOM OF DATA ********************************

Command ===> _____________________________________________ Scroll ==> CSR
                                                                     24/015
```

Figure 36-7 AM: TR-TOR CICS Transaction

```
FBIP1C00 IBM AM for z/OS - SQL Analysis --------------------------------------
         2004/11/03 10:43:48 to 2004/11/03 10:45:34       Row 0 of 0
History Data Set: CHABERT.AM.CICSC22F.D2004308.T1043474
    Jobname: CICSC22F        Stepname: CICSC22F          System: STLABF6
    LU Name:                 DB2 SSID:                   Module:
  Plan Name:                 Auth Id.:                 Oper Id.:
DB2  Auth-ID  Plan     DBRM     St No O SQLRC Type     Count Rel. DB2 CPU Time
---- -------- -------- -------- ----- - ----- -------- ----- <--------------->
****************************** NO DATA AVAILABLE *****************************

Command ===> _____________________________________________ Scroll ==> CSR
                                                                     24/015
```

Figure 36-8 AM: TR-TOR SQL Analysis

```
FBIP1C00 IBM AM for z/OS - Data Set Analysis Report --------------------------
         2004/11/03 10:43:48 to 2004/11/03 10:45:34       Row 29 of 42        >
History Data Set: CHABERT.AM.CICSC22F.D2004308.T1043474
     System: STLABF6          Jobname: CICSC22F        Stepname: CICSC22F
   Duration: 00:01:46

DD        Acc  Rec    Rec CI/Blk       EXCP Data Set
Name     Meth Form Length   Size      Count Name
-------- ---- ---- ------ ------ ---------- ----------------------------------
INTR     BSAM F        80     80          0 CICPROD.CICSC22F.STC00472.D0000108.
CAFF     BSAM V       133    137          0 CICPROD.CICSC22F.STC00472.D0000109.
CEEMSG   BSAM V       161    165          0 CICPROD.CICSC22F.STC00472.D0000110.
CEEOUT   BSAM V       133    137          0 CICPROD.CICSC22F.STC00472.D0000111.
CRPO     BSAM V       133    137          0 CICPROD.CICSC22F.STC00472.D0000112.
CCTLDD   BPAM U           32,760         22 IMSVS.IM8F.SDFSRESL
ERCFILE  VSAM             18,432          0 CHABERT.FBI.TS22.FILE
DFHDBFK  VSAM              2,048          0 CICSVS.C22F.DFHDBFK
DFHLRQ   VSAM              4,608          0 CICSVS.C22F.DFHLRQ
DFHCSD   VSAM             18,432          0 CICSVS.C22F.DFHCSD
DFHGCD   VSAM              8,192          0 CICSVS.C22F.DFHGCD
DFHLCD   VSAM              2,048          0 CICSVS.C22F.DFHLCD
DFHINTRA VSAM              4,096          0 CICSVS.C22F.DFHINTRA
DFHTEMP  VSAM              4,096          0 CICSVS.C22F.DFHTEMP
Command ===> _____________________________________________ Scroll ==> CSR
                                                                    24/015
```

Figure 36-9 AM: TR-TOR Data Set Analysis Report

Application Monitor reports for our AOR

Because the AOR runs the user application as if it was a local application, all the reports are there. From the Application Monitor point of view it seems that all CICS resources are local in the AOR.

Figure 36-10 shows the TR-AOR CICS transactions.

```
FBIP1B00 IBM AM for z/OS - CICS Transactions ---------------------------------
         2004/11/03 10:43:48 to 2004/11/03 11:42:48       Row 1 of 3
History Data Set: CHABERT.AM.CICSC22F.D2004308.T1043474
   CICS Job: CICSC31F     CICS APPLID: CICSC31F          System: STLABF6

Tran Program   Count  Busy%
---- -------- ------ ------
CEX2 DFHD2EX2      1  99.95 ==============>
TRAR MYTRADMD     61   0.04 .
CWBG DFHWBGB       1   0.00
****************************** BOTTOM OF DATA ******************************



Command ===> _____________________________________________ Scroll ==> CSR
                                                                    24/015
```

Figure 36-10 AM: TR-AOR CICS Transactions

The report shows the TRAR transaction MYTRADMD was the first program called by the application, but is it the only one? To get an answer we can drill down to the CICS Calls in a Transaction report by placing the cursor on the transaction name and pressing Enter.

Figure 36-11 shows the CICS Calls in a Transaction report.

```
FBIP1C00 IBM AM for z/OS - CICS Calls in a Transaction ------------------------
         2004/11/03 10:43:48 to 2004/11/03 11:52:12       Row 73 of 90
History Data Set: CHABERT.AM.CICSC22F.D2004308.T1043474
   CICS Job: CICSC31F     CICS APPLID: CICSC31F          System: STLABF6
Transaction: TRAR          CICS calls: 45

Program  Offset   Count Activity                     Total time
-------- -------- ----- ---------------------- ---------------
MYTRADMD 0000130A     7 EXEC CICS                    0.000117 .
MYTRADMD 0000130C     7   RETURN                     0.000000
MYTRADMD 00000F98     7 EXEC CICS                    0.000000
MYTRADMD 00000F9A     7   RECEIVE MAP                0.000523 .
MYTRADMD 0000305E     5 EXEC CICS                    0.000090 .
MYTRADMD 00003060     5   SEND                       0.000139 .
MYTRADMD 0000308E     5 EXEC CICS                    0.000081 .
MYTRADMD 00003090     5   RETURN                     0.000000
MYTRADD  00001CBC     1 EXEC SQL                     0.000527 .
MYTRADD  00001CBE     1   INSERT                     0.004154 ==>
MYTRADD  00001D4C     3 EXEC SQL                     0.000162 .
MYTRADD  00001D4E     3   UPDATE                     0.000544 .
MYTRADD  00001B96    19 EXEC SQL                     0.000707 .
MYTRADD  00001B98    19   SELECT                     0.008737 ======>
MYTRADD  00001DCC    11 EXEC SQL                     0.000488 .
Command ===> _____________________________________________ Scroll ==> CSR
                                                                  24/015
```

Figure 36-11 AM: TR -AOR CICS Calls in a Transaction

The display shows that at least two programs have been involved to run this transaction, one of them, MYTRADD had DB2 calls. From here we drill down to the CSECT Utilization Analysis by placing the cursor on the program name (MYTRADD) and pressing Enter.

Figure 36-12 shows the CSECT Utilization Analysis.

```
FBIP1F00 IBM AM for z/OS - CSECT Utilization Analysis ------------------------
         2004/11/03 10:43:48 to 2004/11/03 10:45:34       Row 1 of 1
History Data Set: CHABERT.AM.CICSC22F.D2004308.T1043474
    Jobname: CICSC31F        Stepname: CICSC31F        Duration: 00:01:19
    Samples: 253               Module: MYTRADD
     Csects: 18
   From Dsn: CHABERT.FBI.TEST.LOAD
       Desc:

CSECT     Language  Start     End       < 10 20 30 40 50 60 70 80 90 >  Pct.
--------  --------  --------  --------  ------------------------------  ------
MYTRADD   COBOL*    0DBC7CD0  0DBDE979  ============================>  100.00
****************************** BOTTOM OF DATA ******************************









Command ===> _____________________________________________ Scroll ==> CSR
                                                                   24/015
```

Figure 36-12 AM-TR AOR CSECT Utilization Analysis

Because an asterisk is appended to the name of the language, it means that the compiler listing is available to Application Monitor for reference. So we now drill down to the Statement Analysis display by placing the cursor on the language column and pressing Enter.

Statement Analysis report

In this case only CICS and SQL calls are displayed, as shown in Figure 36-13. Remember, Application Monitor does not trace all statements, but points out which ones have been the most expensive, and usually CICS and SQL statements are much more expensive than COBOL statements.

```
FBIP1F00 IBM AM for z/OS - Statement Analysis -------------------------------
         2004/11/03 10:43:48 to 2004/11/03 10:45:34       Row 1 of 5
History Data Set: CHABERT.AM.CICSC22F.D2004308.T1043474
    Jobname: CICSC31F        Stepname: CICSC31F        Duration: 00:01:19
    Samples: 253               Module: MYTRADD          CPU Pct: 1.24



Stmtno Statement                                        <   CPU%   >  Pct.
------ ------------------------------------------------ ------------ ------
   691 Call 'DFHEI1' using by content x'0e08000007000010 =>            24.44
   866 CALL 'DSNHLI' USING SQL-PLIST3.                   ====>         42.22
   908 CALL 'DSNHLI' USING SQL-PLIST4.                   .              2.22
   928 CALL 'DSNHLI' USING SQL-PLIST5.                   .              6.66
   951 CALL 'DSNHLI' USING SQL-PLIST6.                   =>            24.44
****************************** BOTTOM OF DATA ******************************





Command ===> _____________________________________________ Scroll ==> CSR
                                                                   24/015
```

Figure 36-13 AM: TR-AOR Statement Analysis

This display (as well as CICS Calls in a Transaction) shows that the time that the application spent in DB2 (under the cover of CICS) to process the SELECT statements is the most expensive. To check how these SELECT statements have been executed, we use the DB2 reports.

SQL Analysis report

We cursor-select the **Y** value in the DB2 Rpts column for CICSC31F and press Enter to drill down to the SQL Analysis report as shown in Figure 36-14.

```
FBIP1C00 IBM AM for z/OS - SQL Analysis ------------------------------------
         2004/11/03 10:43:48 to 2004/11/03 10:45:34       Row 1 of 5
History Data Set: CHABERT.AM.CICSC22F.D2004308.T1043474
    Jobname: CICSC31F        Stepname: CICSC31F          System: STLABF6
    LU Name: APLDB1E         DB2 SSID: DB1E              Module: MYTRADD
  Plan Name: MYTRADD         Auth Id.: CICSUSER        Oper Id.: CICSUSER
DB2  Auth-ID  Plan     DBRM     St No O SQLRC Type     Count Rel. DB2 CPU Time
---- -------- -------- -------- ----- - ----- -------- ----- <---------------->
DB1E CICSUSER MYTRADD  MYTRADD    477 S     0 SELECT      14 =====>
DB1E CICSUSER MYTRADD  MYTRADD    533 S     0 UPDATE       3 >
DB1E CICSUSER MYTRADD  MYTRADD    517 S     0 INSERT       1 >
DB1E CICSUSER MYTRADD  MYTRADD    550 S     0 SELECT      11 ====>
DB1E CICSUSER MYTRADD  MYTRADD    477 S   100 SELECT       5 =>
****************************** BOTTOM OF DATA ******************************



Command ===> _____________________________________________ Scroll ==> CSR
                                                                     24/0
```

Figure 36-14 AM: TR-AOR SQL Analysis

We need to have more detail, so we cursor-select statement number **550** and press Enter.

SQL Statement Summary report

For us, Figure 36-15 is an intermediate step to having the full text and dynamic explained.

```
FBIP1D00 IBM AM for z/OS - SQL Statement Summary ----------------------------
         2004/11/03 10:43:48 to 2004/11/03 10:45:34       Row 1 of 1
History Data Set: CHABERT.AM.CICSC22F.D2004308.T1043474
    Jobname: CICSC31F        Stepname: CICSC31F          System: STLABF6
    LU Name: APLDB1E         DB2 SSID: DB1E              Module: MYTRADD
  Plan Name: MYTRADD         Auth Id.: CICSUSER        Oper Id.: CICSUSER
  Statement: 550                 Type: SELECT
X  <-------------------- S T A T E M E N T   T E X T------------------------->
.  SELECT SHR_VAL , SHR_VAL1 , SHR_VAL2 , SHR_VAL3 , SHR_VAL4 , SHR_VAL5 , SH
****************************** BOTTOM OF DATA ******************************

Command ===> _____________________________________________________ Scroll ==> CSR
                                                                      24/015
```

Figure 36-15 AM: TR-AOR SQL Statement Summary

SQL Statement Full Text

If we need to have the full text statement, in order to know which host variables have been used, we can drill down to the full text by placing our cursor anywhere on the statement text and pressing Enter. A panel like Figure 36-16 is displayed.

```
FBIP1D00 IBM AM for z/OS - SQL Statement Full Text ---------------------------
         2004/11/03 10:43:48 to 2004/11/03 10:45:34       Row 1 of 14
History Data Set: CHABERT.AM.CICSC22F.D2004308.T1043474
    Jobname: CICSC31F        Stepname: CICSC31F          System: STLABF6
    LU Name: APLDB1E         DB2 SSID: DB1E              Module: MYTRADD
  Plan Name: MYTRADD         Auth Id.: CICSUSER        Oper Id.: CICSUSER
  Statement: 550                 Type: SELECT
X  <-------------------- S T A T E M E N T   T E X T------------------------->
.  SELECT SHR_VAL , SHR_VAL1 , SHR_VAL2 , SHR_VAL3 , SHR_VAL4 , SHR_VAL5 ,
.  SHR_VAL6 , SHR_VAL7 , COMM_BUY , COMM_SELL INTO : H , : H , : H , : H , :
.  H , : H , : H , : H , : H , : H FROM COMPANY_DETAILS WHERE COMPANY = : H
     V0001: DCLCOMPANY-DETAILS.SHR-VAL
     V0002: DCLCOMPANY-DETAILS.SHR-VAL1
     V0003: DCLCOMPANY-DETAILS.SHR-VAL2
     V0004: DCLCOMPANY-DETAILS.SHR-VAL3
     V0005: DCLCOMPANY-DETAILS.SHR-VAL4
     V0006: DCLCOMPANY-DETAILS.SHR-VAL5
     V0007: DCLCOMPANY-DETAILS.SHR-VAL6
     V0008: DCLCOMPANY-DETAILS.SHR-VAL7
     V0009: DCLCOMPANY-DETAILS.COMM-BUY
     V0010: DCLCOMPANY-DETAILS.COMM-SELL
     V0011: DCLCOMPANY-DETAILS.COMPANY
****************************** BOTTOM OF DATA ******************************
Command ===> _____________________________________________________ Scroll ==> CSR
                                                                      24/015
```

Figure 36-16 AM: TR-AOR SQL Statement Full Text

Now we need to know how DB2 processed this statement, to see if there is a way to improve it. We place the cursor on the **X** selection field and press Enter to bring up the SQL Detailed Explain Analysis display shown in Figure 36-17.

SQL Detailed Explain Analysis

Figure 36-17 provides information from the plan table that is created to hold Explain output.

```
FBIP1D00 IBM AM for z/OS - SQL Detailed Explain Analysis ---------------------
         2004/11/03 10:43:48 to 2004/11/03 10:45:34       Row 1 of 2
History Data Set: CHABERT.AM.CICSC22F.D2004308.T1043474
    Jobname: CICSC31F        Stepname: CICSC31F          System: STLABF6
    LU Name: APLDB1E         DB2 SSID: DB1E              Module: MYTRADD
  Plan Name: MYTRADD         Auth Id.: CICSUSER        Oper Id.: CICSUSER
  Statement: 550                 Type: SELECT
Table                Plan Method Match Acc  Join Index UQ JN OB GB   UQ JN OB GB
Name                  Nbr          Col Type Grp. Only
CUSTOMER_DETAILS        2      0     0         0
COMPANY_DETAILS         1      0     0         0
****************************** BOTTOM OF DATA ******************************



Command ===> _____________________________________________ Scroll ==> CSR
                                                                     24/015
```

Figure 36-17 AM: TR-AOR SQL Detailed Explain Analysis

We can use this display to analyze how DB2 processed our statement.

If we need more details, we can cursor-select either a value in the Table Name column to drill down to the DB Table Information display, or a value in the Plan Nbr column to drill down to the SQL Explain Detail display.

SQL Explain Detail

From the previous display we cursor-select the Plan Nbr 1 to get Figure 36-18.

```
FBIPXPLA IBM AM for z/OS - SQL Explain Detail ---------------------------------
         2004/11/03 10:43:48 to 2004/11/03 10:45:34
History Data Set: CHABERT.AM.CICSC22F.D2004308.T1043474
Command ===> _____________________________________________________________
                                                                  More:     +
    Jobname: CICSC31F        Stepname: CICSC31F          System: STLABF6
    LU Name: APLDB1E         DB2 SSID: DB1E              Module: MYTRADD
  Plan Name: MYTRADD         Auth Id.: CICSUSER        Oper Id.: CICSUSER
  Statement: 550                 Type: SELECT
                          Table Name: COMPANY_DETAILS
MixOP Seq . .: 0          Table Number.: 0          Plan Number .: 1
Query Number.: 0          Method. . . .: 0          Match Columns: 0
Acces Creator:            Access Degree: 0          Prefetch. . .:
Access Group.: 0          Column Eval .:            Join Degree .: 0
Access Type .:            Join Group. .: 0          Index Only. .:
Sort New. . . . .:                     Lock Mode.  . . .:
Sort Composite. .:                     Access Name . . .:
Package Version .:                     Collection ID.  .:
Time Stamp. . . .: -  -     .  .  .
                                                                     04/015
```

Figure 36-18 AM: TR-AOR SQL Explain Detail

This display is useful to understand and to improve the DB2 access path by creating an index and updating the SQL statements.

DB Table Information report

From the Detailed Explain Analysis display we cursor-select COMPANY_DETAILS, and Application Monitor shows the panel in Figure 36-19.

```
FBIPXPLA IBM AM for z/OS - DB2 Table Information ------------------------------
         2004/11/03 10:43:48 to 2004/11/03 10:45:34
History Data Set: CHABERT.AM.CICSC22F.D2004308.T1043474
Command ===> _____________________________________________________________
                                                                  More:     +
    Jobname: CICSC31F        Stepname: CICSC31F          System: STLABF6
    LU Name: APLDB1E         DB2 SSID: DB1E              Module: MYTRADD
  Plan Name: MYTRADD         Auth Id.: CICSUSER        Oper Id.: CICSUSER
  Statement: 550                 Type: SELECT
                          Table Name: COMPANY_DETAILS
Plan Number .: 1          Mix OP Seq. .: 0          Table Number.: 0
Table Type. .: T          Table Stat. .: X          Edit Proc . .:
Valid Proc. .:            Database ID .: 4          Column count.: 11
Table OBID. .: 22         Percent Pages: 8          Uniq Index ID: 23
Key Columns .: 1          Parents . . .: 0          Children. . .: 0
Record Length . .: 66                   % Compression . .: 0
Creator . . . . .: CHABERT              Creation Tbl Spac: TRADER
Creation Database: DSNDB04              Base Table . . . .
Creation RBA. . .: 0000027A0491         Cardinality . . .: 6
Alter RBA . . . .: 0000027A0491         Pages with rows .: 1
Check Flag. . . .:                      Check RID . . . .:
Data Capture. . .:                      Auditing. . . . .:
Created by  . . .: CHABERT              Create Date . . .: 2004-10-20-15.25.01
Alter Date stamp.: 2004-10-20-15.25.01 Runstats Date . .: 2004-10-20-16.03.07
                                                                     04/015
```

Figure 36-19 AM: TR-AOR DB2 Table Information

Because our application accesses a VSAM data set, we can use Dsn Rpts to see how this file has been used by cursor-selecting the **Y** value in this column and pressing Enter. Application Monitor brings up the panel in Figure 36-20.

Data Set Analysis Report

This report provides information about the data sets that CICSC31F accesses.

```
FBIP1C00 IBM AM for z/OS - Data Set Analysis Report -------------------------
         2004/11/03 10:43:48 to 2004/11/03 10:45:34       Row 37 of 51        >
History Data Set: CHABERT.AM.CICSC22F.D2004308.T1043474
     System: STLABF6          Jobname: CICSC31F        Stepname: CICSC31F
   Duration: 00:01:46

DD        Acc  Rec    Rec CI/Blk       EXCP Data Set
Name     Meth Form Length   Size      Count Name
-------- ---- ---- ------ ------ ---------- ----------------------------------
CEEMSG   BSAM V       161    165          0 CICPROD.CICSC31F.STC00405.D0000113.
CEEOUT   BSAM V       133    137          0 CICPROD.CICSC31F.STC00405.D0000114.
CRPO     BSAM V       133    137          0 CICPROD.CICSC31F.STC00405.D0000115.
INTR     BSAM F        80                  0 CICPROD.CICSC31F.STC00405.D0000116.
CCTLDD   BPAM U           32,760         22 IMSVS.IM8F.SDFSRESL
COMPFILE VSAM              3,072          0 CHABERT.C31F.TRADER.COMPFILE
DFHDPFMP VSAM              3,072          0 CICSVS.TS310.CICS.DFHDPFMP
DFHDBFK  VSAM              2,048          0 CICSVS.C31F.DFHDBFK
DFHLRQ   VSAM              2,048          0 CICSVS.C31F.DFHLRQ
DFHCSD   VSAM             18,432          0 CICSVS.C31F.DFHCSD
DFHGCD   VSAM              8,192          0 CICSVS.C31F.DFHGCD
DFHLCD   VSAM              2,048          0 CICSVS.C31F.DFHLCD
DFHINTRA VSAM              4,096          0 CICSVS.C31F.DFHINTRA
DFHTEMP  VSAM              4,096          0 CICSVS.C31F.DFHTEMP
Command ===> ___________________________________________ Scroll ==> CSR
                                                                  24/015
```

Figure 36-20 AM: TR-AOR Data Set Analysis Report

Our user file is COMPFILE. We cursor-select the Acc Meth (because it is a VSAM file) and press Enter to obtain more useful information. Application Monitor returns the panel shown in Figure 36-21.

VSAM Data Set Analysis Detail Report

This report contains detailed information about the VSAM file.

```
FBIPI080 IBM AM for z/OS - VSAM Data Set Analysis Detail Report --------------
         2004/11/03 10:43:48 to 2004/11/03 10:45:34
History Data Set: CHABERT.AM.CICSC22F.D2004308.T1043474
Command ===> _____________________________________________________________________

     System: STLABF6          Jobname: CICSC31F        Stepname: CICSC31F
   Duration: 00:01:46
    DD Name: COMPFILE         DS Name: CHABERT.C31F.TRADER.COMPFILE
                            Data Source: JOB CTL BLOCKS
Status   Type Format  Access   Opts            LSR  Misc
CLSD-OUT KSDS         LSR      DIR   ADRKEY    001  31B
                                                     DSNSHR UPAD
DATA:-------------------------------------------------------------------------
 CIFR  CAFR   CISP   CASP   CISZ   CI-CA   Keyln  Keyps  Maxlr
10     10     0      0      3,072  225     17     1      2,560

EXTNT EXCPs      Reccount     Retrved     Updated     Deleted     Insrted
1     0          2            50          0           0           0

INDEX:------------------------------------------------------------------------
 CIFR   CAFR  CISP       CASP  CISZ    CA     LVL    EXT Reccnt    EXCP
0       0     0          0     2,560   17     1      1   1         0


                                                                        04/015
```

Figure 36-21 AM: TR-AOR VSAM Data Set Analysis Detail Report

This display shows that CICS uses the local shared resource pool number 1 to manage COMPFILE's records. We cursor-select that value, **001** to drill down to the VSAM Local Shared Resource Pools report as shown in Figure 36-22.

```
FBIP1C00 IBM AM for z/OS - VSAM Local Shared Resource Pools ------------------
         2004/11/03 10:43:48 to 2004/11/03 10:45:34       Row 1 of 22         >
History Data Set: CHABERT.AM.CICSC22F.D2004308.T1043474
     System: STLABF6          Jobname: CICSC31F        Stepname: CICSC31F
   Duration: 00:01:46

   User Key Str Type    Buf    Buf   Hbuf ------ Gets - ------ Puts - -Hiper Re
ID Cnt  Len Nbr         Len    Nbr   Nbr  --IO-- ------ -User- ------ --OK-- --
-- ---- --- --- ---- ------ ------ ------ ------ ------ ------ ------ ------ --
 1    2 255  50 D       512     10      0      0      0      0      0      0
 1    2 255  50 D     1,024     10      0      0      0      0      0      0
 1    2 255  50 D     2,048     10      0      0      0      0      0      0
 1    2 255  50 D     4,096     10      0      3    192      0      0      0
 1    2 255  50 D     8,192     10      0      0      0      0      0      0
 1    2 255  50 D    12,288     10      0      0      0      0      0      0
 1    2 255  50 D    16,384     10      0      0      0      0      0      0
 1    2 255  50 D    20,480     10      0    283    682     57      0      0
 1    2 255  50 D    24,576     10      0      0      0      0      0      0
 1    2 255  50 D    28,672     10      0      0      0      0      0      0
 1    2 255  50 D    32,768     10      0      0      0      0      0      0
 1    2 255  50 I       512      5      0      5    192      0      0      0
 1    2 255  50 D     1,024      5      0     33  1,227      0      0      0
 1    2 255  50 D     2,048      5      0      0      0      0      0      0
Command ===> _____________________________________________ Scroll ==> CSR
                                                                        24/015
```

Figure 36-22 AM: TR-AOR VSAM Local Shared Resources Pools

This panel shows the LSRPool definition done in CICS, and how buffers are used. It can help us to improve the buffer definitions. For instance: Do we have to defined some hiperspace buffers?

36.2 Application Monitor and Function Shipping

CICS function shipping enables CICS application programs to:

- Access CICS files owned by other CICS systems by shipping file control requests.
- Transfer data to or from transient-data and temporary-storage queues in other CICS systems by shipping requests for transient-data and temporary-storage functions.
- Initiate transactions in other CICS systems, by shipping interval control START requests.

Applications can be written without regard for the location of the requested resources; they simply use file control commands, temporary-storage commands, and other functions in the same way.

36.2.1 System configuration

For our purpose we will use the Function Shipping feature to access file which are not owned by the CICS where we start the transaction and where we run the programs (T.O.R: Terminal Owning Region and A.O.R: Application Owning Region at the same time), but this file is owned by another CICS (F.O.R: File Owning Region) as illustrated in Figure 36-23.

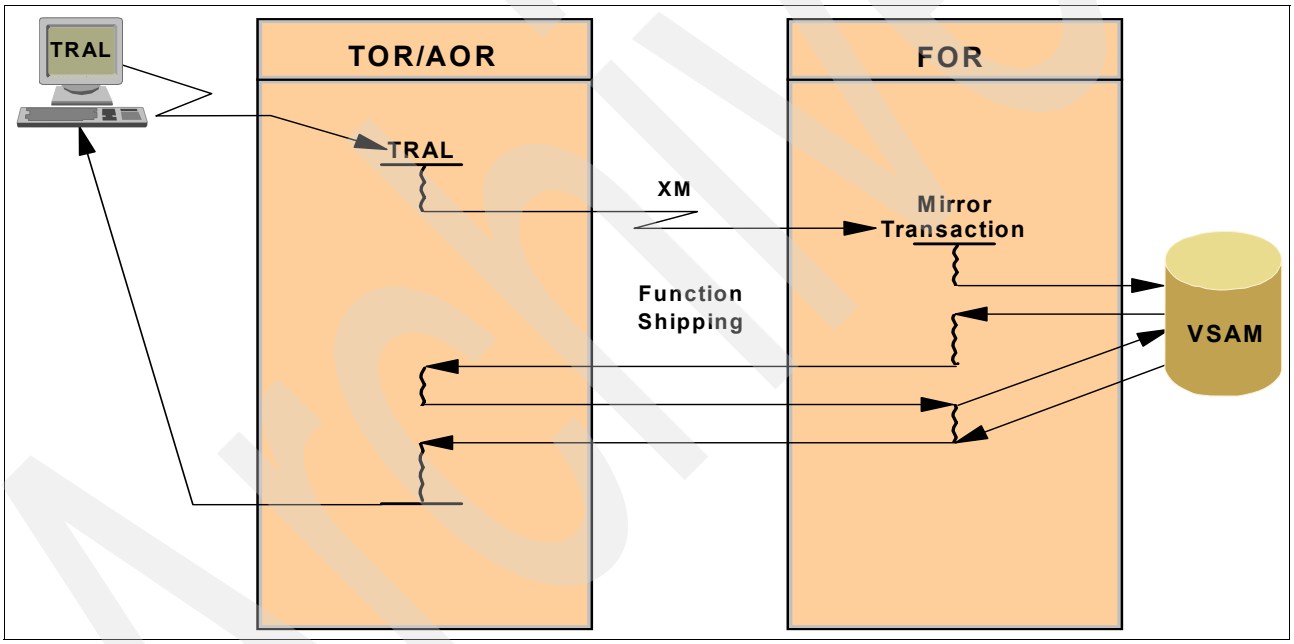


Figure 36-23 CICSplex: Function Shipping configuration

36.2.2 Application Monitor set-up

In order to monitor our CICSplex configuration, we use the CICSPLEX Named Set. The following sections briefly describe how to create a Named Set, using Application Monitor option **7.2** Create, Edit and Delete Named Sets, and how to Launch a Named Set, using Application Monitor's option **3** Analyze a Named Set of Jobs.

36.2.3 Application Monitor reports

If CICSC22F owns Transactions (TOR) and Programs (AOR) but it uses Function Shipping to access files owned by CICSC31F (FOR), all Application Monitor reports are in TOR/AOR, but the Dsn Rpts are in FOR.

Note: If a CICS File Control call is shipped from AOR to FOR where it is executed under the Mirror Transaction, and CPU cycles are actually used in FOR, Application Monitor collects the samples and reports CPU and CICS calls related data under AOR, excluding Dsn Rpts, which will be reported under FOR.

Because we are using the same transaction as for the Transaction Routing scenario (only some CICS resource definitions have been changed to move from Transaction Routing to Function Shipping), we have the same reports. Our intention it not to re-display all of them, but just to point out that all of them are available under CICSC22F, except Dsn Rpts which are under CICSC31F.

36.3 Application Monitor and Dynamic Program Link

CICS distributed program link enables CICS application programs to run programs residing in other CICS regions by shipping program-control LINK requests.

An application can be written without regard for the location of the requested programs; it simply uses program-control LINK commands in the usual way.

36.3.1 System configuration

An illustration of a DPL request is shown in Figure 36-24. In this figure, a program (known as a client program) running in CICSC22F issues a program-control LINK command for a program called MYTRADS (the server program). From the installed program definitions, CICS discovers that this program is owned by a remote CICS system called CICSC31F. CICS changes the LINK request into a suitable transmission format, and then ships it to CICSC31F for execution.

In CICSC31F, the mirror transaction is attached. The mirror program recreates the original request, issues it on CICSC31F, and, when the server program has run to completion, returns any communication-area data to CICSC22F.

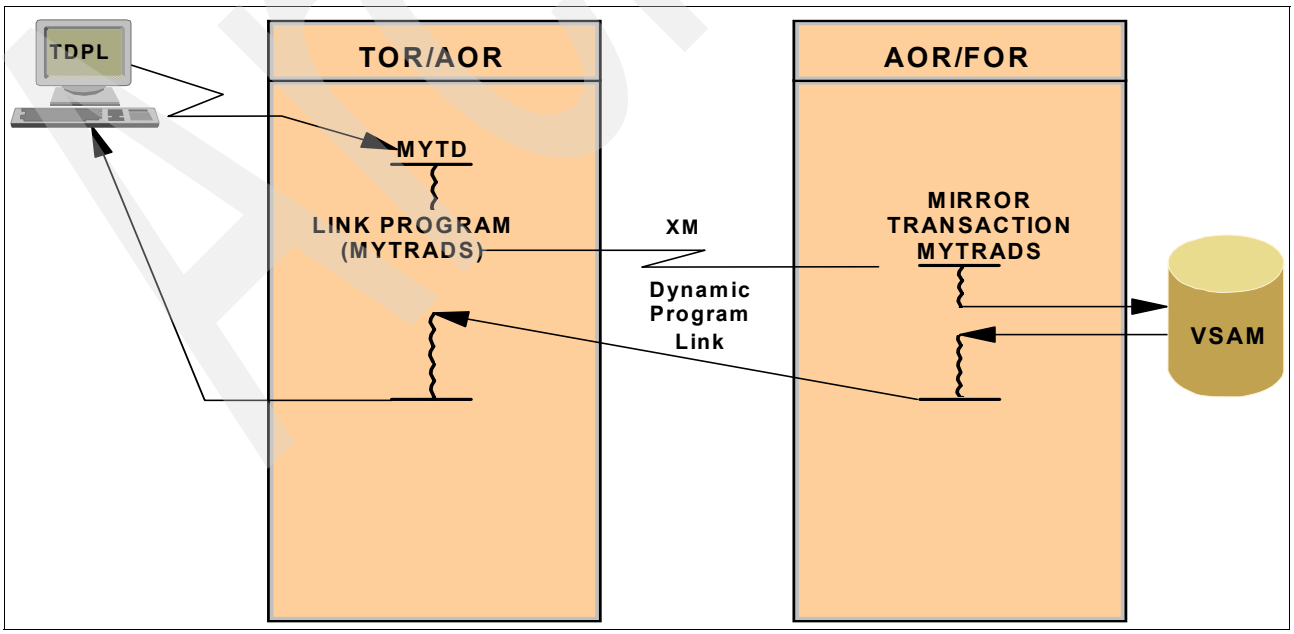


Figure 36-24 CICSplex - Dynamic Program Link configuration

36.3.2 Application Monitor set-up

In order to monitor our CICSplex configuration, we use the CICSPLEX Named Set. The following sections briefly describe how to create a Named Set using Application Monitor option **7.2** Create, Edit and Delete Named Sets, and how to Launch a Named Set, using Application Monitor's option **3** Analyze a Named Set of Jobs.

36.3.3 Application Monitor reports

We use a history data set to display the reports.

Application Monitor shows first the Historical Reports Available display as shown in Figure 36-25. CICSC22F is our TOR/AOR and CICSC31F is our AOR/FOR.

```
FBIP1H10 IBM AM for z/OS - Historical Reports Available ----------------------
         2004/11/03 18:40:31 to 2004/11/03 18:44:39       Row 1 of 2
History Data Set: CHABERT.AM.CICSC22F.D2004308.T1840305
  Interval Start: 2004/11/03  18:40:31
   Interval Stop: 2004/11/03  18:43:08

System   Jobname  Stepname Pgm  CICS IMS  DB2  Dly  Dsn  CPU%   Sample
-------- -------- -------- Rpts Rpts Rpts Rpts Rpts Rpts ------  Count
STLABF6  CICSC31F CICSC31F    7 YES       YES  YES  YES    0.09    391
STLABF6  CICSC22F CICSC22F   10 YES       YES  YES  YES    0.36    186
****************************** BOTTOM OF DATA ******************************



Command ===> _____________________________________________ Scroll ==> CSR
                                                                   24/015
```

Figure 36-25 AM: DPL Historical Reports Available

Both CICS have Pgm Rpts, CICS Rpts, Dly Rpts and Dsn Rpts reports available. Because Dly Rpts and Dsn Rpts have nothing special, we do not go through them.

CICSC22F (TOR): CICS Transactions

We log on to CICSC22F when we start the MYTD transaction, so we start our analysis from here, as shown in Figure 36-26.

```
-------- IBM AM for z/OS - CICS Transactions ---------------------------------
         2004/11/03 18:40:31 to 2004/11/04 08:28:10       Row 1 of 5
History Data Set: CHABERT.AM.CICSC22F.D2004308.T1840305
   CICS Job: CICSC22F     CICS APPLID: CICSC22F          System: STLABF6

Tran Program   Count  Busy%
---- -------- ------ ------
CEX2 DFHD2EX2      1  99.94 ==============>
MYTD MYTRADMV     35   0.05 .
CATD DFHZATD       3   0.00
CSNE DFHZNEP       4   0.00
CWBG DFHWBGB       1   0.00
****************************** BOTTOM OF DATA ******************************



Command ===> _____________________________________________ Scroll ==> CSR
                                                                   24/015
```

Figure 36-26 AM: DPL-TOR CICS Transactions

We cursor-select the **MYTD** transaction in order to drill down to the CICS Calls in Transaction, and we sort the display in descending order by Total Time. Application Monitor returns the panel shown in Figure 36-27.

CICSC22F: CICS Calls in a Transaction

Figure 36-27 shows that **LINK** call are the most expensive.

```
-------- IBM AM for z/OS - CICS Calls in a Transaction ------------------------
         2004/11/03 18:40:31 to 2004/11/04 08:39:29       Row 1 of 70
History Data Set: CHABERT.AM.CICSC22F.D2004308.T1840305
   CICS Job: CICSC22F     CICS APPLID: CICSC22F          System: STLABF6
Transaction: MYTD          CICS calls: 35

Program  Offset   Count Activity                    Total time
-------- -------- ----- ---------------------- ---------------
MYTRADMV 000023FC     8   LINK                       0.026840 ======>
MYTRADMV 000028B0     3   LINK                       0.015861 ===>
MYTRADMV 00002CB8     2   LINK                       0.014996 ==>
MYTRADMV 00001D2A    16   RECEIVE MAP                0.001320 .
MYTRADMV 000013AC     8   RECEIVE MAP                0.000655 .
MYTRADMV 0000262C     8   SEND MAP                  0.000577 .
MYTRADMV 0000213A     1   STARTBR                    0.000523 .
MYTRADMV 00001338    16 EXEC CICS                    0.000497 .
MYTRADMV 00001440     8   SEND MAP                  0.000350 .
MYTRADMV 0000262A     8 EXEC CICS                    0.000337 .
MYTRADMV 0000182C     4   RECEIVE MAP                0.000320 .
MYTRADMV 000023FA     8 EXEC CICS                    0.000294 .
MYTRADMV 0000143E     8 EXEC CICS                    0.000250 .
MYTRADMV 00001640     3   RECEIVE MAP                0.000244 .
MYTRADMV 000015C6     8 EXEC CICS                    0.000238 .
Command ===> _____________________________________________ Scroll ==> CSR
                                                                   09/002
```

Figure 36-27 AM: DPL-TOR CICS Calls in a Transaction

Because the Program column contains only one name (MYTRADMV) it means that the links are for remote programs.

We press PF3 several times to cancel and return to the Historical Reports Available. We cursor-select the **CICS Rpts** for CICSC31F (AOR); in response, Application Monitor shows Figure 36-28.

```
-------- IBM AM for z/OS - CICS Transactions ---------------------------------
         2004/11/03 18:40:31 to 2004/11/04 09:31:13       Row 1 of 2
History Data Set: CHABERT.AM.CICSC22F.D2004308.T1840305
   CICS Job: CICSC31F     CICS APPLID: CICSC31F          System: STLABF6

Tran Program   Count  Busy%
---- -------- ------ ------
CEX2 DFHD2EX2      1  99.95 ==============>
MYTD DFHMIRS      25   0.04 .
****************************** BOTTOM OF DATA ********************************

Command ===> _____________________________________________ Scroll ==> CSR
                                                                     24/015
```

Figure 36-28 AM: DPL-AOR CICS Transactions

We can see that our user transaction MYTD first ran the CICS DFHMIRS program. DFHMIRS is the CICS mirror program, which is used for Dynamic Program Link as well as for any Function Shipping request. To view our user programs we have to cursor-select **MYTD** and press Enter to drill down to the CICS Calls in a Transaction as shown in Figure 36-29.

```
-------- IBM AM for z/OS - CICS Calls in a Transaction ------------------------
         2004/11/03 18:40:31 to 2004/11/04 09:36:29       Row 6 of 20
History Data Set: CHABERT.AM.CICSC22F.D2004308.T1840305
   CICS Job: CICSC31F     CICS APPLID: CICSC31F          System: STLABF6
Transaction: MYTD          CICS calls: 10

Program  Offset   Count Activity                    Total time
-------- -------- ----- ---------------------- ---------------
MYTRADS  00001A2E     3   REWRITE                    0.012027 ======>
MYTRADS  00001DDE   101 EXEC CICS                    0.001575 .
MYTRADS  00001DE0   101   ASKTIME ABSTIME            0.003943 =>
MYTRADS  00001EE0   102 EXEC CICS                    0.001673 .
MYTRADS  00001EE2   102   FORMATTIME                 0.011007 =====>
MYTRADS  00002188   102 EXEC CICS                    0.008544 ===>
MYTRADS  0000218A   102   WRITEQ TS                  0.003354 >
MYTRADS  000016BE     8 EXEC CICS                    0.000131 .
MYTRADS  000016C0     8   READ                       0.000633 .
MYTRADS  00001B16    13 EXEC CICS                    0.000217 .
MYTRADS  00001B18    13   READ                       0.000650 .
MYTRADS  00000EEE    12 EXEC CICS                    0.000210 .
MYTRADS  00000EF0    12   RETURN                     0.000000
DFHMIRS  000011AA    25 EXEC CICS                    0.000000
DFHMIRS  000011AC    25   LINK                       0.000000
Command ===> _____________________________________________ Scroll ==> CSR
                                                                     24/016
```

Figure 36-29 AM: DPL-AOR CICS Calls in a Transaction

It confirms that DFHMIRS issues a link to MYTRADS, which runs locally. We can now use any other Application Reports to analyze how MYTRADS runs, how the files are accessed, and if it is possible to improve our application.

36.4 Conclusions

This chapter has demonstrated the following concepts:

Transaction Routing

If CICS (TOR) uses Transaction Routing to run a transaction into another CICS (AOR), all Application Monitor's reports are under the AOR.

Function Shipping

If CICS (TOR/AOR) uses Function Shipping to access files owned by another CICS (FOR), all Application Monitor's reports are under the TOR/AOR.

Distributed Program Link

If CICS (AOR1) links to a remote program running into another CICS (AOR2), then for EXEC CICS LINK, Application Monitor reports the overall elapsed time taken by the dynamic program link in AOR1.

If we also analyze the AOR2 (CICSC31F in our case) where the target program runs, Application Monitor breaks the time down into the individual elapsed times taken by the CICS calls in the transaction.

The difference between the elapsed times in the two AORs represents transmission and scheduling delays for the dynamic program link.

37

Application Monitor and Parallel Sysplex

The scenarios described in this chapter are running on a Parallel Sysplex environment. These scenarios were designed to highlight features of Application Monitor in a brief, but effective manner. We cover:

- Batch job in a sysplex environment
  - System configuration
  - Application Monitor set-up
  - Application Monitor reports
  - Conclusion
- CICS and DB2 data sharing group
  - System configuration
  - Application Monitor set-up
  - Application Monitor reports
  - Conclusion

37.1 Batch job in a Parallel Sysplex environment

In this example we run a batch job in a Parallel Sysplex environment. It consists of two LPARs named STLABF6 and STLABF7. Each LPAR runs a z/OS operating system, using a coupling facility to exchange data and messages.

37.1.1 System configuration

The system configuration is as illustrated in Figure 37-1.

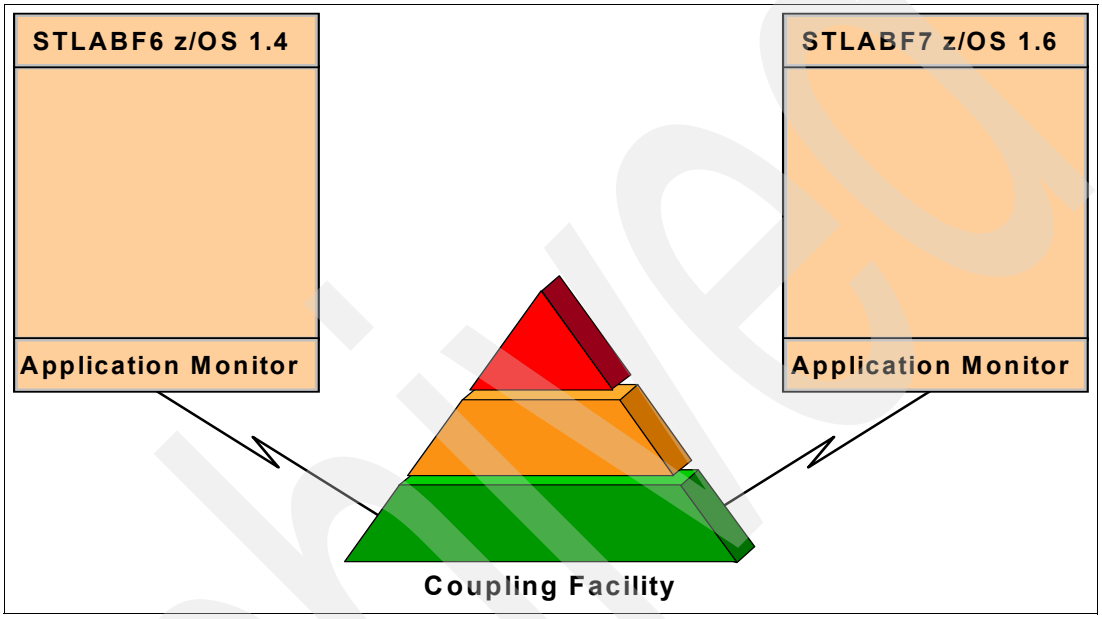


Figure 37-1 AM: Sysplex configuration

In such an environment, even though we know which batch job we have to run, we never know where it will run because it could be on either of these two systems, based on our WLM policy. Application Monitor must be able to monitor any job running on any system.

37.1.2 Application Monitor set-up

We have to start an Application Monitor instance in each system (STLABF6 and STLABF7). Because of the coupling facility these two Application Monitor instances are able to communicate through an XCF group named FBIG0020.

XCF is part of the z/OS base control program, providing high-performance communication links between MVS images that are linked in a sysplex (systems complex) by channel-to-channel links, ESCON® channels, or coupling facility links as shown in Figure 37-2.

```
D XCF,GROUP,DFHIR000,ALL
IXC333I  09.08.26  DISPLAY XCF 385
   INFORMATION FOR GROUP DFHIR000
   MEMBER NAME:        SYSTEM:     JOB ID:    STATUS:
   CICSC22F            STLABF6     CICSC22F   ACTIVE
   CICSC23G            STLABF7     CICSC23G   ACTIVE
   CICSC31F            STLABF6     CICSC31F   ACTIVE
   CICSC31G            STLABF7     CICSC31G   ACTIVE
```

Figure 37-2 AM: Sysplex and FBI XCF group

We know the name of the batch job that we have to run, but we cannot predict where it will run. We set up two deferred requests for that job, one for each operating system. The Administer Deferred Requests display in Figure 37-3 shows these two requests.

```
-------- IBM AM for z/OS - Administer Deferred Requests ------ Row 1 to 2 of 2


Enter a line command to administer a monitoring request

  D - Delete (Delete the monitoring request)
  P - Stop (Stop monitoring)
  V - View (See details about the request)


C Status   System   Name       Request Enabled     Request Disabled
_ Pending  STLABF7  BATCH001   2004/11/04 14:16:00 2004/11/04 15:00:00
_ Pending  STLABF6  BATCH001   2004/11/04 14:16:00 2004/11/04 15:00:00
****************************** Bottom of data ******************************









Command ===> _____________________________________________ Scroll ==> CSR
                                                                   24/015
```

Figure 37-3 AM: Sysplex: Administer Deferred Requests

As soon as the batch job starts, the Status of one of these two request becomes Active, then Complete when BATCH001 ends. We can now use primary option **5,** Analyze Historical Data to perform our analysis.

37.1.3 Application Monitor reports

The Historical Reports Available display in Figure 37-4 shows a value of 5 for Pgm Rpts.

```
-------- IBM AM for z/OS - Historical Reports Available ----------------------
         2004/11/04 14:21:36 to 2004/11/04 14:27:53       Row 1 of 1
History Data Set: CHABERT.AM.BATCH001.D2004309.T1416426
  Interval Start: 2004/11/04  14:21:36
   Interval Stop: 2004/11/04  14:22:12

System   Jobname  Stepname Pgm  CICS IMS  DB2  Dly  Dsn  CPU%   Sample
-------- -------- -------- Rpts Rpts Rpts Rpts Rpts Rpts ------  Count
STLABF6  BATCH001 STEP01      5                           23.90  6,038
****************************** BOTTOM OF DATA ********************************

Command ===> _____________________________________________ Scroll ==> CSR
                                                                   24/015
```

Figure 37-4 AM: Sysplex - Historical Reports Available

We cursor-select this value of 5 to drill down to the CPU and Loadmod Analysis as shown in Figure 37-5.

```
-------- IBM AM for z/OS - CPU and Loadmod Analysis --------------------------
         2004/11/04 14:21:36 to 2004/11/04 14:22:12       Row 1 of 5
History Data Set: CHABERT.AM.BATCH001.D2004309.T1416426
    Jobname: BATCH001        Stepname: STEP01          Duration: 00:00:35
    Samples: 6,038              %Supv: 0.06               %Prob: 99.93
    CPU Pct: 23.90           %InReady: 106.24           %InWait: 0.00
       %Out: 0.00             %OutRdy: 0.00             %OutLog: 0.00
Module   Area Address  <  10   20   30   40   50   60   70   80   90   > Pct.
-------- ---- -------- ---------------------------------------------------------
IGZCPAC  EPVT 0C564080 ==========================>                          58.21
COBPLIAM EPVT 0BF00678 ===================>                                 41.71
CSAHIGH  ECSA 05D15000 .                                                    0.03
CEEPLPKA EPVT 0BF1B350 .                                                    0.01
IAXVP    NUC  016C4B50 .                                                    0.01
****************************** BOTTOM OF DATA ********************************

Command ===> _____________________________________________ Scroll ==> CSR
                                                                   24/015
```

Figure 37-5 AM: Sysplex - CPU and Loadmod Analysis

Now we cursor-select our user program named COBPLIAM to drill down to CSECT Utilization Analysis as shown in Figure 37-6.

```
-------- IBM AM for z/OS - CSECT Utilization Analysis ------------------------
         2004/11/04 14:21:36 to 2004/11/04 14:22:12       Row 1 of 5
History Data Set: CHABERT.AM.BATCH001.D2004309.T1416426
    Jobname: BATCH001        Stepname: STEP01          Duration: 00:00:35
    Samples: 6,038             Module: COBPLIAM
     Csects: 23
   From Dsn: CHABERT.FBI.TEST.LOAD
       Desc:

CSECT     Language  Start     End       < 10 20 30 40 50 60 70 80 90 >  Pct.
--------  --------  --------  --------  ------------------------------  ------
COBPLIAM  COBOL*    0BF00678  0BF038C9  .                                 0.75
EPLISP11  PL/1      0BF038E8  0BF041EF  ====================>            78.88
COBOLSP2  COBOL*    0BF047C0  0BF079F9  ====>                            18.81
IGZCBSO   BAL       0BF07A00  0BF07F67  .                                 0.07
COBOLSP1  COBOL*    0BF08328  0BF0AFF9  .                                 1.46
****************************** BOTTOM OF DATA *******************************




Command ===> _____________________________________________ Scroll ==> CSR
                                                                    24/015
```

Figure 37-6 AM: Sysplex - CSECT Utilization Analysis

It shows that COBPLIAM is a mutli-CSECT, multi-language program. Because the COBOL language is followed by an asterisk, the compiler listing for COBOL CSECT is available as we can see in Figure 37-7.

```
-------- IBM AM for z/OS - Statement Analysis ---------------------------------
         2004/11/04 14:21:36 to 2004/11/04 14:22:12       Row 1 of 10
History Data Set: CHABERT.AM.BATCH001.D2004309.T1416426
    Jobname: BATCH001        Stepname: STEP01          Duration: 00:00:35
    Samples: 6,038             Module: COBPLIAM         CPU Pct: 23.90



Stmtno Statement                                        <   CPU%   >  Pct.
------ ------------------------------------------------ ------------ ------
    41 COMPUTE LOAN-AMOUNT = FUNCTION NUMVAL(LOAN-AMOUNT .              2.73
    42 COMPUTE INTEREST = FUNCTION NUMVAL(INTEREST-IN).  .              0.99
    43 COMPUTE NO-OF-PERIODS = FUNCTION NUMVAL(NO-OF-PER .              0.74
    45 PERFORM GET-AMOUNTS VARYING COUNTER FROM 1 BY 1 U .              1.99
    48 COMPUTE PAYMENT = FUNCTION PRESENT-VALUE(INTEREST ========>     75.87
    51 MOVE PAYMENT TO PAYMENT-OUT.                      .              5.72
    61 INSPECT OUTPUT-LINE REPLACING ALL "_" BY SPACES.  .              0.24
    63 MOVE "OK" TO CALL-FEEDBACK.                       .              2.73
    68 MOVE BUFFER-ARRAY (COUNTER) TO INPUT-1.           .              1.99
    69 COMPUTE VALUE-AMOUNT (COUNTER) = FUNCTION NUMVAL( .              6.96
****************************** BOTTOM OF DATA *******************************

Command ===> _____________________________________________ Scroll ==> CSR
                                                                    24/015
```

Figure 37-7 AM: Sysplex - Statement Analysis

As shown in Figure 37-6, CSECT Utilization Analysis, the EPLISP11 PL/1 CSECT uses 78.88% of available CPU used by COBPLIAM. It will be useful to have its source code, to analyze whether we can improve it.

By cursor-selecting EPLISP11 and pressing Enter, we drill down to the Statement Analysis Member selection from where we can select which listing as to be used, as shown in Figure 37-8.

```
-------- IBM AM for z/OS - Statement Analysis Member selection ----------------
Member List  CHABERT.FBI.TEST.PLILIST                      Row 00001 of 00004
   Name     Prompt          Size    Created           Changed            ID
. EPLISP1
. FBIEPLI
. FBIEPLIE
. FBIZS01
  **End**

Command ===> _____________________________________________ Scroll ===> PAGE
                                                                    24/016
```

Figure 37-8 AM: Sysplex - Statement Analysis Member selection

By selecting EPLISP1 from the list, we drill down to the Statement Analysis display shown in Figure 37-9.

```
-------- IBM AM for z/OS - Statement Analysis ---------------------------------
         2004/11/04 14:21:36 to 2004/11/04 14:22:12       Row 1 of 45
History Data Set: CHABERT.AM.BATCH001.D2004309.T1416426
    Jobname: BATCH001        Stepname: STEP01          Duration: 00:00:35
    Samples: 6,038             Module: COBPLIAM         CPU Pct: 23.90



Stmtno Statement                                        <   CPU%   >  Pct.
------ ------------------------------------------------ ------------ ------
    36 2            CHAR2 = 'for applications and sys p .              1.45
    37 2            CHAR3 = 'A non-intrusive monitor    .              3.57
    38 2            CHAR4 = 'provides resource utilizat .              2.36
    39 2            CHAR5 = 'information for your appli .              3.12
    40 2            CHAR6 = 'APM provides info on your  .              2.01
    41 2            CHAR7 = 'The output can be the curr .              2.41
    42 2            CHAR8 = 'system data                .              1.81
    43 2            CHAR9 = '(Real time analysis)       .              2.46
    44 2            CHAR11= 'the collected data appears .              2.66
    45 2            CHAR12= 'real time or printed forma .              1.50
    46 2            CHAR13= 'or it might be the data    .              2.01
    47 2            CHAR14= 'collected over a period of .              1.66
Command ===> ___________________________________________ Scroll ==> CSR
                                                                    24/015
```

Figure 37-9 AM: Sysplex - Statement Analysis

If we press PF3 to return to the prior panel, an asterisk now follows the PL/1 language for the EPLISP11 CSECT as shown in Figure 37-10.

```
-------- IBM AM for z/OS - CSECT Utilization Analysis ------------------------
         2004/11/04 14:21:36 to 2004/11/04 14:22:12       Row 1 of 5
History Data Set: CHABERT.AM.BATCH001.D2004309.T1416426
    Jobname: BATCH001        Stepname: STEP01          Duration: 00:00:35
    Samples: 6,038             Module: COBPLIAM
     Csects: 23
   From Dsn: CHABERT.FBI.TEST.LOAD
       Desc:

CSECT     Language  Start     End       < 10 20 30 40 50 60 70 80 90 >  Pct.
--------  --------  --------  --------  ------------------------------  ------
COBPLIAM  COBOL*    0BF00678  0BF038C9  .                                0.75
EPLISP11  PL/1*     0BF038E8  0BF041EF  ====================>          78.88
COBOLSP2  COBOL*    0BF047C0  0BF079F9  ====>                         18.81
IGZCBSO   BAL       0BF07A00  0BF07F67  .                                0.07
COBOLSP1  COBOL*    0BF08328  0BF0AFF9  .                                1.46
****************************** BOTTOM OF DATA ******************************




Command ===> _____________________________________________ Scroll ==> CSR
                                                                   24/015
```

Figure 37-10 AM: Sysplex - CSECT Utilization Analysis

37.1.4 Conclusion

Application Monitor can monitor any job running in a sysplex without any specific customization.

Application Monitor can monitor any job having multi-csects, using multi-languages, and it is able to display each source code.

37.2 CICS and DB2 data sharing group

For this scenario, we use TRADER application, but the DB2 tablespaces have been defined in an DB2 data sharing environment.

37.2.1 System configuration

The DB2 data sharing group we are using has two members, one in each LPAR. The DB2 data sharing group name is D80G, the DB2 member name on STLABF6 is D82G, the DB2 member name on STLABF7 is D81G, as illustrated in Figure 37-11.

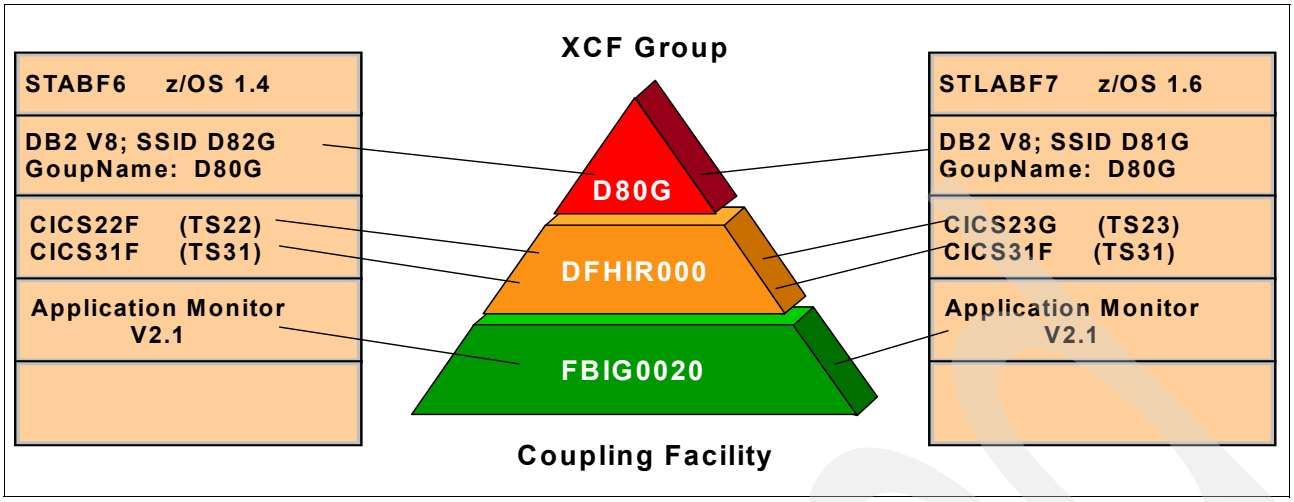


Figure 37-11 AM: Data Sharing configuration

Both CICS and DB2 subsystems use XCF group for their communications as shown in Figure 37-12 and Figure 37-13.

```
D XCF,GROUP,D80G,ALL
IXC333I  09.05.55  DISPLAY XCF 378
   INFORMATION FOR GROUP D80G
   MEMBER NAME:        SYSTEM:     JOB ID:     STATUS:
   DB2INMVSD81G        STLABF7     D81GMSTR    ACTIVE
   DB2INMVSD82G        STLABF6     D82GMSTR    ACTIVE
```

Figure 37-12 AM: Sysplex DB2 XCF group

```
D XCF,GROUP,DFHIR000,ALL
IXC333I  09.08.26  DISPLAY XCF 385
   INFORMATION FOR GROUP DFHIR000
   MEMBER NAME:        SYSTEM:     JOB ID:     STATUS:
   CICSC22F            STLABF6     CICSC22F    ACTIVE
   CICSC23G            STLABF7     CICSC23G    ACTIVE
   CICSC31F            STLABF6     CICSC31F    ACTIVE
   CICSC31G            STLABF7     CICSC31G    ACTIVE
```

Figure 37-13 AM: Sysplex CICS XCF group

The CICS on each LPAR has been customized to access the DB2 data sharing group. Figure 37-14 is a DB2CONN definition for one of them.

```
 I DB2CON _
 STATUS:  RESULTS - OVERTYPE TO MODIFY
   Accountrec( Uow )              Planexitname( DSNCUEXT )
   Authid(          )             Priority( High )
   Authtype( Userid )             Purgecyclem( 00 )
   Comauthid(          )          Purgecycles( 30 )
   Comauthtype( Cuserid )         Resyncmember( Resync )
   Comthreadlim( 0001 )           Signid( CHABERT  )
   Comthreads(0000)               Security(              )
   Connecterror( Sqlcode )        Standbymode( Reconnect )
   Connectst( Connected )         Statsqueue( CDB2 )
   Db2groupid( D80G )             Tcblimit( 0012 )
   Db2id( D82G )                  Tcbs(0000)
   Db2release(0810)               Threaderror(N906d)
   Drollback(Rollback)            Threadlimit( 0003 )
   Msgqueue1( CDB2 )              Threads(0000)
   Msgqueue2(      )              Threadwait( Twait )
   Msgqueue3(      )
   Nontermrel( Release )
   Plan(          )

                                                   SYSID=C22F APPLID=CICSC22F
  RESPONSE: NORMAL                            TIME:  17.10.56  DATE: 11.04.04
PF 1 HELP       3 END       5 VAR        7 SBH 8 SFH 9 MSG 10 SB 11 SF
                                                                      01/012
```

Figure 37-14 AM: Sysplex: CICS DB2Conn definition

Only the DB2groupid has to be specified. CICS determines which member (DB2id) it has to connect with when the DB2conn is installed.

37.2.2 Application Monitor set-up

Application Monitor has been customized to access the DB2 data sharing group, and not a specific member.

We select Application Monitor primary option **6**, Set Your Defaults, then option **1**, Specify Default Sampling Parameters, to specify our DB2 data sharing group name as Default DB2 SSID as displayed in Figure 37-15.

```
FBIP6100 IBM AM for z/OS - Specify Default Sampling Parameters ---------------

Option ===> _____________________________________________________________


Update the defaults below and then press PF3 or select an option

   Max Duration ==> 999      (1-999 minutes)
   Max Samples  ==> 99999999 (100-99999999)
   Analyze DB2 activity for every job   /   Default DB2 SSID==> D80G
   Analyze Delays for every job         /
   Analyze Data Sets for every job      /

   1 Reinstate your profile defaults
   2 Clear your defaults to built-in default values


Enter END (PF3) to save your updates and go back




                                                                    03/014
```

Figure 37-15 AM: Sysplex: Specify Default Sampling Parameters

When the desired defaults have been typed we press PF3 (END) to save them in our ISPF profile and go back.

Then we have to monitor our CICS, using one of the following options:

- Primary option **1**, Select an Active Job to Analyze
- Primary option **2**, Analyze a Specified Job
- Primary option **3**, Analyze a Named Set of Jobs (if the application runs in a CICSplex configuration)
- Primary option **4**, Schedule a Deferred Monitoring Request

Whichever option we use, monitoring data can be kept in a historical data set. If we keep the historical data set we can go back later and analyze this session's data again using Application Monitor primary option **5**, Analyze Historical data.

37.2.3 Application Monitor reports

In our case, we used primary option **1**, Select an Active Job to Analyze, then we ran our CICS application and we kept the historical data set. The following reports are from this historical data set. The first line of each display shows its name.

The purpose of this scenario is to show how Application Monitor reports performance data collected for an application accessing a DB2 data sharing group. Therefore, we do not analyze all available reports here; instead, we focus only on DB2 reports.

Specify the Time Intervals to Analyze

Figure 37-16 shows that Application Monitor collects data during more than four minutes.

```
FBIP1H00 IBM AM for z/OS - Specify the Time Intervals to Analyze -------------
                                                        Row 1 of 1
History Data Set: CHABERT.AM.CICSC22F.D2004309.T1705296
Time spanned:     2004/11/04 17:05:30 to 2004/11/04 17:09:06
  Named Set:  N/A

Specify the time range and analysis interval and then select an option

   Start Date ==> 2004/11/04 Time ==> 17:05:30
   End Date   ==> 2004/11/04 Time ==> 17:09:06
   Interval   ==> 1______ Time Units: / Minutes _ Seconds

   1 Analyze the intervals in the specified time range
   X Go back without analyzing (PF3)

Begin    End      System   Job      Step     Pgm  CICS IMS  DB2  Dly  Dsn
-------- -------- -------- -------- -------- Rpts Rpts Rpts Rpts Rpts Rpts
17:05:30 17:09:06 STLABF6  CICSC22F CICSC22F YES  YES       YES  YES  YES
****************************** BOTTOM OF DATA ******************************



Enter END (PF3) to go back without analysis
Option ===> ____________________________________________________ Scroll ==> CSR
                                                                     24/014
```

Figure 37-16 AM: Sysplex: Specify the Time Intervals to Analyze

Because our CICS/DB2 transaction ran in less than 4 minutes, we set the Interval to 1 and Units to Minutes. The data that is extracted for display will be taken from the samples that Application Monitor collected over successive 1 minute intervals.

Historical reports available

Figure 37-17 shows an example of a historical report.

```
FBIP1H10 IBM AM for z/OS - Historical Reports Available ----------------------
         2004/11/04 17:05:30 to 2004/11/05 17:06:49       Row 1 of 1
History Data Set: CHABERT.AM.CICSC22F.D2004309.T1705296
  Interval Start: 2004/11/04  17:05:30
   Interval Stop: 2004/11/04  17:06:30

System   Jobname  Stepname Pgm  CICS IMS  DB2  Dly  Dsn  CPU%   Sample
-------- -------- -------- Rpts Rpts Rpts Rpts Rpts Rpts ------  Count
STLABF6  CICSC22F CICSC22F    5 YES       YES  YES  YES    0.87    132
****************************** BOTTOM OF DATA ******************************









Command ===> ___________________________________________________ Scroll ==> CSR
                                                                     24/015
```

Figure 37-17 AM: Sysplex - Historical Reports Available

The Interval Start and Interval Stop fields show which interval Application Monitor uses to create reports.

From Figure 37-17, we cursor-select the YES value in the DB2 Rpts to drill down to the SQL Analysis report as shown in Figure 37-18.

```
FBIP1C00 IBM AM for z/OS - SQL Analysis ------------------------------------
         2004/11/04 17:05:30 to 2004/11/04 17:06:30       Row 1 of 5
History Data Set: CHABERT.AM.CICSC22F.D2004309.T1705296
    Jobname: CICSC22F        Stepname: CICSC22F          System: STLABF6
    LU Name: DB2D82G         DB2 SSID: D82G              Module: MYTRADD
  Plan Name: MYTRADD         Auth Id.: CHABERT         Oper Id.: CHABERT
DB2  Auth-ID  Plan     DBRM     St No O SQLRC Type     Count Rel. DB2 CPU Time
---- -------- -------- -------- ----- - ----- -------- ----- <---------------->
D82G CHABERT  MYTRADD  MYTRADD    477 S     0 SELECT       7 ==>
D82G CHABERT  MYTRADD  MYTRADD    533 S     0 UPDATE       3 >
D82G CHABERT  MYTRADD  MYTRADD    517 S     0 INSERT       2 ==>
D82G CHABERT  MYTRADD  MYTRADD    550 S     0 SELECT       8 ====>
D82G CHABERT  MYTRADD  MYTRADD    477 S   100 SELECT       6 =>
****************************** BOTTOM OF DATA ******************************

Command ===> _____________________________________________ Scroll ==> CSR
```

Figure 37-18 AM: Sysplex - SQL Analysis

The first part of this display shows which member (D82G) of the DB2 data sharing group had been used to perform SQL calls.

All other fields have already been explained (see Chapter 11, “Application Monitor: Analyzing a job with subsystems” on page 149). Statement number 550 has the highest percentage of CPU time, so we cursor-select it and press Enter to drill down to the SQL Statement Summary as shown in Figure 37-19.

```
FBIP1D00 IBM AM for z/OS - SQL Statement Summary ----------------------------
         2004/11/04 17:05:30 to 2004/11/04 17:06:30       Row 1 of 1
History Data Set: CHABERT.AM.CICSC22F.D2004309.T1705296
    Jobname: CICSC22F        Stepname: CICSC22F          System: STLABF6
    LU Name: DB2D82G         DB2 SSID: D82G              Module: MYTRADD
  Plan Name: MYTRADD         Auth Id.: CHABERT         Oper Id.: CHABERT
  Statement: 550                  Type: SELECT
X  <-------------------- S T A T E M E N T   T E X T------------------------>
.  SELECT SHR_VAL , SHR_VAL1 , SHR_VAL2 , SHR_VAL3 , SHR_VAL4 , SHR_VAL5 , SH
****************************** BOTTOM OF DATA ******************************

Command ===> _____________________________________________ Scroll ==> CSR
                                                                   24/016
```

Figure 37-19 AM: Sysplex - SQL Statement Summary

To drill down to the SQL Detailed Explain Analysis we place the cursor on the X selection field and we press Enter; the panel in Figure 37-20 is displayed.

```
FBIP1D00 IBM AM for z/OS - SQL Detailed Explain Analysis ---------------------
         2004/11/04 17:05:30 to 2004/11/04 17:06:30       Row 1 of 2
History Data Set: CHABERT.AM.CICSC22F.D2004309.T1705296
    Jobname: CICSC22F        Stepname: CICSC22F          System: STLABF6
    LU Name: DB2D82G         DB2 SSID: D82G              Module: MYTRADD
  Plan Name: MYTRADD         Auth Id.: CHABERT         Oper Id.: CHABERT
  Statement: 550                 Type: SELECT
Table                Plan Method Match Acc  Join Index UQ JN OB GB   UQ JN OB GB
Name                  Nbr          Col Type Grp. Only
CUSTOMER_DETAILS        2      0     0         0
COMPANY_DETAILS         1      0     0         0
******************************** BOTTOM OF DATA ********************************

Command ===> ____________________________________________________ Scroll ==> CSR
                                                                        10/002
```

Figure 37-20 AM: Sysplex: SQL Detailed Explain Analysis

We can cursor select one of these tables to drill down to the DB2 Table Information as shown in Figure 37-21.

```
FBIPXPLA IBM AM for z/OS - DB2 Table Information ----------------------------
         2004/11/04 17:05:30 to 2004/11/04 17:06:30
History Data Set: CHABERT.AM.CICSC22F.D2004309.T1705296
Command ===> ____________________________________________________________________
                                                                  More:     +
    Jobname: CICSC22F        Stepname: CICSC22F          System: STLABF6
    LU Name: DB2D82G         DB2 SSID: D82G              Module: MYTRADD
  Plan Name: MYTRADD         Auth Id.: CHABERT         Oper Id.: CHABERT
  Statement: 550                 Type: SELECT
                            Table Name: COMPANY_DETAILS
Plan Number .: 1          Mix OP Seq. .: 0          Table Number.: 0
Table Type. .: T          Table Stat. .: X          Edit Proc . .:
Valid Proc. .:            Database ID .: 4          Column count.: 11
Table OBID. .: 12         Percent Pages: -1         Uniq Index ID: 13
Key Columns .: 1          Parents . . .: 0          Children. . .: 0
Record Length . .: 66                   % Compression . .: -1
Creator . . . . .: CHABERT              Creation Tbl Spac: TRADER
Creation Database: DSNDB04              Base Table . . . .
Creation RBA. . .: BC1284A2CCC6         Cardinality . . .: -1
Alter RBA . . . .: BC1284A2CCC6         Pages with rows .: -1
Check Flag. . . .:                      Check RID . . . .:
Data Capture. . .:                      Auditing. . . . .:
Created by  . . .: CHABERT              Create Date . . .: 2004-11-04-16.32.11
Alter Date stamp.: 2004-11-04-16.32.11 Runstats Date . .: 0001-01-01-00.00.00
                                                                        04/015
```

Figure 37-21 AM: Sysplex - DB2 Table Information

Several fields have a value of -1, which means that statistics have not been gathered. We have to schedule a DB2 RunStat, so next time Application Monitor will display more useful informations for such fields.

37.2.4 Conclusion

Application Monitor does not require any special customization to monitor a DB2 data sharing group.

We specify the DB2 data sharing group name as DB2 SSID using any of the following Application Monitor options:

- 6.1 - Specify Default Sampling Parameters if we monitor a job using option 1 Select an Active Job to Analyze
- 2 - Analyze a Specified Job if we the job to be analyzed and if we need to override the monitoring defaults
- 4.2 - Specify Deferred Job Details if we have to override the monitoring default for a deferred monitoring request.
- Through the Specify a Named Set Entry display when we create a named set.

Part 9

Using WSED with Debug Tool

Part 9 covers the use of Debug Tool and DB2 stored procedures and WebSphere Studio Enterprise Developer (WSED).

The chapters cover:

- Remote development - the debugger
- Using PL/I or COBOL DB2 stored procedures

38

Remote development: Debugger

This chapter describes the functionality that the debugger offers when used in combination with Debug Tool on the mainframe.

Remote Debugging means that the application runs on the host, whereas the control of the debug session is on the client workstation. WebSphere Studio Enterprise Developer (WSED) takes care of that control.

This chapter covers the functionality that comes with the WSED debugger. The debugger is invoked by using batch jobs or the DTCN panel from CICS.

Refer to redbook “SG24-6111 VisualAge COBOL Remote Edit/Compile/Debug - Taking COBOL to the Next Level” for complete coverage of how to compile and run programs in debug mode.

38.1 Overview of the Debug perspective

The Debug perspective is used when a debug session is started. Regarding Remote Debugging, this means that the perspective is activated when the Listener detects that a communication request arrives from the mainframe.

The Listener is found in the Debug View and is shown in Figure 38-1.

Figure 38-1 The Listener icon in the Debug View

The Listener can be in one of two possible states: It is either active or inactive. How to tell what the status is can be seen in Figure 38-2 and Figure 38-3. When the cursor is located on top of the Listener icon and the text `Start listening on port: 8001` is displayed, the Listener is inactive as shown in Figure 38-2.

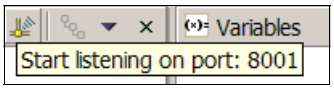


Figure 38-2 The inactive listener

As long as the Listener is inactive, no remote debug session is established. To activate the Listener, click the icon and it appears as a pushed button. To check whether the icon has activated the Listener, position the cursor once more on top of the icon. The text `Stop listening on port: 8001` should appear as shown in Figure 38-3.

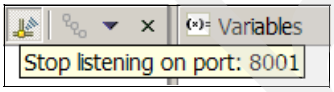


Figure 38-3 The active listener

The port number depends on the setting as arranged by the system administrators. If you want to change the port number do the following: Select **Window** → **Preferences**, expand the **Debug** entry, and click **Debug Daemon**. The Daemon Port number is now presented in the right half of the window as shown in Figure 38-4. Change the default value of 8001 to the value you want and click **OK**.

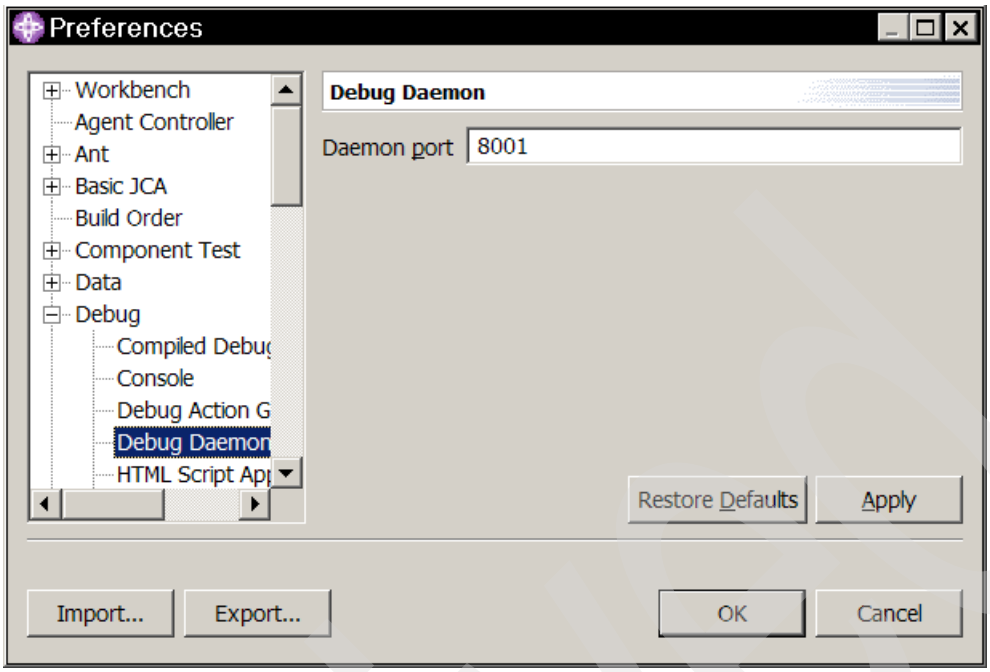


Figure 38-4 Preference window to change the Debug Daemon port number

Having the correct port number specified, you now can receive a debug session. How the debugger is invoked is covered in an earlier chapter. The debug perspective is activated and gets the focus. The perspective contains a large number of views as shown in Figure 38-5.

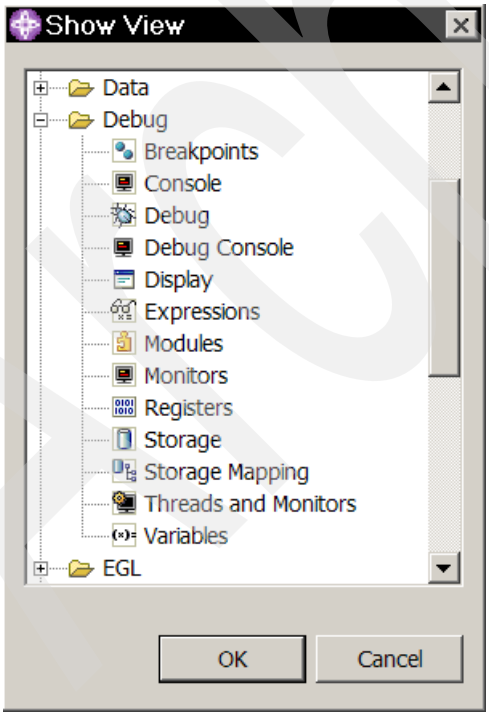


Figure 38-5 The possible views in the Debug perspective

The perspective itself, with some of the views, is presented in Figure 38-6.

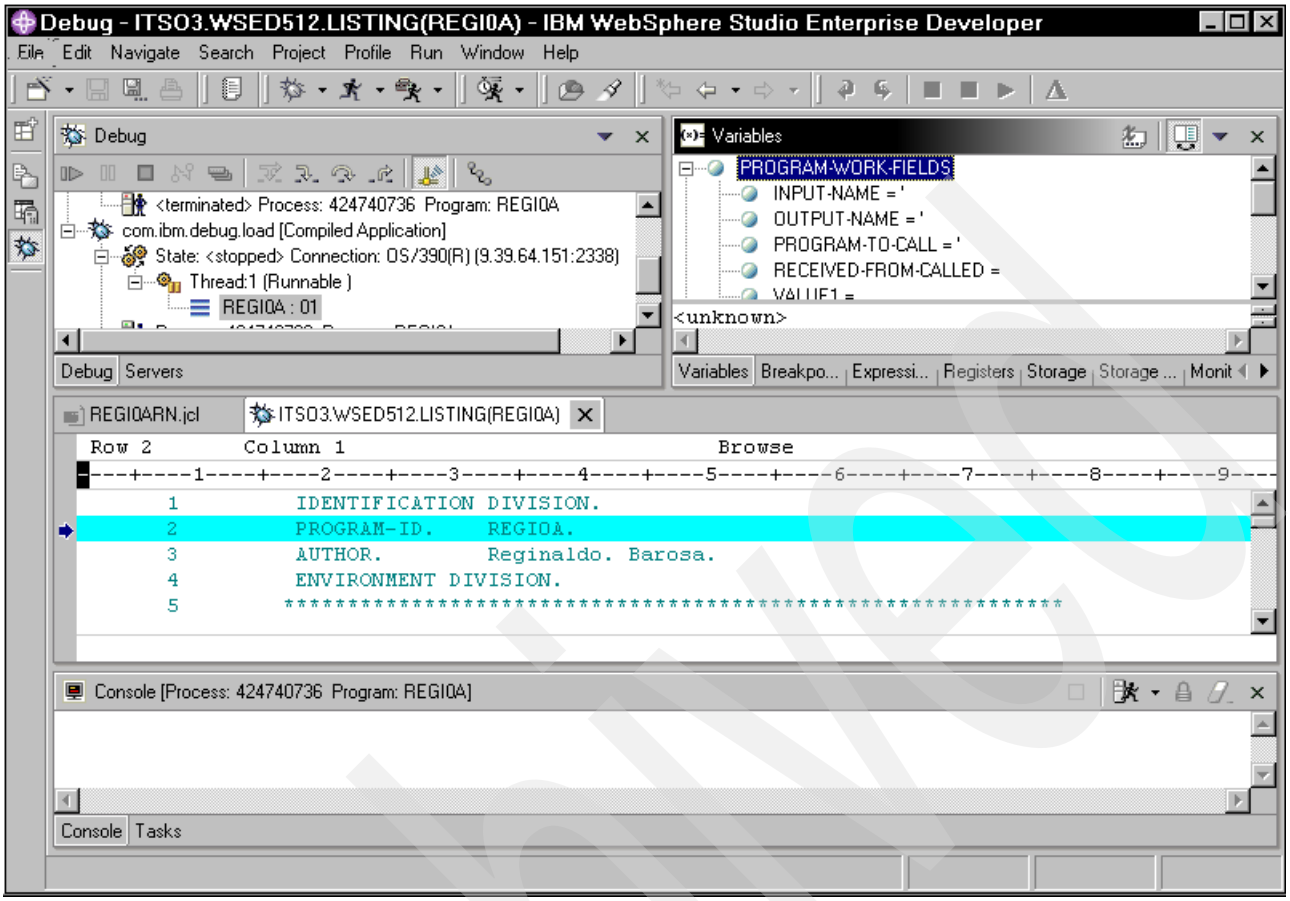


Figure 38-6 The Debug Perspective in an overview

The perspective contains a lot of information which at first looks rather confusing. Let us try to make things clear by explaining each view of this perspective.

We can roughly say that the Debug perspective by default has four areas:

- The upper left corner is the Debug view.
- The upper right pane contains the following views: Variables, Breakpoints, Expressions, Registers, Storage, Storage Mapping, Monitors, Modules, Threads and Monitors, and Display.
- The middle pane contains the source code or listing, which is displayed in the Editor view.
- The lower pane contains the Console and the Debug Console views.

Each view is briefly explained in the following paragraphs. Apart from the “real” Debug views, some other views can be present in the Debug Perspective. The following views can be seen, or they can be closed because they are not used by the debugger: Tasks, Servers, and Outline views.

Debug view

This is one of the required views. The Debug view allows you to manage the debugging of a program in the workbench. It displays the stack frame for the suspended threads for each target you are debugging. Each thread in your program appears as a node in the tree. If the thread is suspended, its stack frames are shown as child elements.

If the resource containing a selected thread is not open or active, the file opens in the editor and becomes active, focusing on the source with which the thread is associated.

In the Debug view, you can perform various start, step, and terminate debug actions. How to start, step and terminate debug actions is discussed in 38.2, “Controlling the flow” on page 948.

Editor view

The Editor view is another required view. It displays the source listing showing the current position in the debug session.

Console view

This view is currently not used by the debugger. It can be closed.

Debug Console view

This view is optional. It can be used to enter Debug Tool commands. Many of these commands are now available in WSED version 5.1.2 and are discussed later in this chapter.

Variables view

This view displays information about the variables in the currently selected stack frame. For a COBOL program this means that all variables in the Working Storage Section are displayed. It is an optional view that is not recommended for use due to the fact that COBOL programs can contain many variables. It can slow down the debug stepping considerably. In case of large programs it is best to close this view.

Breakpoints view

The Breakpoints view lists all the breakpoints you have set in the workbench projects. You can double-click a breakpoint to display its location in the editor. In this view, you can also enable or disable breakpoints, delete them, or add new ones. All types of breakpoints are shown in this view.

Expressions view

This view is only used by Java, and will therefore not be covered here. The view can be closed.

Registers view

This is an optional view that can be used if the developer wants to look at the program’s register.

Storage view

This view displays the storage contents and is an optional view.

Storage Mapping view

This view also displays storage contents, where it can display storage using XML mapping files to lay out memory using data types.

Monitors view

This is another required view where the developer adds the variables and expressions that he wants to monitor during the debug session.

Modules view

An optional view which shows loaded compiled units and modules. The view can be used to add a breakpoint in a function.

Display view

This view displays the result of evaluating an expression in the context of the current stack frame. This is typically a view used in a Java application, so it will not be covered in this book.

Threads and Monitors view

This view is used for Java only and can be closed.

38.2 Controlling the flow

Once the debugger is running, a pop-up window appears stating that the environment is not yet fully initialized as shown in Figure 38-7.

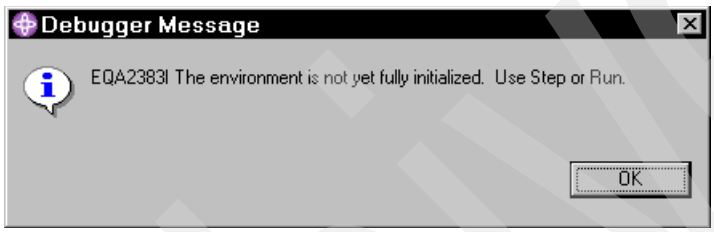


Figure 38-7 The initial message of a new debug session

This means that the debugger cannot yet fully determine the values of the various variables. By stepping into the program the initialization is taken care of. Therefore, click **OK**. The source listing is now presented in the Editor view, and we want to take over the control of the program flow. The debug session stops at the first line in the program, which is the Program ID line. From there on it is up to the developer how to continue the session.

38.2.1 Controls in the Debug View

The debug view has a number of icons in the title bar. They help you in deciding what to do next. The Debug View is displayed in Figure 38-8.

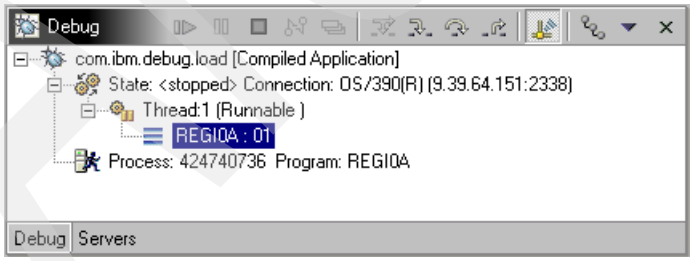


Figure 38-8 The Debug view with the icons in the title bar for controlling the flow

In the Debug View you can see the actual thread (REGI0A). It also shows that we debug a compiled application and that a connection with the mainframe exists where the application is executed.

Table 38-1 identifies the icons that are present in the title bar.

Table 38-1 Debug view icons

Icon	Title	Explanation
	Resume	The resume button (or the equivalent F8 key) executes the program until a breakpoint or exception is encountered. The program stops executing and waits for the user input.
	Pause	Clicking the Pause button will hold the execution of the program. Click Resume to continue the execution.
	Stop	This button terminates the debug session.
	Step Debug	The main purpose of the Step Debug is to step into the code that has debug information, whereas it will step over the code that does not contain debug information.
	Step Into	The Step Into action jumps to the next thread and resumes debugging that thread. That thread needs to be compiled for debugging, otherwise the program is executed but the debugger cannot take over the control. For COBOL programs it means that a called module will be debugged when a Step Into is issued at the line of the Call. If the line where the Step Into is executed is not a call to another module, the debugger stops at the next debuggable line.
	Step Over	The Step Over steps to the next executable line. In case that line contains a call to another module the Step Over, in contrast to the Step Into, will not hand over the control to the next executed module (thread). Instead, it means that the called program is executed and the control is handed over to the calling program once the called program is finished executing.
	Step Return	The Step Return hands control back to the calling program in case another module is called.
	Disconnect	This icon terminates the debug session because the connection between the client debugger and the z/OS Debug Tool is stopped.
	Remove all Terminated Launches	Every debug session leaves an entry in the Debug View. The icon makes sure that all entries from previous sessions are deleted.
	Enable/Disable Step-by-Step Debug	This is an action that is only available for Java programs. We skip it for COBOL debugging.

Apart from the icons on the title bar, there are other options to make the program execute in a specific manner. For instance, the Run To Location action makes the program execute until the line where the Run To Location is issued from is reached.

To use the Run to Location feature go to the line where you want the debugger to stop, right-click, and select Run To Location as shown in Figure 38-9.

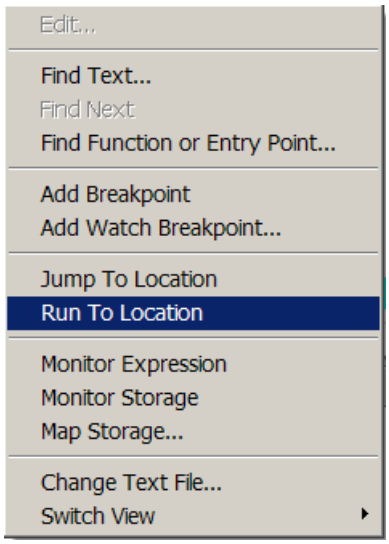


Figure 38-9 The Run To Location menu option

An alternative to the Run To Location is the action Jump To Location. The difference is that the Jump To Location jumps to the line where the action is issued from without executing any line between the current line and the Jump To Location line. The Jump To Location is found and executed in the same way as the Run To Location.

Where the Step functions are used to take one step at a time, the breakpoints work differently. Breakpoints are conditions set by the user that tell the debugger to stop in case those conditions are met. Various types of breakpoints can be used with the WSED debugger, they are described in the following sections.

38.2.2 Entry breakpoint

With an *entry breakpoint* the debugger stops as soon as the entry point of that program is reached. The debugger stops at the line where the Program ID is specified. An entry breakpoint can be added in the Breakpoints view. It is possible to add a single entry breakpoint, but it is also possible to have the debugger stop at all function entries. To do this, right-click in the breakpoints view and select **Add Breakpoint** → **Stop At All Function Entries**. This will cause the debugger to stop at every module no matter what the name is.

Once this choice is made you will see a check mark in front of the menu choice as shown in Figure 38-10.

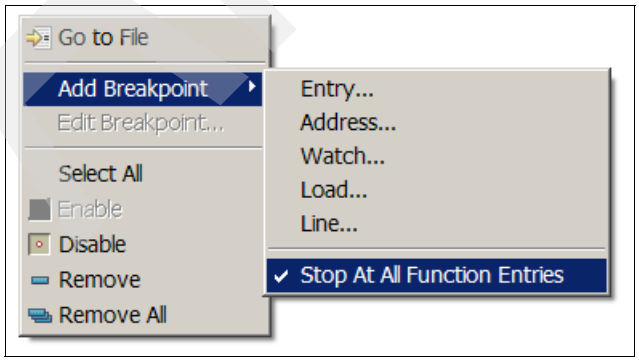


Figure 38-10 The choice for Stopping at all Function Entries

To deselect this choice, click **Stop At All Function Entries** once more, this removes the check mark. It can also be removed by selecting **Stop At All Function Entries** in the Breakpoints view and right-clicking **Remove**.

38.2.3 Address breakpoint

The debugger can stop at a specific address. In the Breakpoints view, right-click, then select **Add Breakpoint** → **Address**. Enter the address or expression and click **Finish**.

38.2.4 Watch breakpoint

A *watch breakpoint* can be used in two ways: the debugger can be stopped at any change of a variable or expression, or you can set a conditional breakpoint on that variable or expression. In the second case the debugger only stops when the condition is met.

Setting a watch breakpoint can be done from the listing pane or the breakpoints view. In the listing select a variable (in our case BRANCHFLAG) and right-click it, then select **Add Watch Breakpoint**. A pop-up window appears with the variable in the Address or Expression field as shown in Figure 38-11. Here you can also specify the number of bytes that need to be watched. It can be safely set to **Auto**.

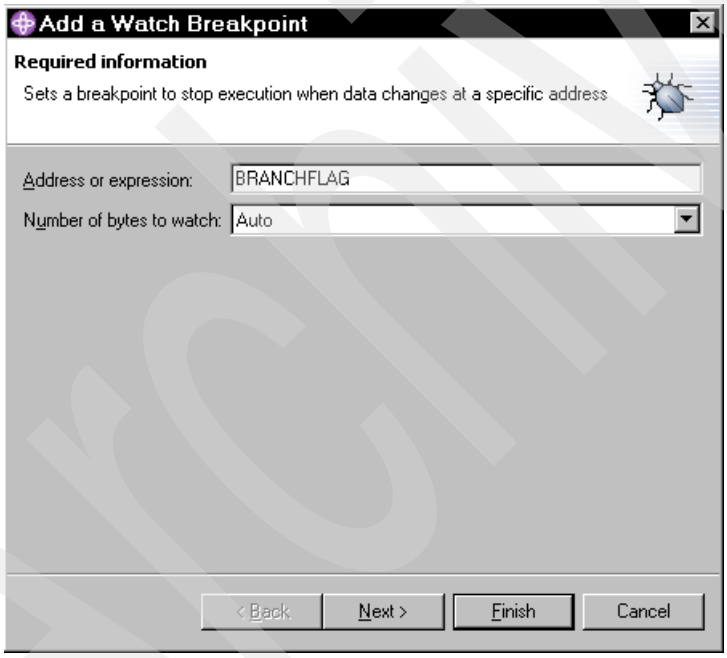


Figure 38-11 The first screen for adding a Watch Breakpoint

Click **Next** and Figure 38-12 in displayed. In this screen you can specify the expression that you want the debugger stop at. In our example we want the debugger to stop as soon as BRANCHFLAG reaches the value 2.

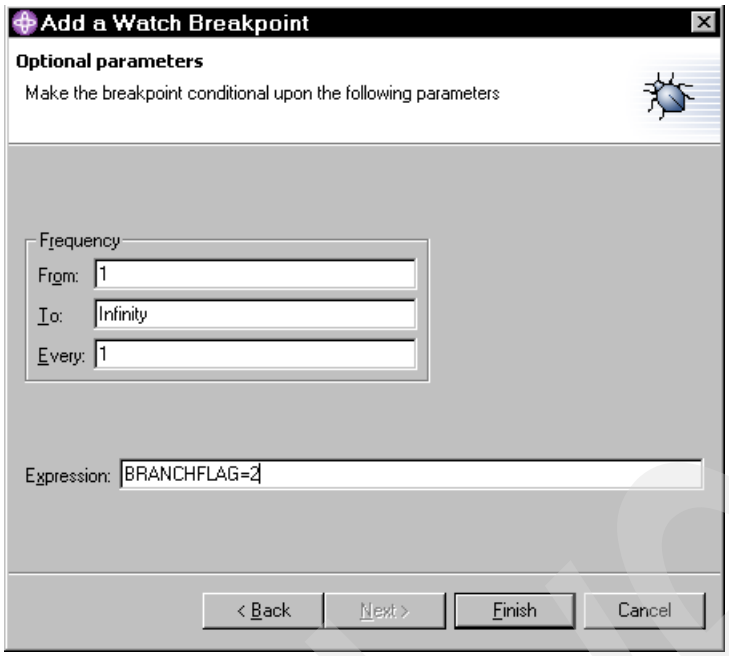


Figure 38-12 The second screen for adding a Watch Breakpoint

Click **Finish**; the breakpoint is now set.

To disable or remove a watch breakpoint go to the Breakpoints view, select the breakpoint and right-click the appropriate action.

38.2.5 Load breakpoint

A *load breakpoint* makes the debugger stop as soon as the load module of the debugged program is loaded in memory. The load is not activated at that moment. The next step will enter the program and stop the debugger at the Program ID line.

To add a load breakpoint, go to the Breakpoints view, right-click, and select **Add Breakpoint** → **Load**. Specify the name of the load module that you want the debugger to stop at and click **OK**.

A load breakpoint can also be removed or disabled.

38.2.6 Line breakpoint

With a *line breakpoint* the debugger will stop as soon as that line is reached.

A line breakpoint can be set in various ways. It can be done by double-clicking the line where you want the debugger to stop. The double-click must be done in the area just in front of the listing pane. Figure 38-13 shows the listing and the breakpoint icon in front of line 46. The place where the icon is located is the place where the double-click should be done.

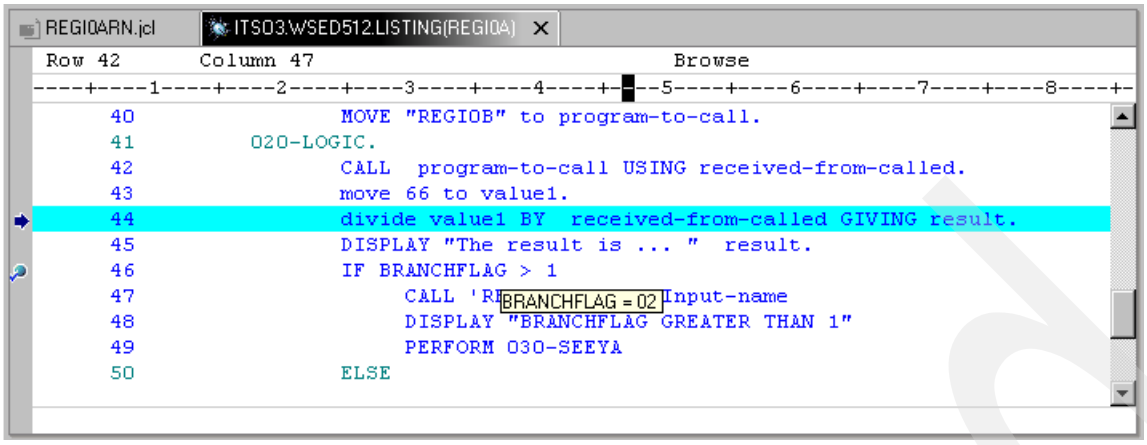


Figure 38-13 The Line Breakpoint on line 46

To remove the line breakpoint double-click in the same place. The icon will disappear.

You can also add a line breakpoint from the Breakpoints view. Right-click in the Breakpoints view, then select **Add Breakpoint** → **Line or Statement**. Specify the line number where you want to set the breakpoint in the pop-up window that is returned.

38.2.7 Disable breakpoints

As mentioned earlier, all breakpoints are listed in the Breakpoints view. Breakpoints can be added or removed in that view, but there is another way of dealing with breakpoints: you can disable and enable them. In the view, select each breakpoint that you want to disable and right-click **Disable**. The breakpoints that are disabled will turn white instead of being active with the blue color. The debugger will not stop at the breakpoint position when they are disabled.

An advantage of disabling breakpoints instead of removing them is that they can be activated again with the enable action. The same actions previously can be performed on the disabled breakpoints: select them and right-click **Enable**.

The Breakpoints view can look like the view in Figure 38-14 with a couple of different breakpoints.

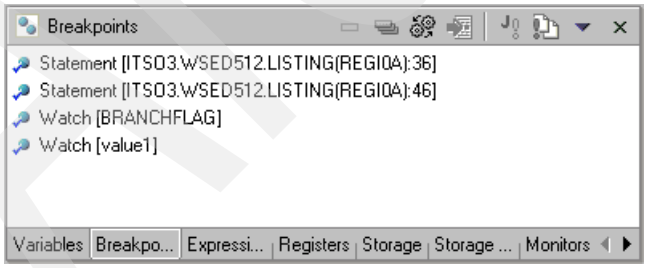


Figure 38-14 The Breakpoints view with various breakpoints

38.3 Dealing with variables

An important function of debugging is the monitoring of variables. We have a couple of views available that can be used to monitor and control the variables.

One of the fine features that works without a view is the Tool Tip Evaluation. You can place the cursor on a variable, leave it there for a short while and the variable and its current value are displayed in a hover text box. Figure 38-15 is an example; the cursor is positioned on top of the variable BRANCHFLAG that has the value 2.

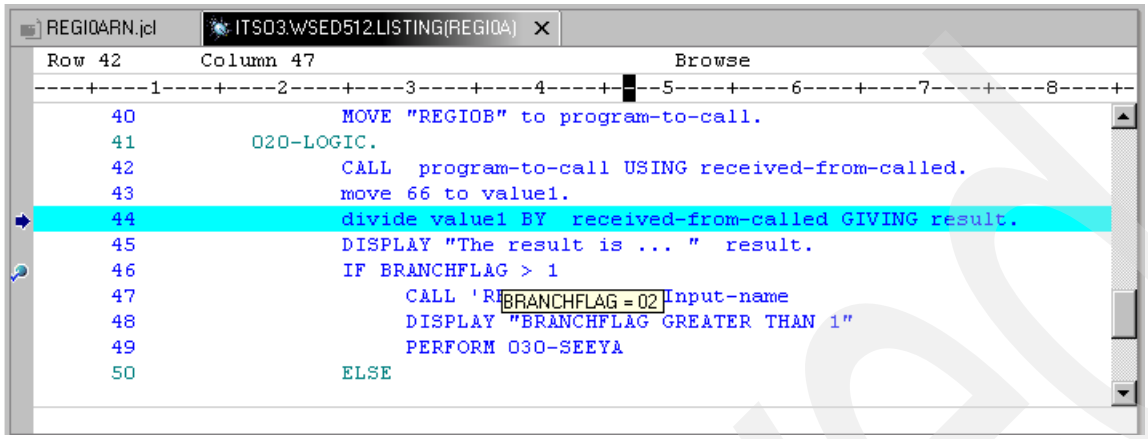


Figure 38-15 The listing with the active Tool Tip Evaluation

This means that at any time during debugging the value of the variable can be viewed without needing any monitor.

On the other hand, you may want to monitor variables throughout the session, which means that these variables should be in separate views in order to keep track of them. The two most important views with this in mind are the Variables view and the Monitors view. The Variables view contains all the variables that are present in the Working-Storage section of the program as shown in Figure 38-16.

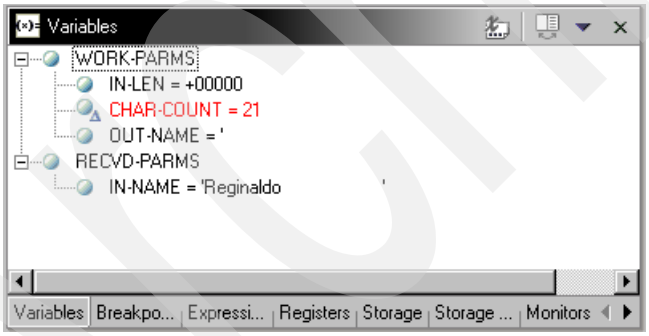


Figure 38-16 An example of the Variables view for a small program

Although it seems handy to have them all in one view, it can be very awkward when dealing with a large program.

Debugging a large program means many variables, which can make it difficult to find the variable that you want to have a look at. Another drawback is performance. Having the focus on the Variables view will cause the debugger to refresh all variables with every step that is taken during the debug session. This can degrade performance, especially when a large number of variables are involved.

Tip: Close the Variables view when debugging a program with many variables. Pick those variables that you want to keep track of and display them in the Monitors view.

The Monitors view is handy because it lets you display only those variables that you really want to monitor. The Monitors view is updated with every step that you take using the debugger as shown in Figure 38-17.

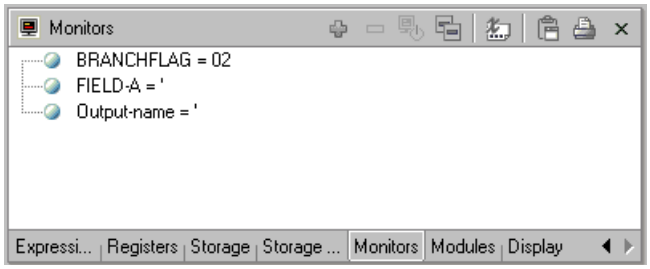


Figure 38-17 The Monitors view with variables chosen by the user

To place a variable in the Monitors view, select the variable by selecting the complete name. Right-click and select **Monitor expression**. The variable is instantly added to the view.

If you have enough information about a variable and you want to take it out of the Monitors view, select the variable, right-click, and select **Remove Monitored Expression**.

In both the Variables view and the Monitors view the representation of a variable can be changed. By default the representation is String. It can be changed to Hexadecimal. Click the variable and select **Change Representation** → **Hexadecimal** using the right mouse button as shown in Figure 38-18.

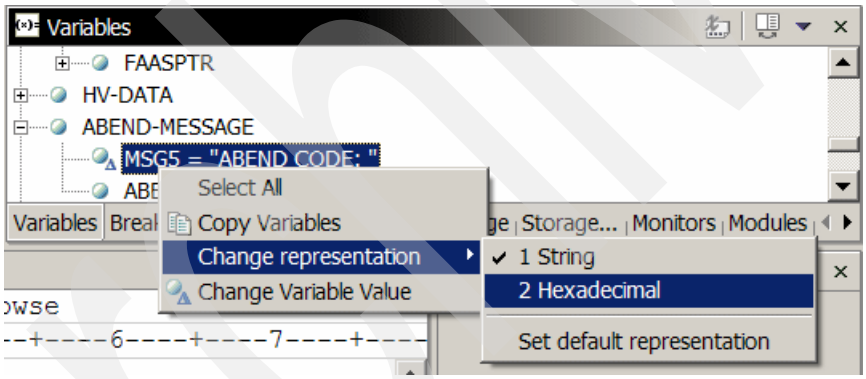


Figure 38-18 Change the representation of a variable

After the change the variable MSG5 is displayed as shown in Figure 38-19.

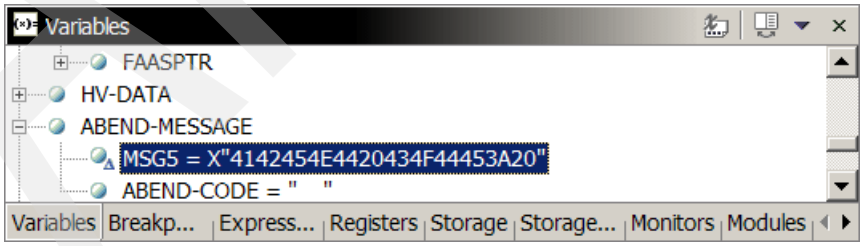


Figure 38-19 The changed representation in the Variables view

Another important feature is the possibility to modify the contents of a variable during the debug session. In the Variables view this is activated by right-clicking the variable and selecting **Change Variable Value**. A pop-up window displaying the current value of the variable is returned. Change the value as desired and click **OK**; the new value will be used in the continuation of the debugger.

Suppose we have a situation where variable BRANCHFLAG receives value 2, which causes the debugger to stop because of a conditional breakpoint. We want to change the value to 10 and we therefore select the variable and choose to change it. This is illustrated in Figure 38-20.

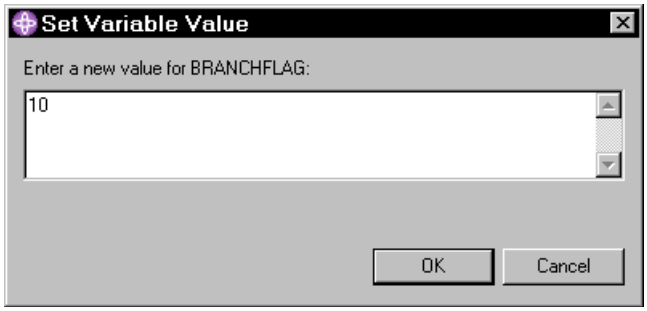


Figure 38-20 Set variable value window

The value is changed to 10. This new value is shown in the Variables view. Figure 38-21 shows the overview of the debug perspective, with the changed variable.

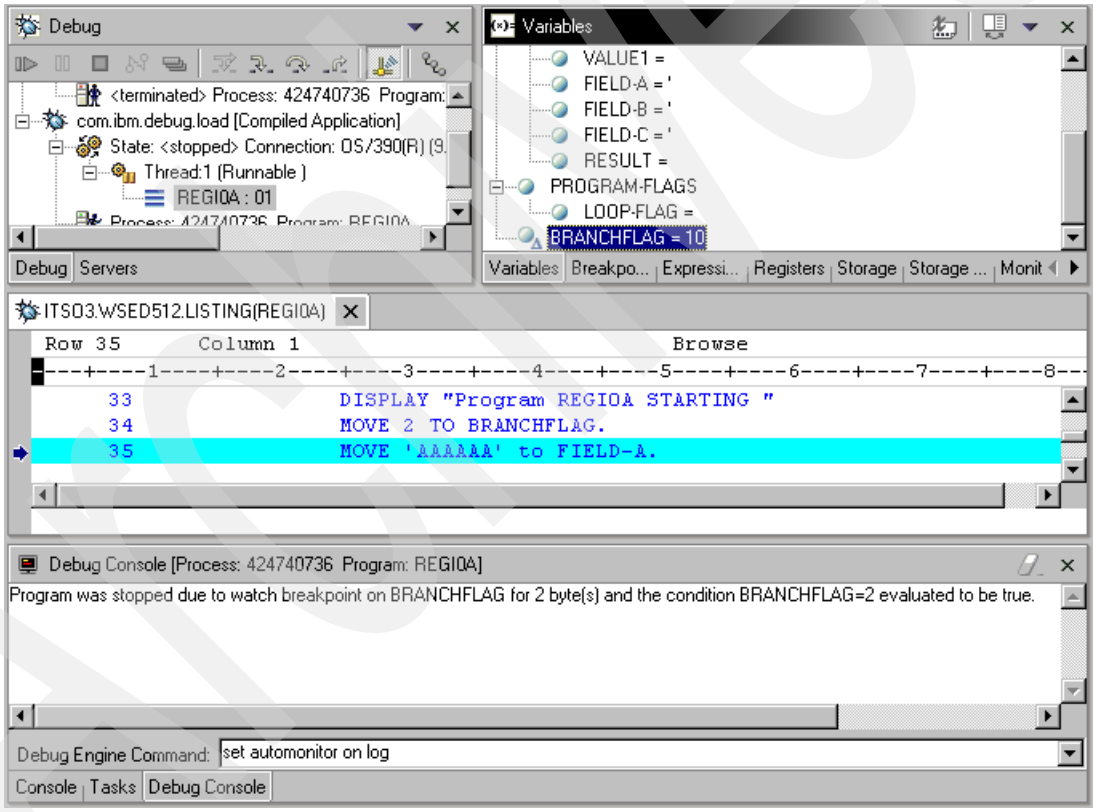


Figure 38-21 The Debug perspective with a changed variable

The debug session continues until the BRANCHFLAG receives value 10.

A very nice feature is the coloring of the variables in the Monitors view and the Variables view. Variables that were changed during the last step are presented in red to make it obvious that those variables have changed.

38.4 Debug Console

The Debug Console view is not automatically opened when the Debug perspective is activated. Open it by selecting **Window** → **Show View** → **Other**. Select **Debug Console** and click **OK** as shown in Figure 38-22.

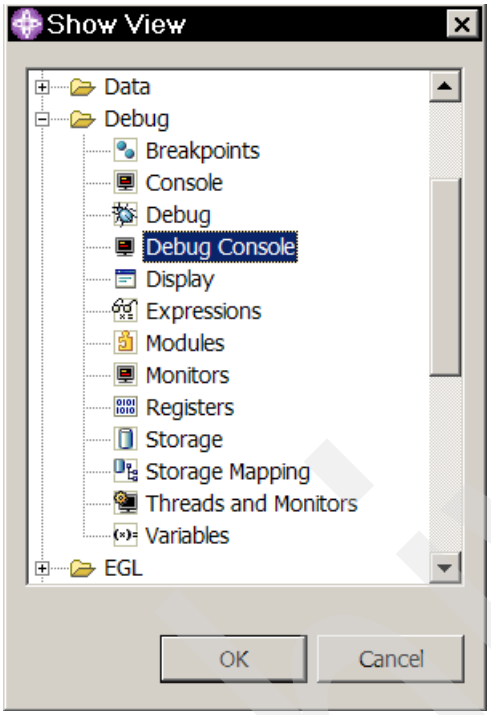


Figure 38-22 Select the Debug Console

The Debug Console is now opened at the bottom part of WSED. The Debug Console has functions that only work when Debug Tool version 5.1 is used on the mainframe.

The breakpoints that are encountered by the debugger are presented in the Debug Console instead of showing in pop-up windows, as shown in Figure 38-23. This means that we do not need to click these windows away.

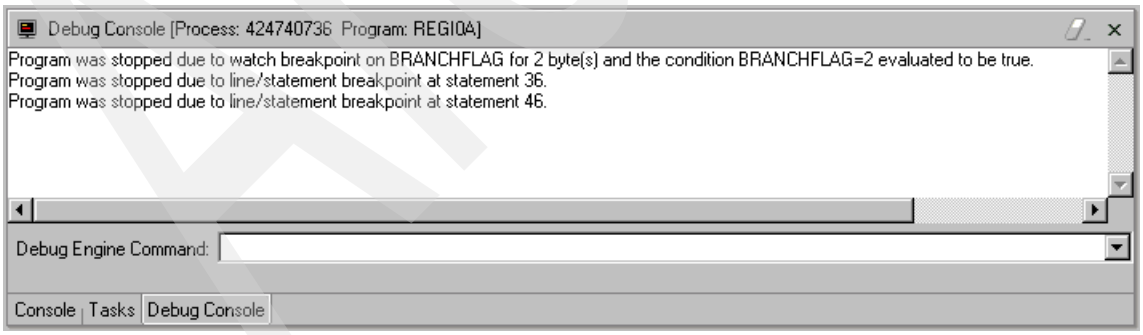


Figure 38-23 The Debug Console showing the breakpoints where the program stopped

Another feature within the Debug Console is the automonitor. It is activated by issuing the following Debug Engine Command:

```
set automonitor on log
```

Debug Console displays the command and the result of it, as shown in Figure 38-24.

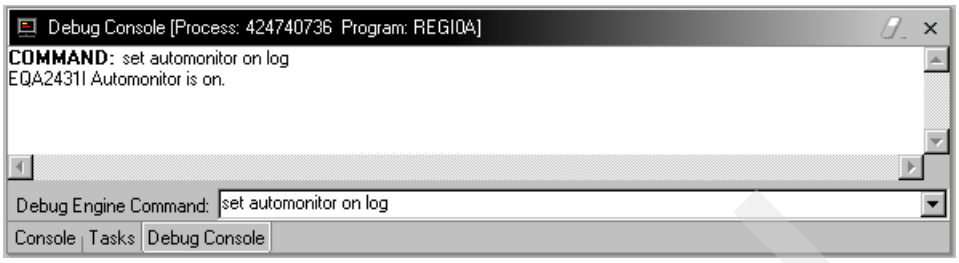


Figure 38-24 Activation of Automonitoring in the Debug Console

As a result, every line where the debugger stops due to breakpoints or steps is shown in the Debug Console. This presentation includes the line number and the contents of the variables which are present in that line. Figure 38-25 shows an example of the automonitor log. Several lines are shown, including the conditional breakpoint where the debugger was interrupted.

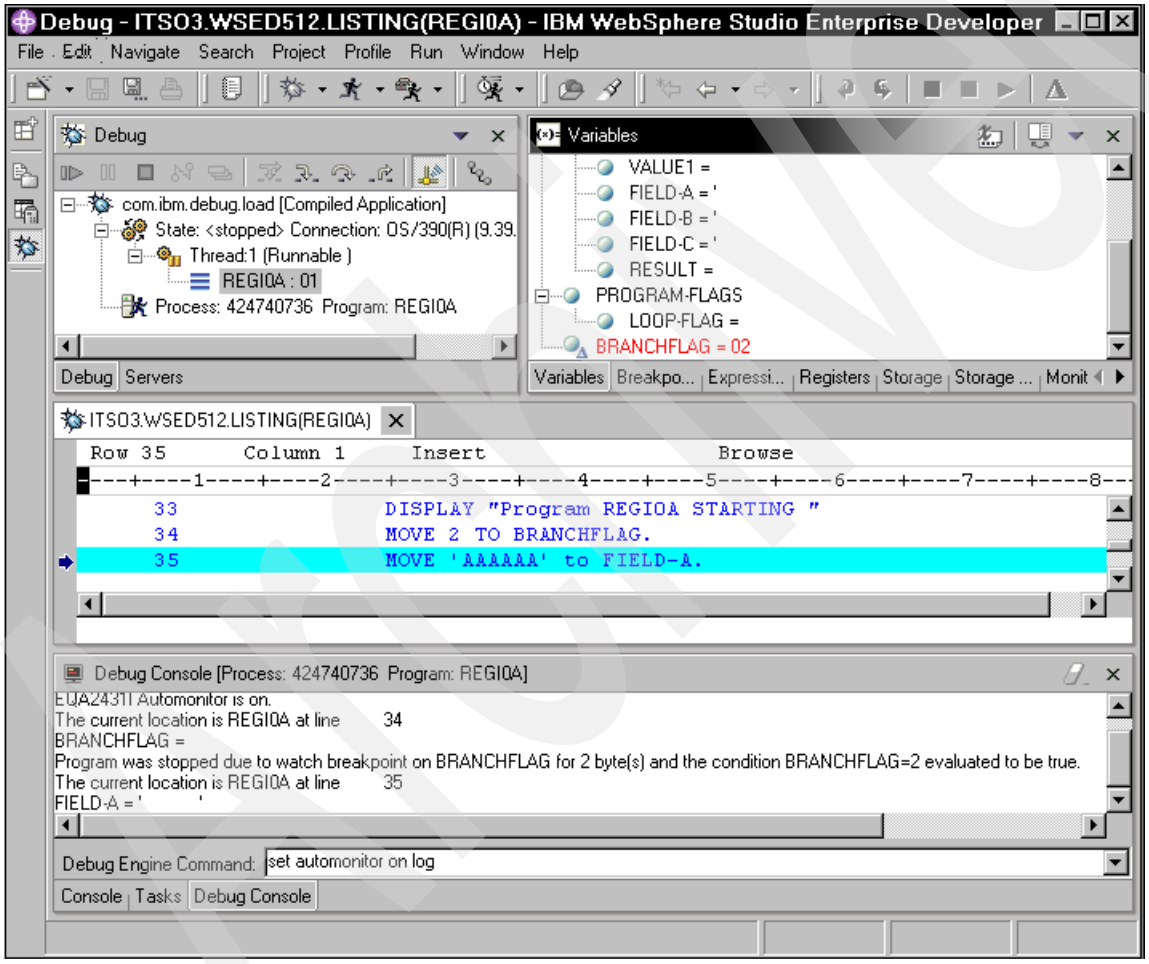


Figure 38-25 The Debug Console with the Automonitor function

Another Debug Engine Command is the description of the compiled unit that is currently active in the debug session. The command to gather the information is:

```
describe cus
```

It results in a display of important information about the program that is debugged. An example is presented in Figure 38-26.

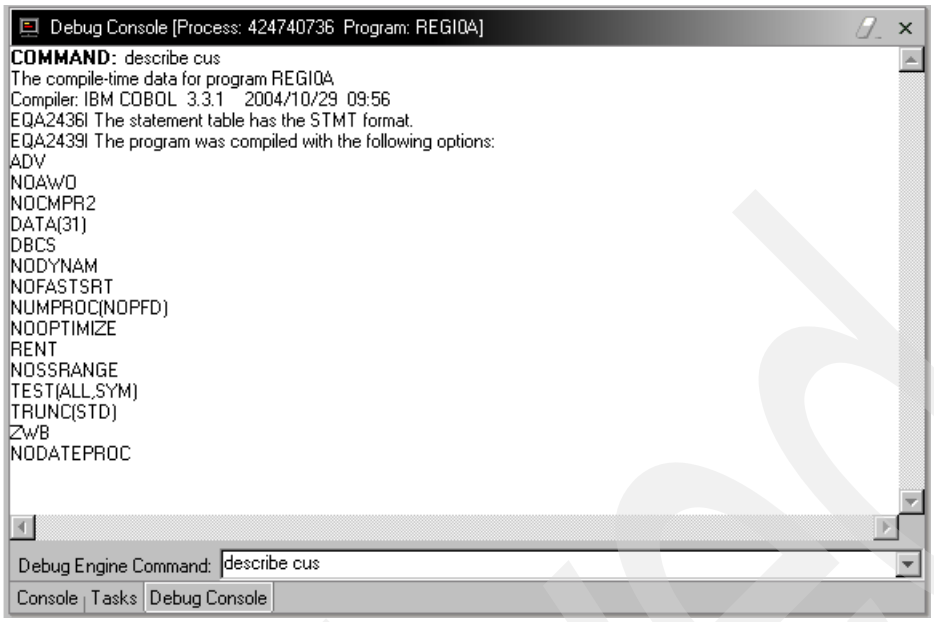


Figure 38-26 Compile Unit information in the Debug Console

It tells the date and time of the compilation, the version of the compiler and what the compiler options were during the compilation.

Finally, a command can be issued that tells the version of Debug Tool that is present on the host. The command is:

```
call %ver
```

This command shows the version of Debug Tool including all the PTFs that are applied to it as shown in Figure 38-27.

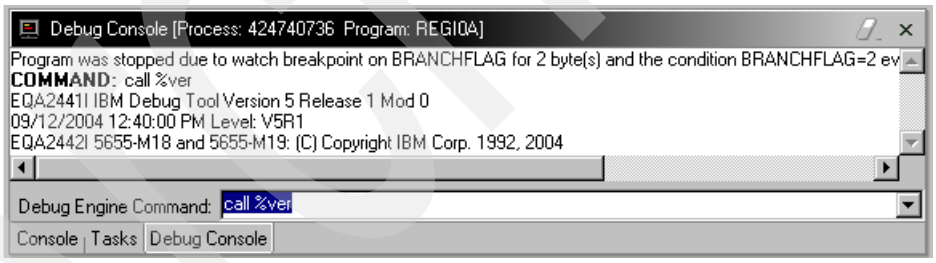


Figure 38-27 Retrieving Debug Tool version information from the host

38.5 Exception handling

The debugger allows you to investigate exceptions that occur while you are debugging a program.

Before we actually explain exception handling, we show you how to set the level of exceptions that you want to intercept. During an active debug session, switch to the Breakpoints view. In the upper left corner is the icon shown in Figure 38-28.

Figure 38-28 The icon to define the level of exception that is intercepted

Click this icon; a pop-up window appears in which the exception levels are displayed as shown in Figure 38-29. The exception levels varies with the platform on which you are running your debug session. Three levels are possible when using Remote Debugging:

- TEST(ALL)
- TEST(ERROR)
- TEST(NONE)

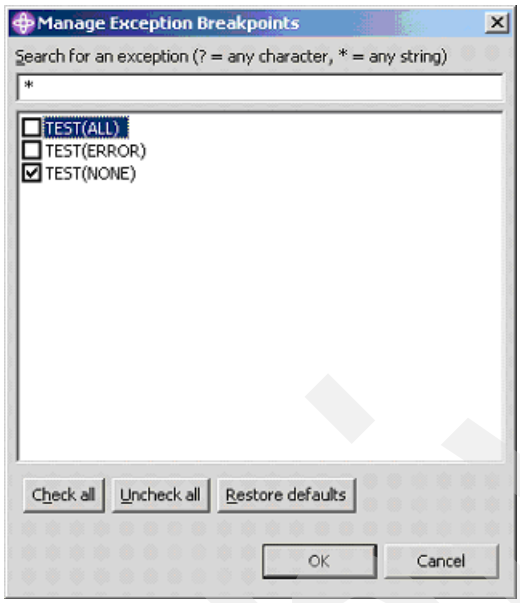


Figure 38-29 Manage Exception Breakpoints window

By default, the level is set to TEST(ALL). This means that every exception (for instance, a call to a submodule) is caught during debugging. This probably is not desired, and it might be wise to set the level to ERROR so the real problem situations are caught, or to NONE when you do not want exceptions to be interfering with your debugging.

When an exception occurs, there are three possible ways of continuation:

- Step Exception: This causes the debugger to step into the first registered exception handler (tracked by the operating system). Execution then stops at the first executable line of code in the exception handler. If no exception handler exists, the exception remains unhandled and the application may be terminated.
- Run Exception: This causes the debugger to run the exception handler that is registered to handle the type of exception encountered. It the application does not have a registered exception handler, the exception remains unhandled, and the application may be terminated.
- Retry Exception: This discards the exception and allows you to investigate the cause of the exception and retry program execution at the statement that triggered the exception. The debugger starts at this statement and attempts to continue.

If the exception level is set to ALL or ERROR, the window shown in Figure 38-30 appears when you encounter an exception.

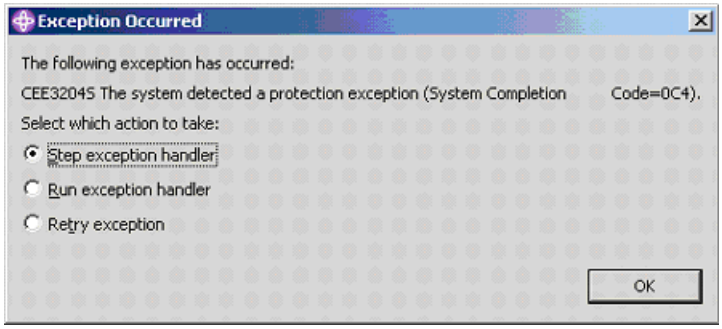


Figure 38-30 The exception screen where the action has to be selected

This example shows an 0C4 error. When the **Step exception handler** or **Run exception handler** action is selected, the exception handler on the host is activated if that handler is present. If the exception handler is not present the debug session is terminated.

The other option is **Retry exception**. This gives the opportunity to retry the code where the error occurred. Clicking **OK** may give an additional error message as displayed in Figure 38-31.

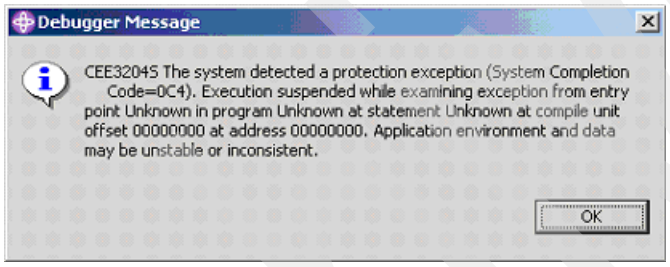


Figure 38-31 An additional message caused by an exception

It can be that, after reading the message, the user has a hunch of what might be the reason of the exception. Click **OK** and the control is back in the debug session at the line where the exception occurred. With the **Jump to Location** action the debug session is repositioned at a line before the line where the exception occurred.

One option is to change the variable value where the user expects the problem to be. After changing the value, a Step Into can be performed and you can see how the debugger reacts to that value.

A famous example of an exception is division by zero. When the exception occurs it is possible to jump to the line before the line where the exception occurred, change the value of the variable into something bigger than zero, and step through the code again. If the exception does not happen again, you know where to change the code in order to prevent that exception from happening again.

39

Using PL/I or COBOL DB2 stored procedures with WSED

This chapter contains all the necessary information to:

- Understand the DB2 Stored procedure
- Create new stored procedures using a wizard
- Build and register stored procedures on a z/OS system
- Run stored procedures on a z/OS system
- Debug stored procedures on a z/OS system
- View result sets, messages, and parameters in the Output view
- Drop stored procedures from the database

Attention: A number of automated viewlet demonstrations are provided as additional material to this redbook. The demonstrations cover:

- z/OS DB2 Stored Procedures using COBOL and Debug Tool V5
- EGL and DB2 on z/OS with Debug V5
- JSF calling EGL calling COBOL z/OS DB2 stored procedures with Debug Tool V5.

The demonstrations can be downloaded from the Redbook Web site by selecting "Additional Material" and navigating to IBM Redbook number SG24-6483 on the FTP site. The demonstrating are approximately 10 minutes in length.

39.1 Overview of DB2 stored procedures

A stored procedure is a user-written program that can be called by an application with a SQL CALL statement. It is a compiled program that is stored at a DB2 server, and can execute SQL statements. Figure 39-1 shows SQL calls with and without stored procedures.

Stored procedures can be called locally (on the same system where the application runs) and Remotely (from a different system). However, stored procedures are particularly useful in a Distributed environment since they considerably improve the performance of distributed Applications by:

- Reducing the traffic of information across the communication network
- Splitting the application logic and encouraging an even distribution of the computational workload
- Providing an easy way to call a remote program
- The advantages provided by stored procedures are clear when comparing them to a standard distributed application where the client may be a workstation as shown in Figure. We see that the client communicates with the server separately for each embedded SQL request.

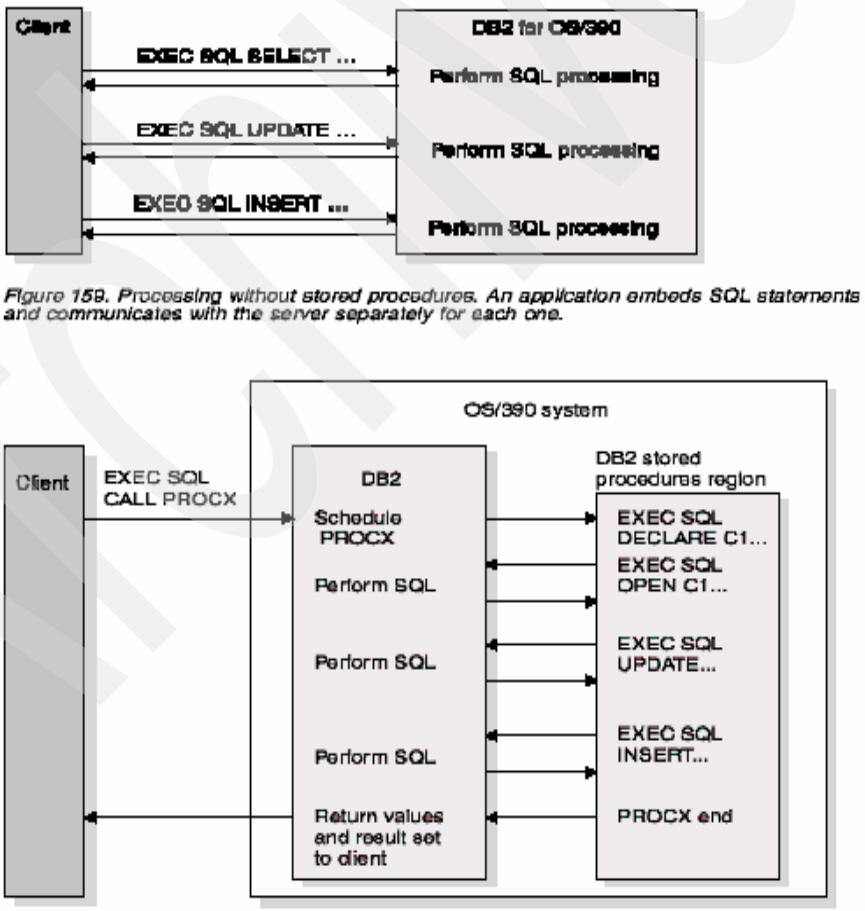


Figure 39-1 SQL calls with and without stored procedures

39.1.1 DB2 stored procedure types

There are two categories of stored procedures when using DB2 and z/OS:

External procedures The stored procedure is available as an external load module in an MVS library. For external stored procedure, definition and program logic are two separate components.

SQL procedures The SQL (or internal) procedure is entirely written in SQL. In this case SQL is the only language used, and the program logic is part of the definition.

In this chapter we focus in the external procedures only. Refer to the redbook "*DB2 for z/OS Stored Procedures: Through the CALL and Beyond,"* SG24-7083 for more details about DB2 stored procedures.

External procedures

An external stored procedure is written by a developer in one of the programming languages available on the server. The available languages on the z/OS server are: COBOL, PL/I, C, C++, Assembler, REXX, and Java. An external stored procedure is much like any other SQL application. Example 39-1 highlights the necessary components of a COBOL stored procedure. From this example you can see that the skeleton is identical to a COBOL subprogram.

WSED can help you to construct this code in COBOL or PL/I.

Example 39-1 COBOL skeleton of a storage procedure

```
ID DIVISION.
PROGRAM-ID. ITSOCOB.
...
...
LINKAGE SECTION.
   01 ...
PROCEDURE DIVISION USING ...
   EXEC SQL
   ...
   END-EXEC.
   GOBACK.
```

The source code for an external stored procedure is separate from the definition of the stored procedure. A stored procedure is only bound to a package and not a plan because it utilizes the invoking plan's thread. The stored procedure load module must be placed in a load library that is included in the STEPLIB DD concatenation in the WLM startup JCL.

The CREATE PROCEDURE statement is used to inform the system of the name of the load module and what parameters are expected when the procedure is called, as well as other execution and environment options.

Example 39-2 shows the information from a CREATE PROCEDURE statement that DB2 needs to locate the load module and to know what source language will be used to create the stored procedure. Stored procedures are DB2 objects; they must be defined with DDL.

Example 39-2 Sample storage procedure CREATE statement

```
CREATE PROCEDURE ITSOCOB
(parameter information)
EXTERNAL NAME ITSOCOB
LANGUAGE COBOL
```

39.2 Creating stored procedure using WSED

Before creating your stored procedure, you must have a source data set to store your procedure in. When creating a PL/I stored procedure, the source data set must have the WSED mapping of pli and when creating a COBOL stored procedure, the source data set must map to cbl.

39.2.1 Allocating the z/OS data sets for stored procedure

You need some data sets to store the generated COBOL or PL/I stored procedures. This can be done using WSED. For details, refer to WSED help.

In our examples we generate COBOL code only, but the PL/I procedure would be exactly the same. Use the following steps to allocate the COBOL data sets that you need:

1. Make sure that you are connected to the remote system and are working in the z/OS Systems perspective or the z/OS Projects perspective.
2. Select the z/OS Systems view and expand the list of remote systems. Right-click the z/OS system where you want the data set to reside, then click **Allocate PDS**.
3. At the Allocate PDS page, enter a data set name, like ITSO4, as High Level Qualifier and WSED512.SP.COBOL as Data set name. Click **Next**.
4. At the Data Set Allocation page, you can have different allocation options. For our example we allocated the data sets as shown in Table 39-1.

Table 39-1 Data sets to be used for COBOL stored procedures

Data Set Name	Category	Type
ITSO4.WSED512.COPYLIB	SOURCE	COBOL
ITSO4.WSED512.OBJS	OTHERS	OBJECT
ITSO4.WSED512.DBRMLIB	OTHERS	OBJECT
ITSO4.WSED512.LISTING	LISTING	COBOL

5. Figure 39-2 shows the results when all data sets are allocated.

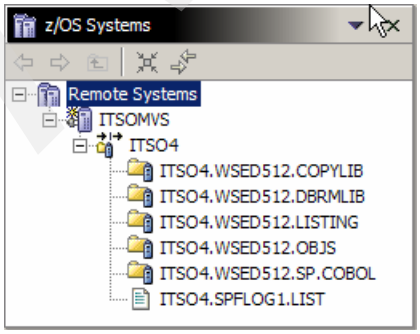


Figure 39-2 PDS allocated at the z/OS

39.2.2 Setting up the z/OS build properties

The data sets that are used to build a stored procedure on the z/OS system have to be specified in the z/OS build properties.

The z/OS build properties can be specified for a particular stored procedure or a set of stored procedures. In our example, the build properties are specified on a high-level qualifier, so they apply to all stored procedures under this high-level qualifier. However, they can also be specified the same way on a particular data set or member.

1. Right-click on a high-level qualifier (ITSO4), and select **Properties**.
2. Select **COBOL Settings** from the list in the left frame and enter the following settings:
   a. Compile Procedure Name: ELAXFCOC. The name of your compile procedure.
   b. Compile Procedure Step Name: COBOL. The Name of the compile step in your compile procedure.
3. Listing Output Data Set: ITSO4.WSED512.LISTING. This data set is required for debugging of COBOL stored procedures.
4. Object Deck Data Set: ITSO4.WSED512.OBJS. The name of the library in which the object decks of your stored procedures should be created.
5. Copy Libraries: ITSO4.WSED512.COPYLIB. The name of the libraries containing the copy members that are referenced by your stored procedures. We do not have copybooks, but we have this just in case we need it.
6. Click on the **DB2** tab, select the **Use DB2 (contains EXEC SQL statements)** check box, and enter the following setting:

   Database Request Module Location: ITSO4.WSED512.DBRMLIB. The name of the library in which the DBRM of your stored procedures should be created. If you will generate PL/I stored procedures, you will need to do the same as we did in the example for COBOL. In that case you use the PL/I Settings from the list in the left frame.
7. Still under the Properties dialog, select **Link Options** from the list in the left frame and enter the following settings:
   a. Linkage Editor Procedure Name: ELAXFLNK. The name of your link-edit procedure.
   b. Linkage Editor Procedure Step Name: LINK. The name of the link-edit step in your link-edit procedure.
   c. Link Libraries: CEE.SCEELKED and DB2.V7R1M0.SDSNLOAD. The name of the Language Environment Link Library and the DB2 Load Library.
   d. Load Module Location: DB2.V7R1M0.DSNA.RUNLIB.LOAD. The name of the library in which the load modules of your stored procedures should be created, this is the load library that is in the STEPLIB for the WLM environment in which this stored procedure will run.
8. Click **Apply**. If you have plans to this setup on other workstations, click **Export** and keep the generated XML for future use. Click **OK** to close this dialog.

Those steps must be done once on each workstation. At this point you are ready to start creating the stored procedure.

39.2.3 Connections required to z/OS when creating the stored procedures

Creating COBOL and PL/I OS/390 stored procedures using WSED requires two connections to z/OS:

1. One connection establishes a link between WSED and DB2 Universal Database server on z/OS, so that DB2 catalog information on tables, columns, stored procedures, and so forth can be displayed and operations (drop, import, sample contents, and the like) on these DB2 assets can be performed. This connection is done using the WSED Data Perspective and DB Servers view.
2. The other connection establishes a link between WSED and z/OS, so that the generated stored procedure can be stored on and read from z/OS. This connection can be done using the WSED z/OS Systems Perspective.

If you want both connections in the same window, use the Data Perspective and open the z/OS Systems view. Just select **Window** → **Show View** → **Other** → **z/OS Systems**. Figure 39-3 is an example when we have both connections in effect.

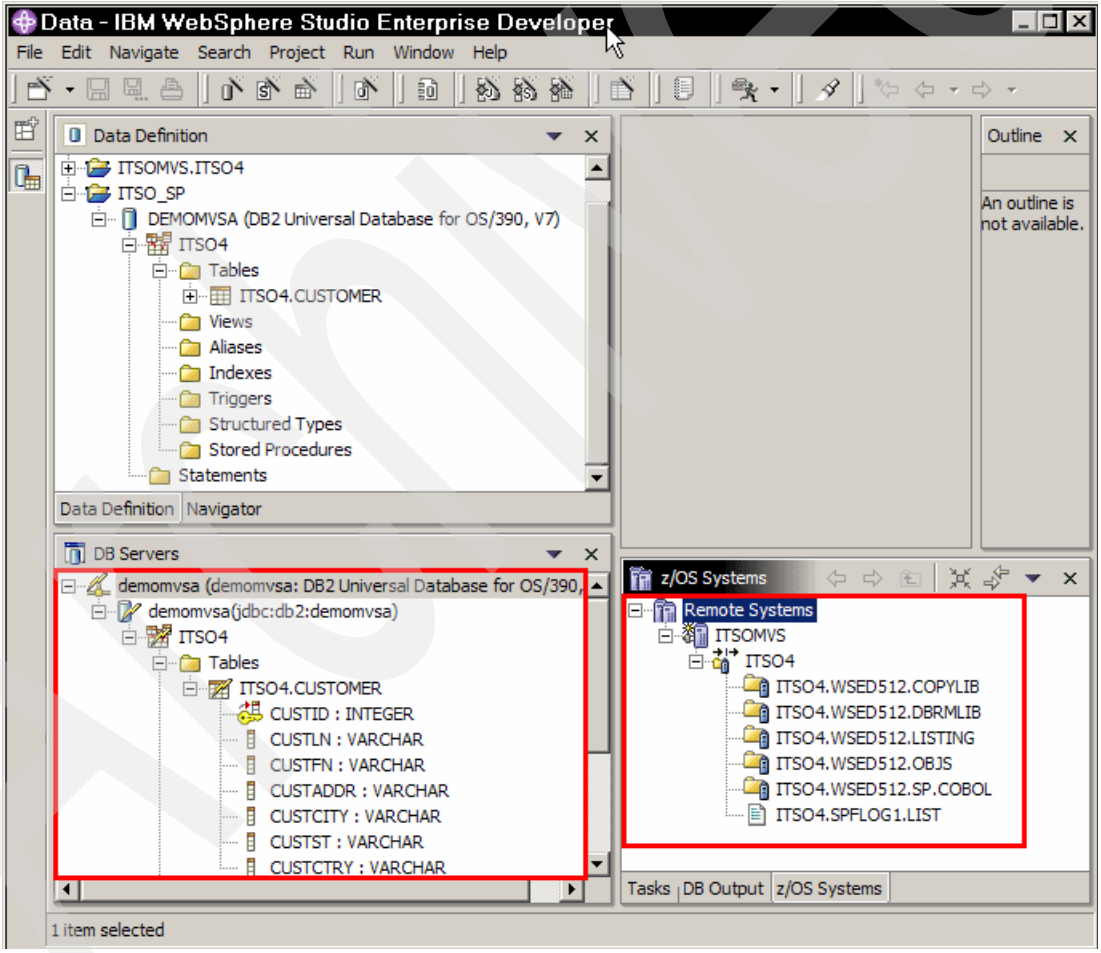


Figure 39-3 Data Perspective with z/OS Systems view

39.2.4 Tips on creating stored procedures

The following techniques will help you to develop more efficient and useful stored procedures:

- Pass all input data and parameters from the client application to the stored procedure at invocation time. You can also pass table names as input data, which can provide additional parameter input values.
- Return all result data and output parameters to the client application only when the stored procedure is complete. For example, a client application might accumulate a collection of database updates and pass them to a stored procedure to be applied as a batch, with a return code that indicates whether the entire batch was applied successfully or rolled back due to a failure.
- Define input and output parameters in both the client application and the stored procedure. The stored procedure must expect the passed parameters. A stored procedure is not required to receive input parameters or return output parameters. Also, a stored procedure is not required to return a result set, and may return one or more result sets.

For our example, we invoked the stored procedure that can have two possible outcomes:

1. If ACTION is INS, we insert the data in the table and execute a query of the inserted data, returning the SQLCODE in the RESPONSE parameter.
2. If the ACTION is SEL, we do a select all and return the results of the query in the RESPONSE parameter.

We used COBOL as the stored procedure language, but if we used PL/I all the dialogs would be the same. Since we needed multiple SQL statements we created the stored procedures and selected Generate multiple SQL statements as described in the next section.

39.2.5 Create a COBOL stored procedure

1. Connect to the database and to z/OS, and open the Data Definition view. Right-click the **Stored Procedures** folder and select **New** → **COBOL Stored Procedure** from the context menu. Note that a PL/I stored procedure could be created instead, as shown in Figure 39-4.

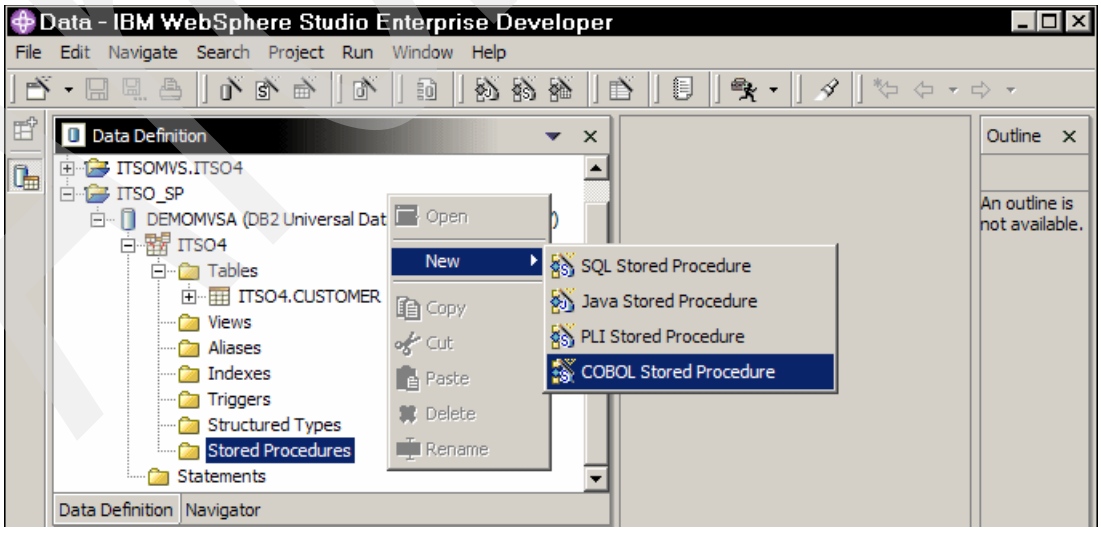


Figure 39-4 Creating a COBOL Store Procedure

2. In the New COBOL Stored Procedure window, type ITSOSP2 in Name and click **Next**.

3. In the New COBOL Stored Procedure window click **Browse for Connected system** and select **ITSOMVS** (or whatever connection name you used in the z/OS Systems view) and click **OK**. This is the z/OS system to which the stored procedure will be written.
4. Click **Browse for Data set name level** and select your data set name level, like ITSO4.* and click **OK.**
5. Click **Browse for Source data set** and select the data set that you allocated before to hold the generated COBOL code, like ITSO4.WSED512.SP.COBOL. This is the library where your new stored procedure will be written.
6. When the Source Location dialog box appears, click **Next**.
7. When the Definition dialog box appears, click **Change**.
8. When the SQL Statement dialog box appears, select **Generate no SQL statement** and click **OK**.

> **Note:** If you prefer you could use the SQL Assist dialog, but you must remember that the COBOL code generated is just a skeleton and will need to be modified for correct compiling. In our example we do not use SQL Assist, but rather inserted the code necessary in the generated COBOL skeleton.

9. In the Definitions select **One as Result set** and **SQL Exception** as shown in Figure 39-5 and click **Next.**

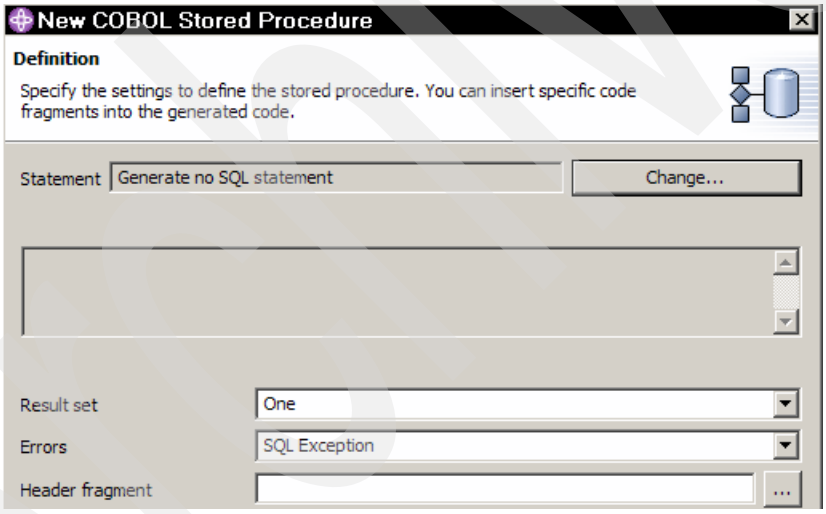


Figure 39-5 Specifying the stored procedure result set

10. Specify the parameters for your stored procedure by clicking **Add**.
11. Specify the characteristics of the parameter and click **Apply** for each parameter; when finished click **OK**. To implement our example, define the parameters for the stored procedures listed in Table 39-2.

Table 39-2 Parameters to be added in the stored procedure

Parameter mode	Name	SQL type	Length
In	ACTION	CHAR	3
In	CUSTID	INTEGER	
In	CUSTLN	VARCHAR	32

Parameter mode	Name	SQL type	Length
In	CUSTFN	VARCHAR	32
In	CUSTADDR	VARCHAR	32
In	CUSTCITY	VARCHAR	32
In	CUSTST	VARCHAR	5
In	CUSTCTRY	VARCHAR	32
Out	RESPONSE	VARCHAR	120

Tip: Do not enter invalid COBOL names for the parameters. For example, CUST_ID would be accepted when creating the stored procedures parameters, but when the COBOL is generated those data names will be invalid, since CUST_ID is an invalid COBOL name. Also do not use invalid DB2 names like CUST-ID. Our suggestion is not to use any special characters as parameter names.

Tip: Be careful when creating the parameters. Double check that the sizes and types are correct and that you have RESPONSE as an output parameter.

12. When the New COBOL Stored Procedure dialog box appears, click **Next**. Type the Collection ID into the entry field or select one from the list by clicking the "**...**" button. See the *DB2 Application Programming and SQL Guide* for more information about the Collection ID. In our example the collection ID is WSEDSP.
13. Click **Advanced**. The Collection ID is used for binding the package of the stored procedure. In the z/OS Options window:
    a. Type your WLM environment, like DSNAWLM, where the stored procedure will run. The stored procedure will run in the DB2 established stored procedure address space if you do not enter a WLM environment name.
    b. Select **DB2** for External security. This option specifies how the stored procedure interacts with an external security product, such as RACF, to control access to non-SQL resources.
    c. Figure 39-6 shows our example. Use the defaults for Build Options. Click **OK** to close this window and click **Next** to continue.

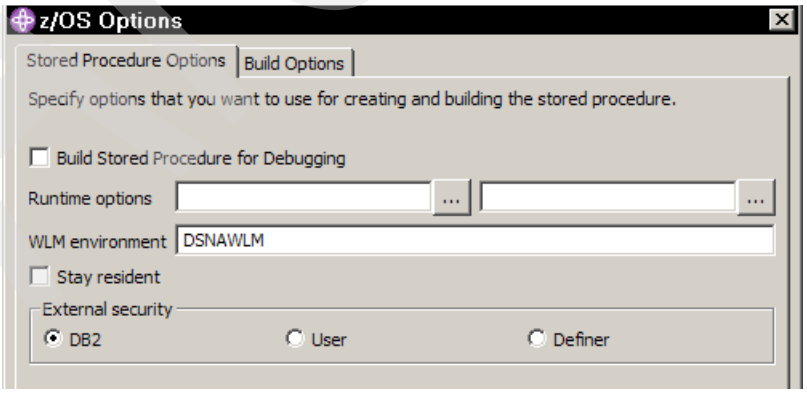


Figure 39-6 WLM environment

14. .The Summary screen shows you the settings that you have specified. You can go back and modify things if necessary. Although the wizard is "one way" only, after the generation

some parameters like the DDL and others can be changed. To see the SQL created, just click **Show SQL.** Click **Close** to continue the generation.

15. On the Summary page of New COBOL Stored Procedure window, click **Finish**. The code is generated. The generated stored procedure source code appears in the Editor view as seen in Figure 39-7. You must be connected to the z/OS since the code will be generated at the z/OS.

.

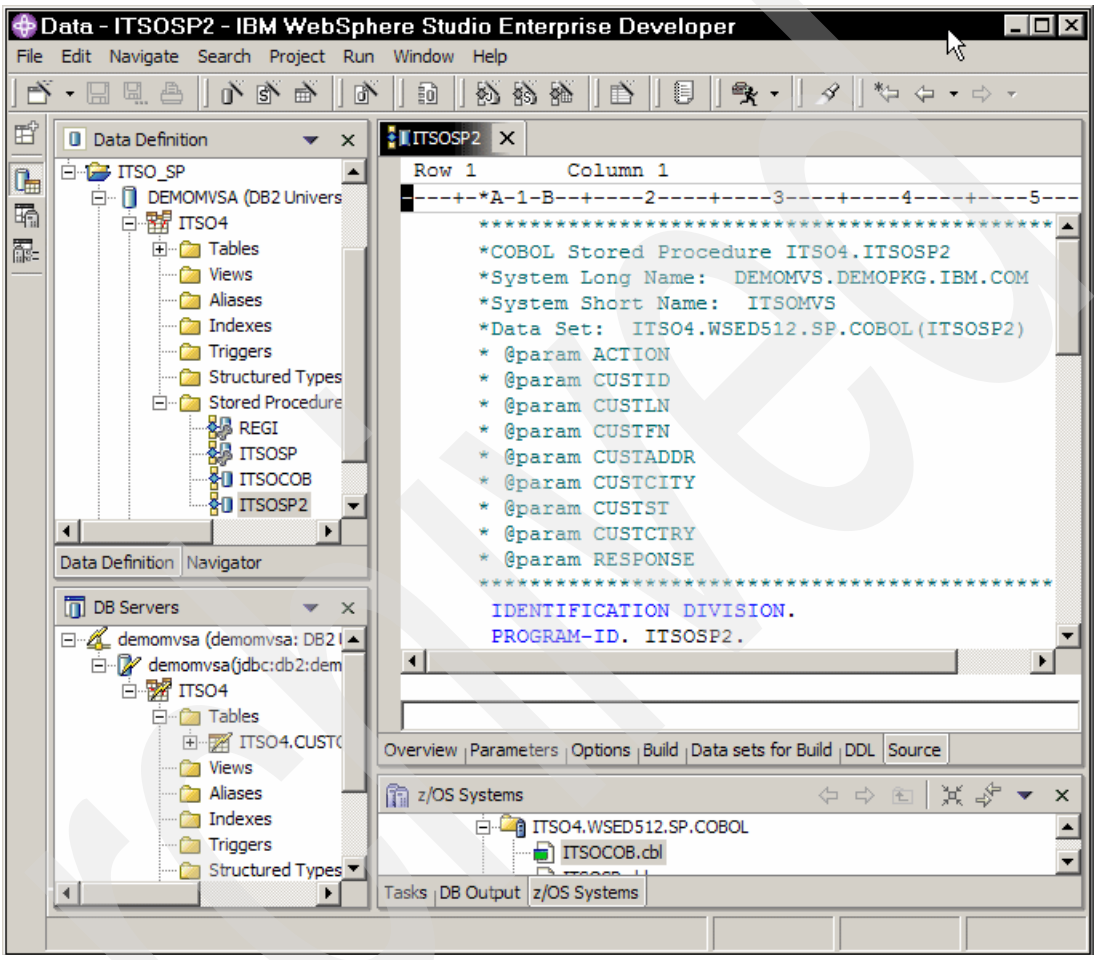


Figure 39-7 COBOL stored procedure generated

16. In the Editor View, click the various tabs: Overview, Parameters, Options, Build, Data sets for Build, and DDL to be sure that all entries are correct for your environment. The Options panel is shown in Figure 39-8.

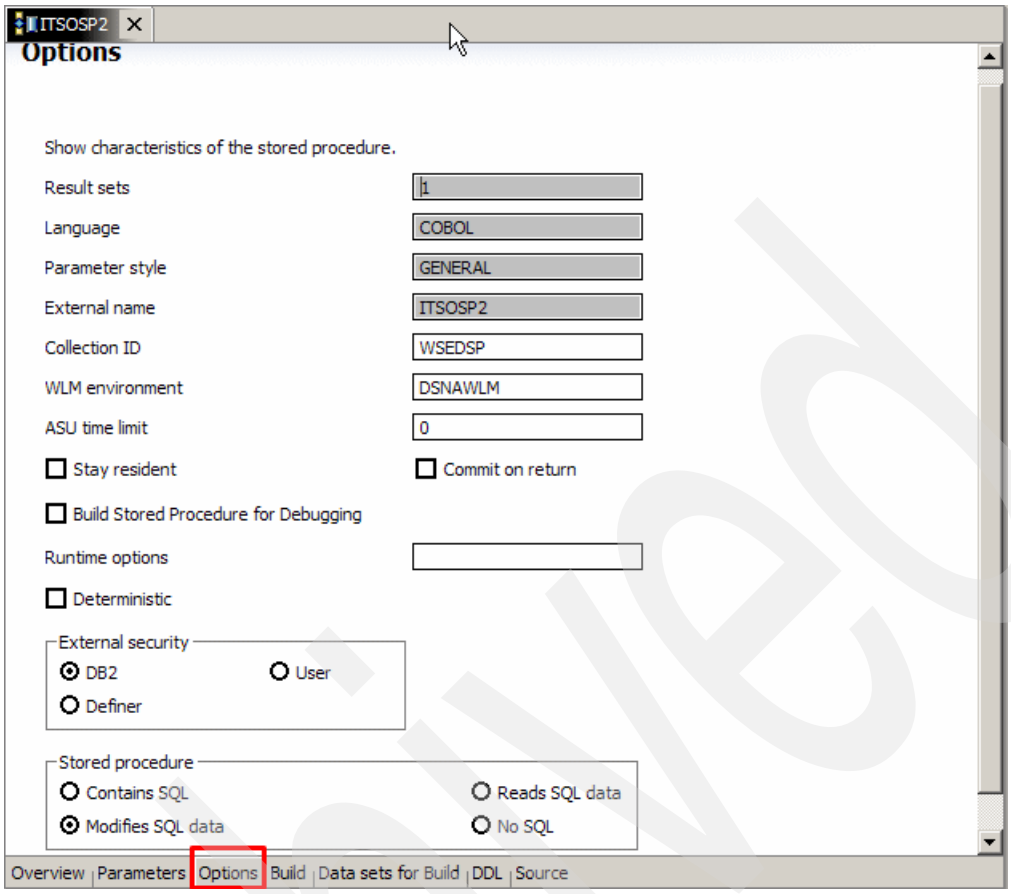


Figure 39-8 Stored procedure Build options

Some of the options you can change here are:

ASU time limit: Specifies the total amount of processor time, in CPU service units, that a single invocation of a stored procedure can run.

Stay resident: Specifies whether the stored procedure load module is to remain resident in memory when the stored procedure ends.

Commit on return: Indicates whether DB2 commits the transaction immediately on return from the stored procedure.

Build Stored Procedure for Debugging: Select this check box to generate the runtime options for debugging

Runtime options: Language Environment runtime options to control certain aspects of the program processing.

Deterministic: Specifies whether the stored procedure returns the same result from successive calls with identical input arguments.

External security: Specifies how the stored procedure interacts with an external security product, such as RACF, to control access to non-SQL resources.

Stored procedure: Indicates whether the stored procedure can execute any SQL statements and, if so, what type. Use the NO SQL option if your stored procedure does not contain SQL statements. The build utility will invoke the SQL statement coprocessor and try to bind a package unless the NO SQL option is set.

See the DB2 SQL Reference for more information about these options.

39.2.6 Adding program logic and SQL statements to the source code

At this point you have created a COBOL skeleton code in the z/OS system and you have to modify it according the business rules.

For our example the business rules are very simple and we need to make changes according to these specifications:

- If the stored procedure is invoked with **INS** in the parameter ACTION, the program will insert the data in the table and execute a query of the inserted data, returning the results in the RESPONSE parameter. The NAMES parameter will have Lastname, Firstname, and the COBOL logic must insert Lastname in the DB2 column CUSTLN and Firstname in the column CUSTFN.
- If the ACTION is **SEL**, the program will do a select all and return the query result in the RESPONSE parameter.

The COBOL changes can be done in the Editor area. If you are familiar with ISPF editor, select the option **Window** → **Preferences** → **LPEX Editor** → **ispf**. This could make your task easier.

Note that the COBOL skeleton generated has all the parameters, the linkage section, and the procedure division created according to your input in the previous dialogs. This skeleton is shown in Example 39-3.

Example 39-3 COBOL skeleton generated from the wizard

```
**************************************************
*COBOL Stored Procedure ITSO4.ITSOSP2
*System Long Name:  DEMOMVS.DEMOPKG.IBM.COM
*System Short Name:  ITSOMVS
*Data Set:  ITSO4.WSED512.SP.COBOL(ITSOSP2)
* @param ACTION
* @param CUSTID
* @param CUSTLN
* @param CUSTFN
* @param CUSTADDR
* @param CUSTCITY
* @param CUSTST
* @param CUSTCTRY
* @param RESPONSE
**************************************************
IDENTIFICATION DIVISION.
PROGRAM-ID. ITSOSP2.
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.
01 NAM PIC X(18) VALUE 'ITSOSP2'.
01 SCHE PIC X(8).
      EXEC SQL INCLUDE SQLCA END-EXEC.
LINKAGE SECTION.
01 ACTION PIC X(3).
01 CUSTID PIC S9(9) COMP-5.
01 CUSTLN.
      49 VAR-LEN PIC S9(4) USAGE BINARY.
      49 VAR-TEXT PIC X9(32) USAGE DISPLAY.
01 CUSTFN.
      49 VAR-LEN PIC S9(4) USAGE BINARY.
```

```
      49 VAR-TEXT PIC X9(32) USAGE DISPLAY.
01 CUSTADDR.
      49 VAR-LEN PIC S9(4) USAGE BINARY.
      49 VAR-TEXT PIC X9(32) USAGE DISPLAY.
01 CUSTCITY.
      49 VAR-LEN PIC S9(4) USAGE BINARY.
      49 VAR-TEXT PIC X9(32) USAGE DISPLAY.
01 CUSTST.
      49 VAR-LEN PIC S9(4) USAGE BINARY.
      49 VAR-TEXT PIC X9(5) USAGE DISPLAY.
01 CUSTCTRY.
      49 VAR-LEN PIC S9(4) USAGE BINARY.
      49 VAR-TEXT PIC X9(32) USAGE DISPLAY.
01 RESPONSE.
      49 VAR-LEN PIC S9(4) USAGE BINARY.
      49 VAR-TEXT PIC X9(120) USAGE DISPLAY.
PROCEDURE DIVISION USING
      ACTION
      CUSTID
      CUSTLN
      CUSTFN
      CUSTADDR
      CUSTCITY
      CUSTST
      CUSTCTRY
      RESPONSE.
      GOBACK.
```

You will need to perform the following changes:

1. Add a data name W-SQLCODE in the working storage to hold the DB2 SQL code. The working-storage will be like Example 39-4.

   Example 39-4 Modified working storage

   ```
   WORKING-STORAGE SECTION.
    01 NAM PIC X(18) VALUE 'ITSOSP2'.
    01 SCHE PIC X(8).
    01 W-SQLCODE PIC S9(3).
        EXEC SQL INCLUDE SQLCA END-EXEC.
   ```

2. Add the code required to perform the business logic in the procedure division. The program will check the contents of the parameter ACTION.
   a. If ACTION is SEL the program will do a SELECT. This select must be modified to do a WHERE clause.
      - If the SQLCODE is 0 a message is sent indicating successful operation, as well as the SQLCODE.
      - If the SQLCODE is not 0 an unsuccessful message must be sent to the parameter RESPONSE.
   b. If ACTION is INS the program checks if First Name and Last Name are not empty.
      - If any or both fields are empty an unsuccessful message is sent.
      - If first name and last name fields are not empty the data is inserted to the table and a successful message is sent to RESPONSE.
   c. If ACTION is none of the above a message is sent to RESPONSE.

3. The Procedure Division must have the content shown in Example 39-5. You can change the table qualifier in the source code to the name of your schema and save the changes.

Example 39-5 Modified Procedure Division

```
          PROCEDURE DIVISION USING
           ACTION
           CUSTID
           CUSTLN
           CUSTFN
           CUSTADDR
           CUSTCITY
           CUSTST
           CUSTCTRY
           RESPONSE.
           Evaluate ACTION
      * -------------------------------------------------------------
           When 'SEL'
                 EXEC SQL
                      DECLARE C1 CURSOR WITH RETURN WITH HOLD FOR
                      SELECT * FROM ITSO4.CUSTOMER
                       WHERE CUSTID >= :CUSTID
                  END-EXEC
                  EXEC SQL
                      OPEN C1
                  END-EXEC
                If SQLCODE = 0 then
                  String 'Select successful'
                  delimited by Size into VAR-TEXT of Response
                Else
                  Move SQLCODE to W-SQLCODE
                  String 'Select unsuccessful: Sqlcode ' W-SQLCODE
                  delimited by Size into VAR-TEXT of Response
                End-If
      * -------------------------------------------------------------
           When 'INS'
            Evaluate true also true
              When VAR-LEN of CUSTLN  = 0 also
                       VAR-LEN of CUSTFN  = 0
                  String 'Insert unsuccessful: '
                  'First Name and Last Name not specified'
                  delimited by Size into VAR-TEXT of Response
                   GOBACK
              When VAR-LEN of CUSTLN = 0 also VAR-LEN of CUSTFN > 0
                  String 'Insert unsuccessful:Last Name not specified'
                  delimited by Size into VAR-TEXT of Response
                  When VAR-LEN of CUSTLN > 0 also
                        VAR-LEN of CUSTFN = 0
                  String 'Insert unsuccessful:First Name not specified'
                      delimited by Size into VAR-TEXT of Response
                   GOBACK
                When Other
                EXEC SQL INSERT INTO ITSO4.CUSTOMER
               (CUSTID,CUSTLN,CUSTFN,CUSTADDR,CUSTCITY,CUSTST,CUSTCTRY)
               VALUES
               (:CUSTid,:CUSTln,:CUSTfn,:CUSTaddr,:CUSTcity,:CUSTst,
```

```
            :CUSTctry )
            END-EXEC
         End-Evaluate
          If Sqlcode = 0 then
            String 'Insert successful'
            delimited by Size into VAR-TEXT of Response
          Else
            Move SQLCode to W-SQLCODE
            String 'Insert unsuccessful: Sqlcode ' W-SQLCODE
            delimited by Size into VAR-TEXT of Response
          End-If
* ------------------------------------------------------------
      When Other
         String 'ACTION "'
         ACTION
         '" not supported'
         delimited by Size into VAR-TEXT of Response
      End-Evaluate.
         Inspect VAR-TEXT of Response tallying VAR-LEN of Response
         for characters before initial x'00'
      GOBACK.
```

4. Using the Options tab, select the **Commit on Return** check box for the stored procedure.
5. Any changes on the **Options** page are automatically propagated to the DDL page. Store the changes by pressing Ctrl+S. Figure 39-9 shows the generated DDL page.

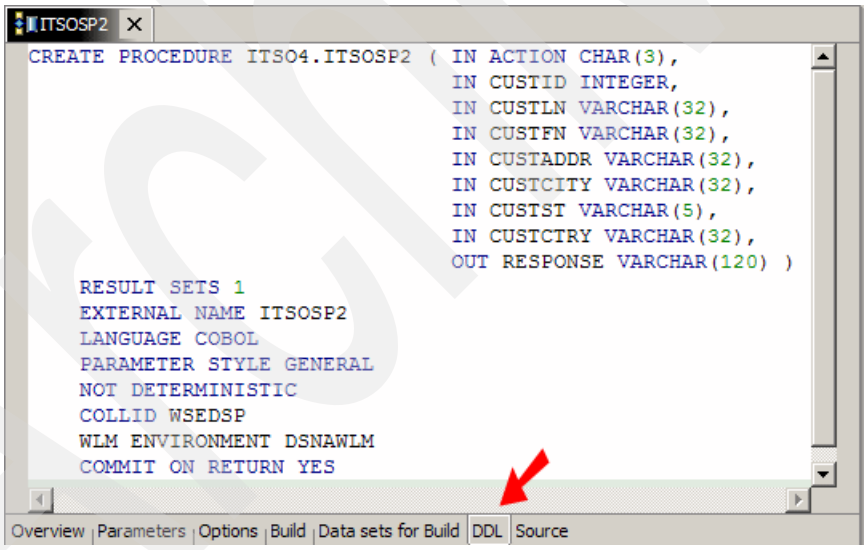


Figure 39-9 Generated DDL

6. Click the **Data sets for Build** tab in the Editor view. Check whether the values are corrected. Some of the options may already have default values, which are obtained from the z/OS Properties. You can change these default property values if you need to. Changing the values in these fields does not affect the z/OS build properties. Changing these values affects only the stored procedure build properties as shown in Figure 39-10.

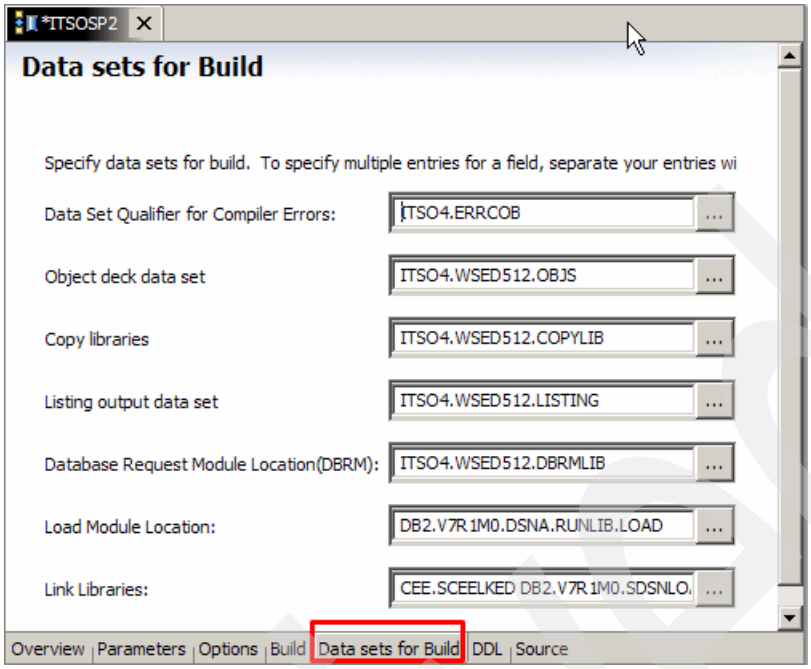


Figure 39-10 Data Sets used for stored procedure build

> **Note:** Multiple entries in a field must be separated by a space. The Dataset Qualifier for Compiler Errors option value is used as the first part of the name of a data set in XML format that will contain any compiler errors. The full data set name will look something like the following: ITSO4.ERRCOB.SPNAME.XXXXXXX.XML
>
> SPNAME is the name of your stored procedure name, and XXXXXXX is a qualifier generated using the timestamp as a basis. An example is: ITSO4.ERRCOB.ITSOSP2.Z421139.XML

7. Store the changes by pressing Ctrl+S. Changing these properties causes the z/OS properties for this particular stored procedure to be updated. If you open this stored procedure for edit later, the editor page for this stored procedure will contain these new properties.

The stored procedure has now been created.

> **Note:** Although the COBOL code that you have created is stored in the z/OS, the stored procedures definitions are stored in your local workspace. That means that if you move from one workspace to another workspace the stored procedures will not be available for modifications. You could export the project using the project interchange capability and then import in the other workspace.

39.3 Building the 390 stored procedure

Before a stored procedure can be invoked from the DB Servers view or any other calling program, it must be built on the remote system.

But you could optionally do a syntax check of your COBOL code, following the steps presented in the next section.

39.3.1 Checking COBOL syntax before doing the building

This step is optional, but could avoid unnecessary compilations at the z/OS.

Before you do the syntax checking you must be sure that your project is prepared for local syntax checking. Perform the following steps to do this:

1. Using the z/OS Systems view, select the program code generated in the z/OS and check its properties. Select the file, and using the context menu select **Properties**. In our example we used the **Local Compile Options** as follows:

   `SQL('DATABASE DEMOMVSA USER ITSO4 USING mypassword')` as Compile Options

   `C:\SQLLIB\include\cobol_a` as SYSLIB parameter

   Figure 39-11 shows the properties. Do not forget the single quote character (') inside of the parenthesis. Note that the database used is on the z/OS and DEMOMVSA is the alias connection.

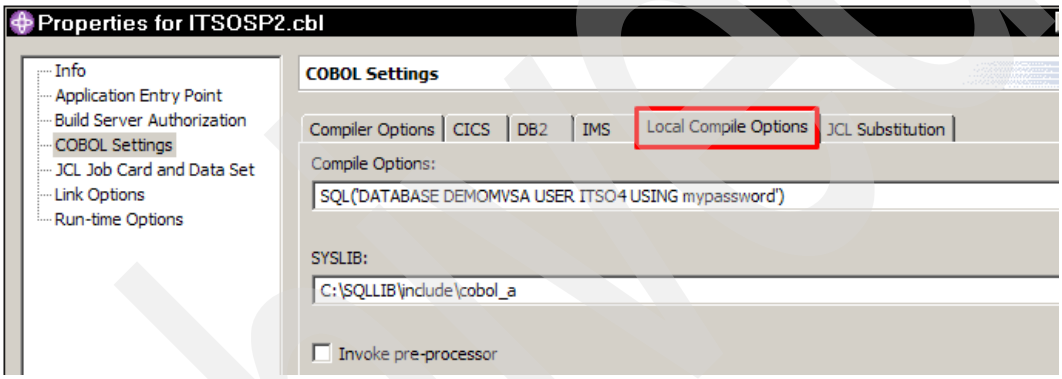


Figure 39-11 Local Compile Options for syntax checking

2. Still using the Data perspective, click the **Source** tab to see the COBOL program. If you have closed the program, you can open it again using the context editor from the ITSOSP2 Stored Procedure and selecting the option Open 390 Stored Procedure. Move the cursor to any area in the COBOL source code and using the context editor select **Syntax Check** → **Local** as seen in Figure 39-12. Note that you also could do a Remote syntax checking instead.

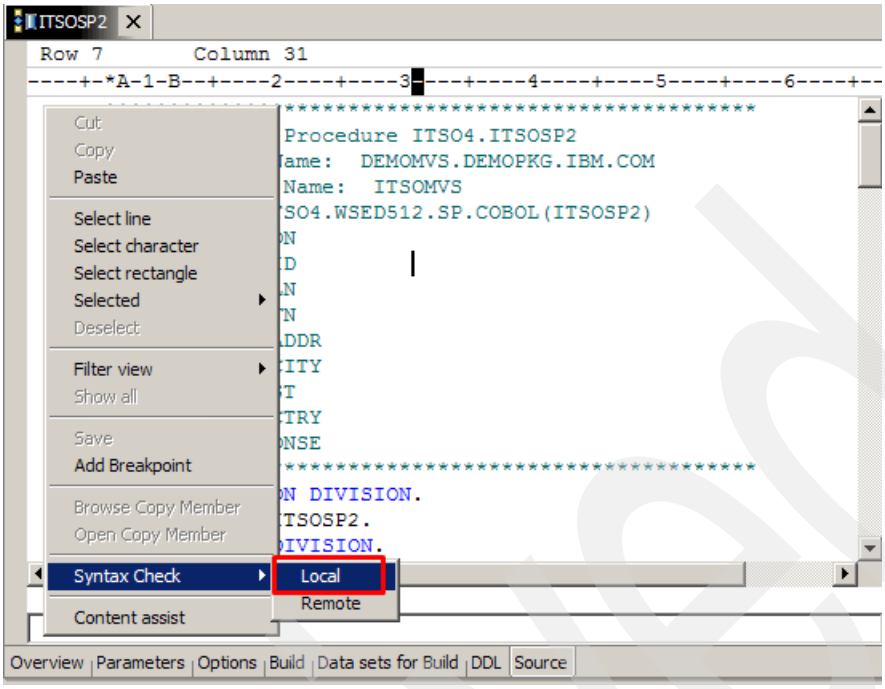


Figure 39-12 Using Local COBOL syntax for the generated COBOL stored procedure

3. Switch to the Tasks view and you will see possible errors or warnings. Figure 39-13 shows the messages that we get for our Remote syntax checking example.

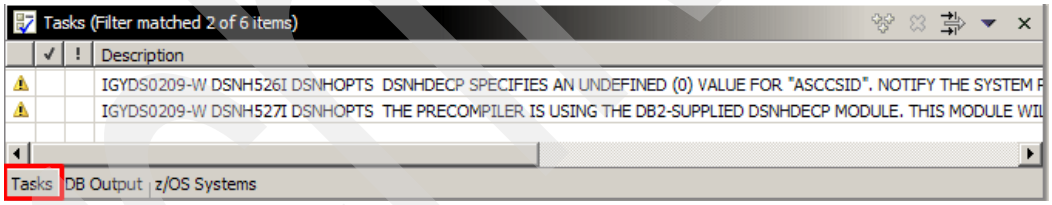


Figure 39-13 Remote Syntax checking results for the generated COBOL stored procedure

You should clear all possible compile errors. This procedure will save you time during the DB2 Stored procedure build that is shown in the next step.

39.3.2 Process for building a DB2 stored procedure

To build the stored procedure, right-click the stored procedure in the Data Definition view and select **Build 390 Stored Procedure**. Figure 39-14 shows our example.

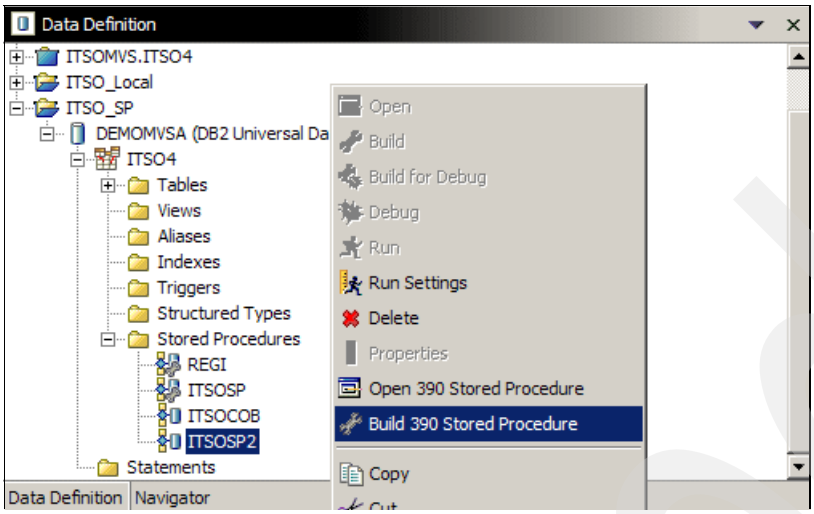


Figure 39-14 Building a Stored Procedure

The build utility is now invoked on the remote system and builds the executable form of the stored procedure. This process includes the compilation of the source code, the linkage of the load module, the binding of the DB/2 package and the registration of the stored procedure in the DB/2 catalog.

The build process is displayed in the DB Output view with the status “In progress” until it is completed. The status in the DB Output view changes to “Success” or “Failure” when the build process is completed.

When execution is successful, a short summary of the build process is displayed. If the build process was not successful, information about the failure will be displayed in the DB Output view. Figure 39-15 shows the successful build of our example.

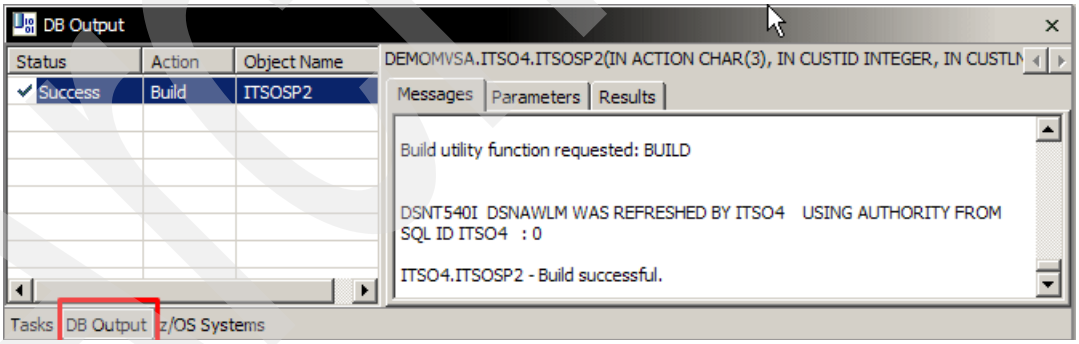


Figure 39-15 Successful stored Procedure build

Once the stored procedure has been built on the remote system and refreshed, it will appear in the DB Servers view. To refresh, right-click the Stored Procedures folder in the DB Servers view and select **Refresh**. Wait until the refresh has been completed. The stored procedure will appear in the Stored Procedures folder of the DB Servers view as shown in Figure 39-16.

:

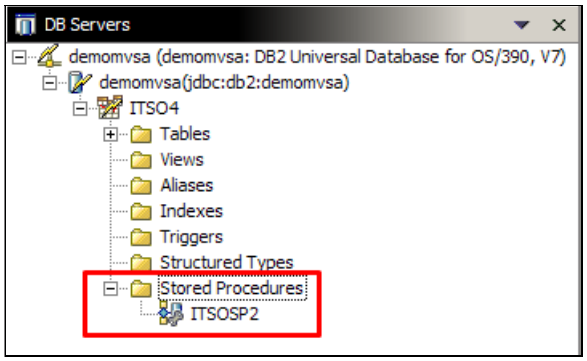


Figure 39-16 Generated stored procedure in the z/OS

At this point you are ready to execute your generated stored procedure.

39.3.3 Checking for compiling and building errors

If you need to check for compilation errors, just go to the z/OS Systems view and edit the generated listing datasets where the compilation is stored. In our example this dataset is ITSO4.WSED512.Listing(ITSOSP2), as shown in Figure 39-17.

:

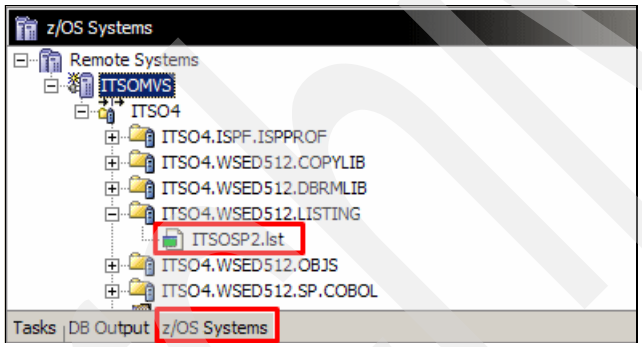


Figure 39-17 Generated compiled listing

Messages and errors that occurred when building the stored procedures are listed in the DB Output view and Messages view window. Be sure that you have the correct data sets allocated; this is a common mistake. Figure 39-18 shows an example where DBRM lib is not allocated.

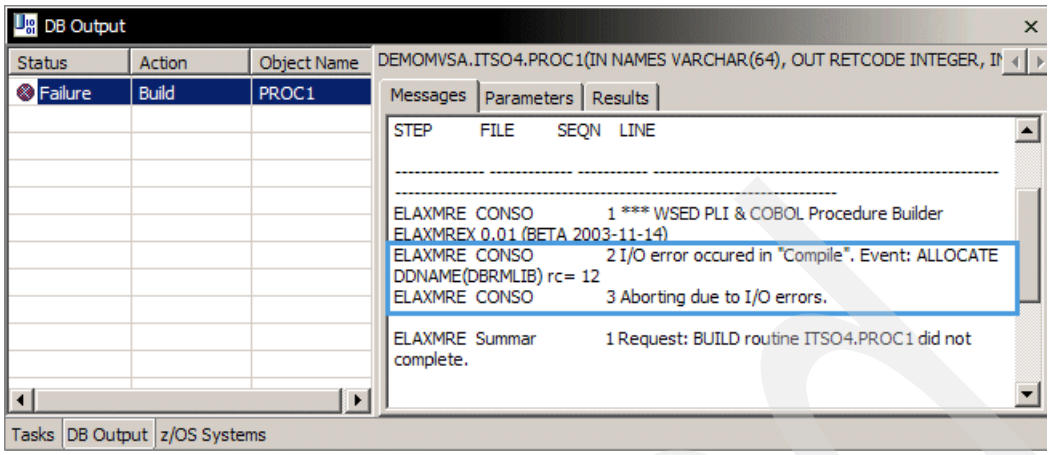


Figure 39-18 Example of error when building the stored procedure

39.4 Running a stored procedure

Once a stored procedure is built on the remote system, it can be invoked from the DB Servers view or the Data Definition view. Right-click the stored procedure and select **Run**.

> **Note:** You can select **Run Settings first** to define variable values that will be remembered for successive executions. With Run Settings you can also define SQL statements to run before and after executing the stored procedure. SQL Assist will help you here to construct SQL statements.

A parameter input panel is displayed if the stored procedure uses input parameters. Specify the parameter values and click **OK**. Figure 39-19 shows our example when doing the **INS** (Insert) as parameter.

.

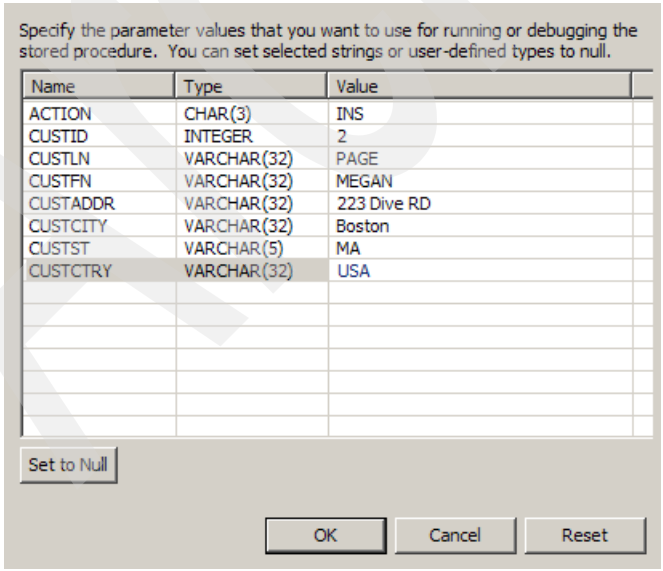


Figure 39-19 Input parameters when running the generated stored procedure

When the run is completed, information about the run will be displayed in the DB Output view.

The Message tab shows a summary of the run process. The Parameter tab shows the Input and Output parameter values of the stored procedure.The Results tab shows the result sets returned by the stored procedure. Figure 39-20 shows our example using INS as parameter.

:

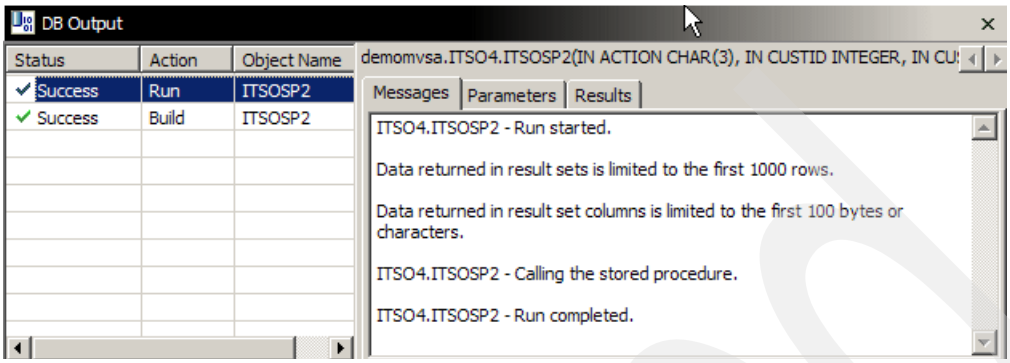


Figure 39-20 Messages when doing running the stored procedure with INS as ACTION

When running with ACTION = SEL and CUSTID = 0 the results shown in the Results tab are as shown in Figure 39-21.

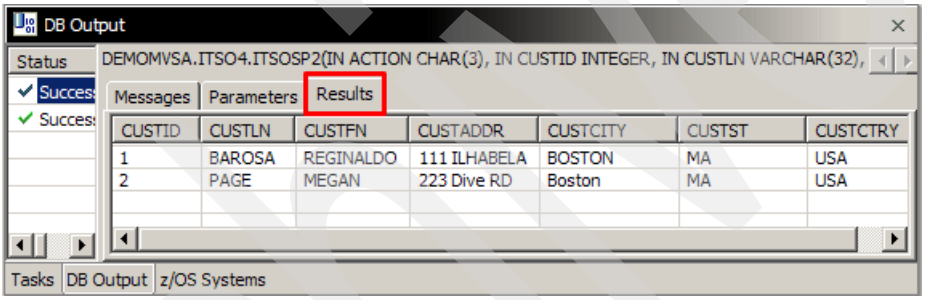


Figure 39-21 Results when running the stored procedure with SEL as ACTION

You also could try running when the ACTION is not correct, or when using the action INS but having empty fields CUSTLN or CUSTFN.

39.5 Removing a stored procedure

To remove a stored procedure from the remote system, it must be deconstruct by the build utility. Right-click the stored procedure in the DB Servers view and select **Drop 390 Stored Procedure** as shown in Figure 39-22.

:

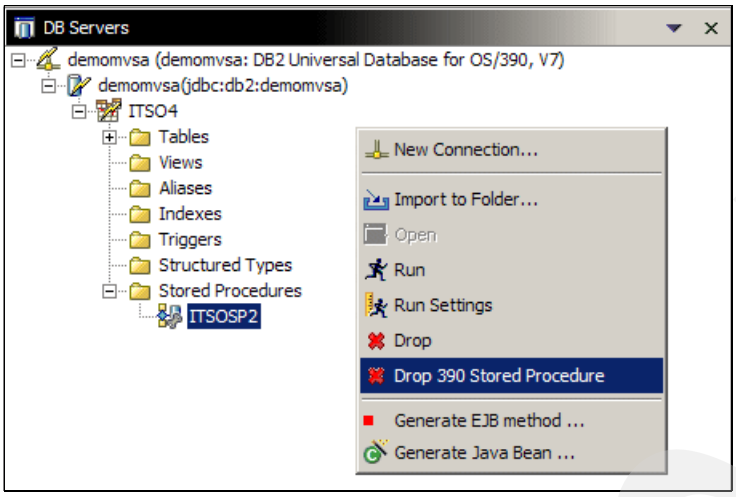


Figure 39-22 Removing a Stored Procedure

Click **Yes** when the dialog box asks if you want to drop your stored procedure. The DB Output view shows whether the stored procedure was successfully removed from the remote system or an error occurred. Figure 39-23 shows a successful remove.

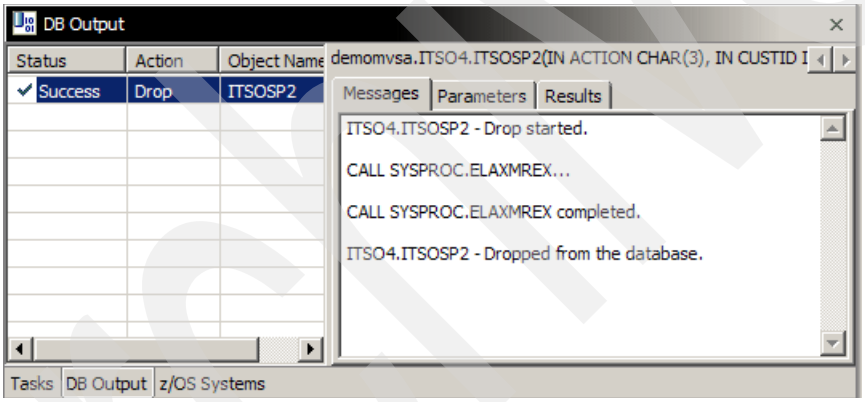


Figure 39-23 Successful stored procedure removal

Note: When dropping a stored procedure, you are only removing its definition in DB2. All your hard work is still there in the WSED workspace and the z/OS data set. After making the necessary changes, you simply need to rebuild it.

39.6 Changing a stored procedure

Currently, the only way to change a stored procedure is to remove it from the remote system, change its definitions in the Data Definition view, and build it again.

39.7 Debugging a stored procedure

To debug a stored procedure, it must be built with specific compile and runtime options. A compile listing is also required for COBOL stored procedures.

Perform the following steps to debug our example:

1. Remove the stored procedure from the remote system. From the DB Servers view, select the stored procedure and using the context menu select **Drop 390 Stored Procedure**.
2. Specify a compile option TEST and add the necessary debug information (IP address). From the Data Definition view, select the stored procedure to be modified and using the context menu select **Open 390 Stored Procedure**.
   a. On the Options tab in the Editor view select the **Build Stored Procedure for Debugging** check box. This will fill in the Runtime options field with the required parameters. Figure 39-24 shows our example. Note that the IP address could change dynamically and you will need to do this again when moving to another IP address.

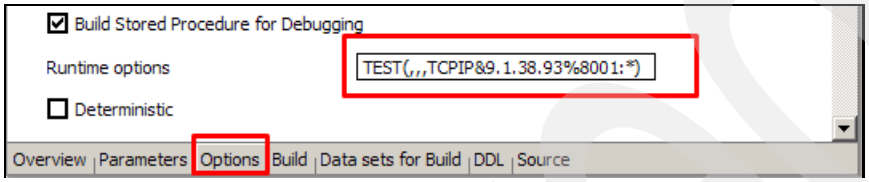


Figure 39-24 Selecting Build Stored Procedure for Debugging

   b. On the Build tab in the Editor view type TEST in the Compile options field as shown in Figure 39-25.

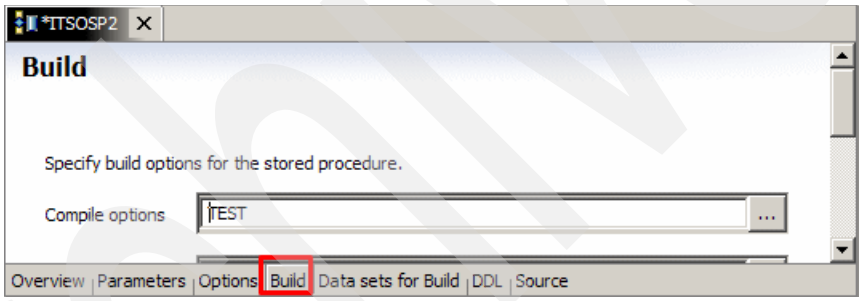


Figure 39-25 Specifying Test option for Compile the stored procedure

   c. Store the changes by pressing Ctrl+S.
3. The generated run-time options from the Options page are propagated to the DDL page. Click the DDL tab in the Editor view and store the modified CREATE PROCEDURE as shown in Figure 39-26.

```
CREATE PROCEDURE ITSO4.ITSOSP2 ( IN ACTION CHAR(3),
                                 IN CUSTID INTEGER,
                                 IN CUSTLN VARCHAR(32),
                                 IN CUSTFN VARCHAR(32),
                                 IN CUSTADDR VARCHAR(32),
                                 IN CUSTCITY VARCHAR(32),
                                 IN CUSTST VARCHAR(5),
                                 IN CUSTCTRY VARCHAR(32),
                                 OUT RESPONSE VARCHAR(120) )
    RESULT SETS 1
    EXTERNAL NAME ITSOSP2
    LANGUAGE COBOL
    PARAMETER STYLE GENERAL
    NOT DETERMINISTIC
    COLLID WSEDSP
    WLM ENVIRONMENT DSNAWLM
    RUN OPTIONS 'TEST(,,,TCPIP&9.1.38.93%8001:*)'     COMMIT ON RETURN YES
```

Overview | Parameters | Options | Build | Data sets for Build | DDL | Source

Figure 39-26 DDL generated when debug in effect

4. Build the stored procedure as described previously.
5. Run the stored procedure as described previously.
   a. Enter a wrong ACTION like xxx and 0 as CUSTID and click **OK**.
   b. The Debugging perspective opens automatically when the stored procedure has been invoked. Click **OK** if the following message appears:

      `EQA2383I The environment is not yet fully initialized. Use Step or Run.`

   c. Using the COBOL Editor view, scroll down the program and set a breakpoint by double-clicking the gray vertical bar on the right side of the Source view as shown in Figure 39-27.

```
JITSOSP2   ITSO4.WSED512.LISTING(ITSOSP2)
Row 58     Column 1                                  Browse
000058   58          RESPONSE.
000059   59          Evaluate ACTION
000060   60      * ----------------------------------------
000061   61          When 'SEL'
000062   62               EXEC SQL
000063   63                    DECLARE C1 CURSOR WITH RETUR
000064   64                    SELECT * FROM ITSO4.CUSTOMER
000065   65                     WHERE CUSTID >= :CUSTID
000066   66                 END-EXEC
000067   67                 EXEC SQL
000068   68                    OPEN C1
000069   69                 END-EXEC
000070   70               If SQLCODE = 0 then
```

Figure 39-27 Setting breakpoints

   d. Click the **Resume** button on the window bar of the Debug view to run from one breakpoint to the next.
   e. Click the **Step Into** button on the window bar of the Debug view to step from one statement to the next. Note that since xxx is an invalid action a message will be sent and the stored procedure will be ended. But we don't want that.

f. From the Variables view, change the contents of the ACTION by double-clicking it. Change to 'SEL' and click OK as shown in Figure 39-28.

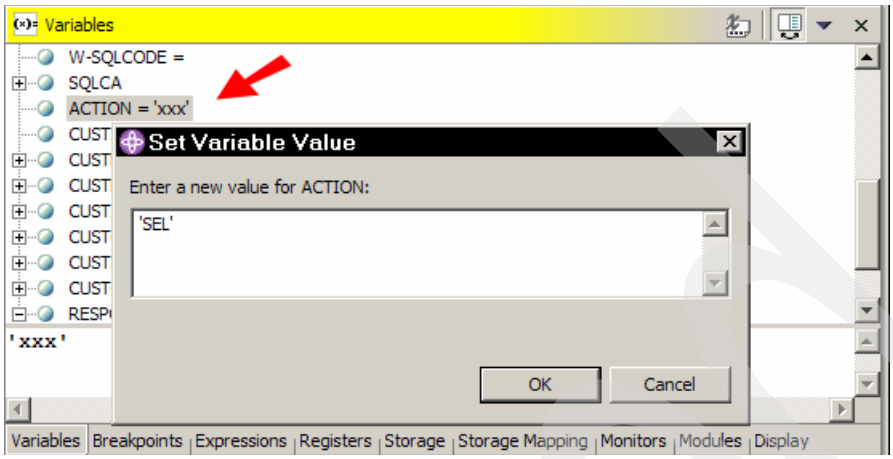


Figure 39-28 Changing variable contents during the debug

g. We need to execute again the Evaluate statement. Using the COBOL editor, browse back tot where the evaluate statement is defined and using the context menu, select the command **Jump To Location**, as shown in Figure 39-29.

:

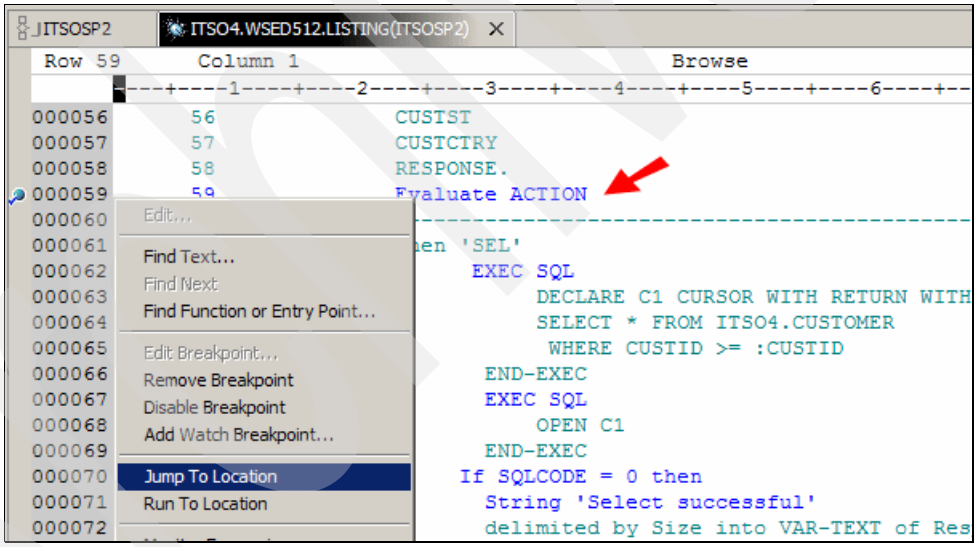


Figure 39-29 Jump To Location when debugging

h. Click **Step Over** and note that the program will behave differently now.

i. Note that moving the cursor to some fields in the COBOL program we can see their contents. Figure 39-30 shows our example where we are inspecting the value of the field SQLCODE.

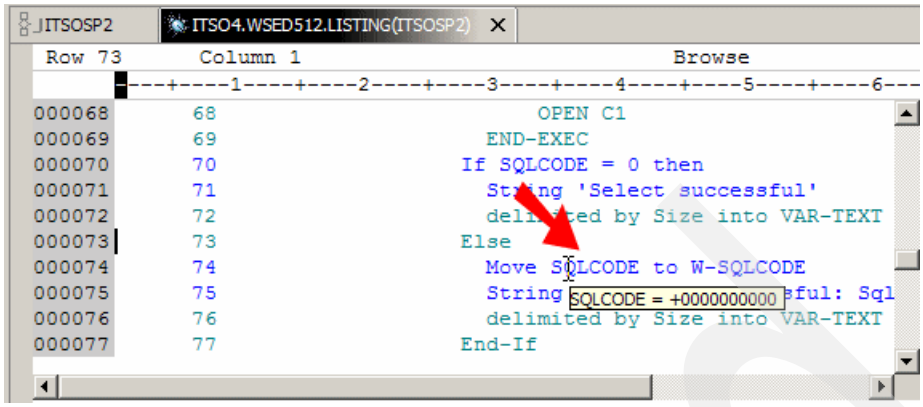


Figure 39-30 View field contents

j. Click **OK** when the Debugger Message window says the Application has Terminated.

k. Click the Data Perspective button to switch back to the Data Perspective as shown in Figure 39-31.

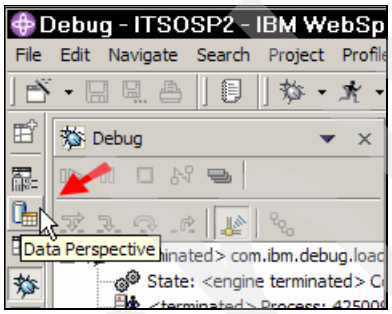


Figure 39-31 Switching to Data perspective

l. The results should be like the Results page shown in Figure 39-32.

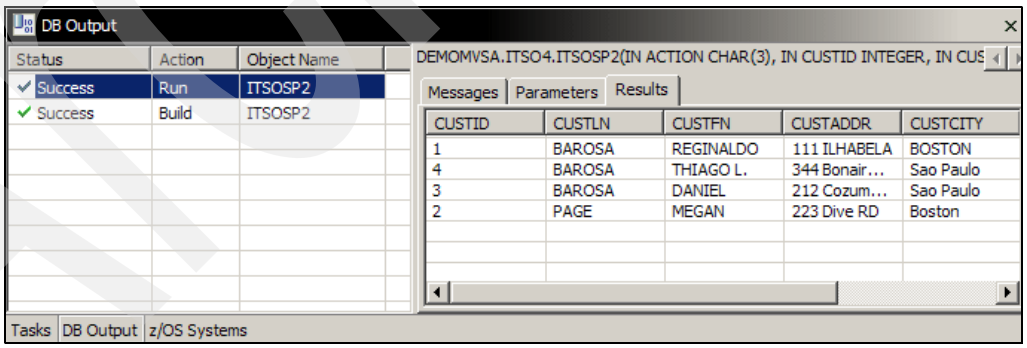


Figure 39-32 SEL action when running with debug

39.8 Current stored procedures SQL assist wizard limitations

The objective of WSED 5.1.2 is to generate a COBOL or PL/I skeleton. Some of the dialogs for creating SQL statements are available at the SQL assist.

For example, if you do a SQL assist using a SELECT, the statement generated will be shown in Figure 39-33. You will be able to run the query against the DB2 table.

:

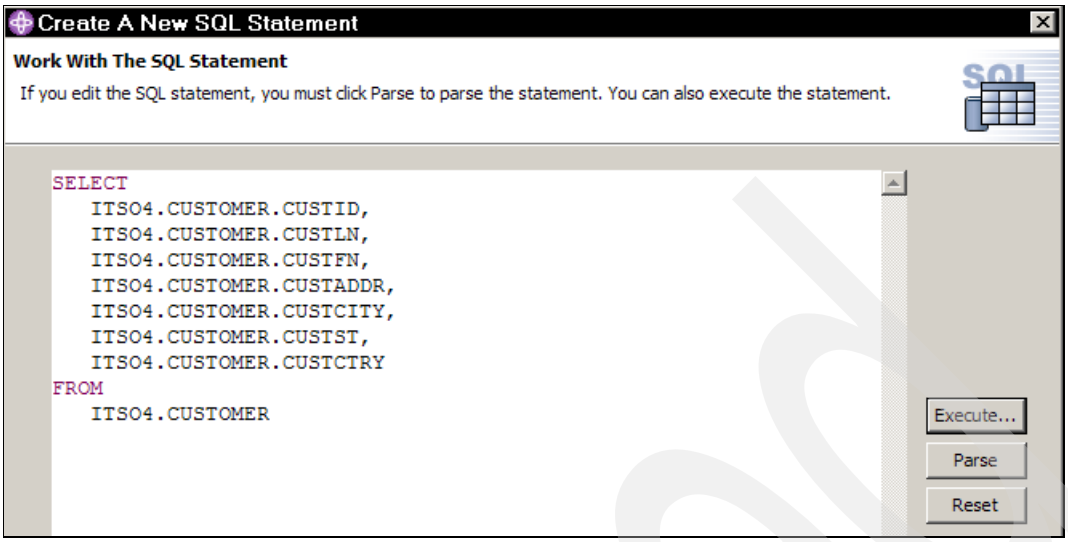


Figure 39-33 SQL statements generated using the SQL Assist wizard

Figure 39-34 shows the COBOL code generated. Notice that this code is not complete since it needs to be updated for correct compilation.

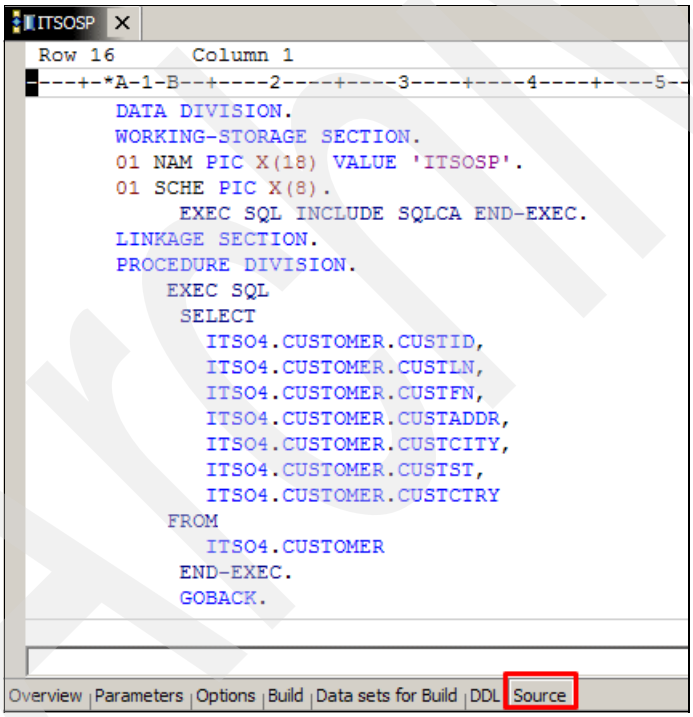


Figure 39-34 COBOL code generated using the wizard.

In order to compile and run this program, some code must be added as shown in Figure 39-35.

```
ITSOSP
Row 33        Column 12
----+-*A-1-B--+----2----+----3----+----4----+----5----+-
            EXEC SQL INCLUDE SQLCA END-EXEC.
       LINKAGE SECTION.
       PROCEDURE DIVISION.
           EXEC SQL
            DECLARE C1 CURSOR WITH RETURN WITH HOLD FOR
            SELECT
              ITSO4.CUSTOMER.CUSTID,
              ITSO4.CUSTOMER.CUSTLN,
              ITSO4.CUSTOMER.CUSTFN,
              ITSO4.CUSTOMER.CUSTADDR,
              ITSO4.CUSTOMER.CUSTCITY,
              ITSO4.CUSTOMER.CUSTST,
              ITSO4.CUSTOMER.CUSTCTRY
            FROM
              ITSO4.CUSTOMER
           END-EXEC.
           EXEC SQL
             OPEN C1
           END-EXEC
           GOBACK.
Overview | Parameters | Options | Build | Data sets for Build | DDL | Source
```

Figure 39-35 Modified COBOL code after wizard generation

Part 10

Appendixes

A

Additional material

This redbook refers to additional material that can be downloaded from the Internet as described below.

Locating the Web material

The Web material associated with this redbook is available in soft copy on the Internet from the IBM Redbooks Web server. Point your Web browser to:

ftp://www.redbooks.ibm.com/redbooks/SG246492

Alternatively, you can go to the IBM Redbooks Web site at:

ibm.com/redbooks

Select the **Additional materials** and open the directory that corresponds with the redbook form number, SG246492.

Using the Web material

The additional Web material that accompanies this redbook includes the following file:

File name	Description
SG246492.zip	Zipped code samples

The following system configuration is recommended:

Hard disk space:	4 MB for the downloaded zip file and unpacked files
Operating System:	Windows 2000/XP
Processor:	Pentium
Memory:	128 MB

System requirements for downloading the Web material

The following system configuration is recommended:

Hard disk space:	4 MB for the downloaded zip file and unpacked files
Operating System:	Windows 2000/XP
Processor:	Pentium
Memory:	128 MB

How to use the Web material

Create a subdirectory (folder) on your workstation, and unzip the contents of the Web material zip file into this folder.

The extracted files are all in binary format. They are the output of the TSO TRANSMIT command.

Use your mainframe file transfer protocol to upload the binary files. You must use the following attributes: FB, LRECL=80, BLKSIZE=3120.

After each file is uploaded, issue the following command from the TSO READY prompt:

RECEIVE INDA(*xxxx*)

In this command, *xxxx* is the name of the file.

You will receive the following messages, if you issue the command against the **source** file as shown in Example A-1.

Example: A-1 Receive INDA(xxxx) messages

```
INMR901I Dataset DAVINR1.REDBOOK.PDPAK.SOURCE from DAVINR1 on NODENAME
INMR906A Enter restore parameters or 'DELETE' or 'END' +
```

You can to reply as shown in Example A-2.

Example: A-2 Receive INDA(xxxx) reply

```
indsn(pdtools.redbook.source)
```

Refer to Chapter 30, “Introduction to the scenarios” on page 807 for detailed information on the scenarios.

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

IBM Redbooks

For information on ordering these publications, see “How to get IBM Redbooks” on page 999. Note that some of the documents referenced here may be available in softcopy only.

- *Introduction to the IBM Problem Determination Tools,* SG24-6296
- *Introduction to the IBM Application Development Tools on z/OS and OS/390,* SG24-6887
- *Using IBM Application Development Tools for z/OS and OS/390*

Other publications

OS PL/I Version 2 Programming: Using PLITEST, 13.2

Coverage Utility User's Guide and Messages V5R1, SC18-9303

These publications are also relevant as further information sources:

- **Application Monitor Version 2, Release 1:**
  - User's Guide, SC18-9084-01
  - Customization Guide, SC18-9085-01
  - Program Directory, GI10-8543-01
- **Debug Tool for z/OS Version 5 Release 1:**
  - Debug Tool Online Book Index, GC18-9308-00
  - Debug Tool Online Book Shelf, GC18-9307-00
  - Customization Guide, SC18-9306-00
  - Fact Sheet, G325-2408-01
  - Licensed Program Specifications, GC18-9313-00
  - Program Directory, GI10-8598-01
  - Reference and Messages, SC18-9304-00
  - Summary of Commands and Built-in Functions, SC18-9305-00
  - User's Guide, SC18-9302-00
- **Debug Tool Utilities and Advanced Functions Version 5 Release 1**
  - CCCA Program Directory, GI10-5080-04
  - CCCA User's Guide, SC26-9400-02

Coverage Utility User's Guide and Reference, SC18-9303-00

Program Directory, GI10-8599-01

Debug Tool Utilities and Advanced Functions Book Shelf, GC18-9309-00

Debug Tool Utilities and Advanced Functions Book Index, GC18-9310-00

- **Fault Analyzer Version 5 Release 1**

  Brochure, G325-2428-01

  Program Directory, GI10-8666-00

  User's Guide and Reference, SC18-9374-01

- **File Manager Version 5, Release 1:**

  Brochure, G325-2429-01

  License Information, GC18-9383-00

  Program Directory, GI10-8667-01

  Customization Guide, GC18-9378-00

  User's Guide and Reference, SC18-9380-00

  User's Guide and Reference for DB2 Data, SC18-9381-00

  User's Guide and Reference for IMS Data, SC18-9382-00

- **Workload Simulator Version 1 Release 1**

  Brochure, GC18-7063-00

  Program Directory, GI10-3234-00

  User's Guide, SC31-8948-00

  Utilities Guide, SC31-8947-00

  User Exits, SC31-8950-00

  Messages and Codes, SC31-8951-00

  Creating WSim Scripts, SC31-8945-00

  Script Guide and Reference, SC31-8946-00

  Test Manager User's Guide and Reference, SC31-8949-00

Online resources

These Web sites and URLs are also relevant as further information sources:

- The "*Welcome to the IBM Problem Determination and Deployment Tools information center*". This information center contains the documentation for the Problem Determination and Deployment Tools products:
  - Latest Product News
  - Service and Support
  - answers to technical questions
  - the latest fixes and PTFs
  - other documentation

  `http://publib.boulder.ibm.com/infocenter/pdthelp/index.jsp`

- The "*z/OS Problem Determination and Deployment Tools*". This site contains the following type of information:
  - Library
  - News
  - Training
  - Services
  - Support

  http://www-306.ibm.com/software/awdtools/deployment/

How to get IBM Redbooks

You can search for, view, or download Redbooks, Redpapers, Hints and Tips, draft publications and Additional materials, as well as order hardcopy Redbooks or CD-ROMs, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Index

A

address breakpoint 951
Application Monitor 97
analyzing a job 125
analyze Historical Data 126
CPU and Load Module Analysis 130
%In Ready 131
%In Wait 131
%Out 131
%Out Log 132
%Out Rdy 131
%Prob 131
%Supv 131
(Pct indicator) 132
Address 132
Area 132
CPU Pct 131
Module 132
Pct 132
Data Set analysis 141
Data Set Analysis Report 142
Acc Meth 143
CI/Blk Size 143
DD Name 143
EXCP Count 143
Rec Form 143
Rec Length 143
Delay Analysis Report 139
Data 141
Delay Impact 141
Dly Samples 141
Module 141
Offset 141
Samples 141
SVC 141
Detail Instruction Display 137
Offset 137
Sample Data 137
Detail Statement Display 135
Instruction Analysis 136
Instructions 137
Offset 136
Sample Data 137
program module reports 130
select a History Data Set 127
special with history data set 126
Statement Analysis 134
Statement 134
Stmtno 134
Statement Analysis Member selection 137
Time Intervals to Analyze 128
Interval 129
Time Spanned 129
Time Units 129
VSAM Data Set Analysis Detail Report 143
Access 144
Data Source 144
Format 144
LSR 145
Opts 145
Status 144
Type 144
analyzing a job with subsystems 149
CICS analysis 150
CICS Calls in a Transaction display 153
Activity 154
CICS Calls 154
Count 154
Offset 154
Program 154
Total Time 154
CICS Calls in a Unit of Work display 156
Activity 157
CICS Calls 157
Count 157
Offset 157
Program 157
Total Time 157
Trans End 157
Trans Start 156
CICS customization 150
FBII transaction 150
FBIMEI20 program 150
FBIMPL20 program 150
FBIO transaction 150
FBIR transaction 150
CICS displays 151
CICS Transaction Executions display 155
1st Program 156
Started 156
Taskid 156
Time in Appl 156
Time in CICS 156
CICS transactions display 152
Busy% 153
Count 153
Program 153
Tran 153
DB2 analysis 157
DB2 customization 157
DB2 displays 159
DB2 Table Information display 170
Mix OP Seq 170
Plan number 170
Table Name 170
Table number 170
Table Status 170
Table Type 170
IMS analysis 173

IMS customization 173
IMS message analysis display 176
% Total App Time 177
% Total IMS Time 177
Call Type 177
Calls 177
DBD Name 176
Message 176
Messages 176
PCB Nbr 176
Proc OPTS 177
Program Name 176
PSB 176
Segment Name 177
SSC 177
IMS Message Call Analysis display 177
#IN-RDY 179
%CPU 179
Appl Active 179
CALLS 179
DBDNAME 179
IMS Active 179
Offset 178
Program 178
SEGNAME 179
TYPE 179
IMS messages processed by region display 173
%Time in IMS 175
%Time Ready 175
Assoc Samples 175
Count 175
CPU Pct 175
Jobname 175
Message 175
Messages 175
IMS segment search criteria analysis display 179
Call Typ 180
DB Name 180
Field 180
Operation 180
PCB 180
Proc Opt 180
Search 180
Segment 180
SQL Analysis display 161
Auth-ID 161
Count 162
DBRM 161
Module 161
O 162
Oper Id 161
Plan 161
Rel. DB2 CPU Time 162
SQLRC 162
St No. 161
Type 162
SQL Detailed Explain Analysis 164
Acc Type 166
Index Only 166
Join Grp 166
Match Col 166
Method 165
Plan Nbr 165
Table Name 165
UQ JN OB GB 166
SQL End Completion Statistics display 171
Pages Scanned 171
Rows Deleted 171
Rows Examined 171
Rows Inserted 171
Rows Processed 171
Rows Qualified 171
Rows Updated 171
SQLRC 171
Stmt Nbr 171
Stmt Type 171
SQL Explain Detail display 167
Access Creator 168
Access Degree 168
Access Group 168
Access Name 169
Access Type 168
Auth ID 167
Collection ID 169
Column Eval 168
Index Only 168
Join Degree 168
Join Group 168
Lock Mode 168
Match Columns 168
Method 168
MixOP Seq 167
Oper ID 167
Package Version 169
Plan Number 167
Prefetch 168
Query Number 167
Sort Composite 169
Sort New 168
Statement 167
Table Name 167
Table Number 167
Type 167
SQL Statement Completion display 172
Message and SQLCA 172
SQLRC 172
SQL Statement Summary display 162
analyzing a job with subsystemsSQL Statement Full Text 163
batch reports 181
JCL - FBIBATCH 182
ISPSTART statement 185
Jobs, Steps, and Time Ranges page 186
overview 182
printed report layouts 186
Primary Options panel 98
selecting a job 105
active job selection 106
administer a deferred monitoring request 121
analyze a Named Set of Jobs 117

Analyze a Specified Job 110
Batch Step Name 111
CICS Transaction Name 111
CICS Transaction Number 111
IMS Message Id 111
Proc Step Name 111
deferred monitoring request 119
define a deferred monitoring request 119
History Recording 120
Job name 120
Max Duration 120
Max Samples 120
Named Set 120
Start 120
Stop 120
managing and analyzing Named Set of Jobs 113
managing named sets 113
adding or editing a named set entry 116
C 115
Count 115
creating or editing a named set 115
Name 115
select an Active Job to Analyze 106
columns tittle and mask line 107
scrollable area 108
sorting the rows in the display 108
using a mask to filter display rows 108
Set Your Defaults panel 98
settings and customization 97
Specify COBOL and PL/I Listing Data Sets 103
COBOL Listings 104
PL/I Listings 104
Specify Defaults Sampling parameters 99
Data Set Analysis 100
DB2 Activity 100
Default DB2 SSID 100
Delay Analysis 100
Max Samples 100
Specify History Allocation Options 102
Data Class 103
Management Class 103
Model Data Set 103
Space Values 103
Storage Class 103
Volume(s) 103
Specify History Names and Options 101
History Prefix 101
jobname 102
No History file 102
Optional Suffix 101
Prompt to keep 102
system 102
tsopref 102
Application Monitor for z/OS 29
functions of Application Monitor 30
data set analysis reports 31
DB2 analysis 31
deferred monitoring request 33
delay analysis reports 31
history data set 33
monitoring CICS 32
monitoring IMS 32
monitoring jobs 32
online analysis 30
program module reports 31
How Application Monitor samples an application 30
How to have help 30
What are the functions of Application Monitor 30

B

breakpoints view 947

C

console view 947
controlling the flow 948
controls debug view 948

D

DB2 stored procedures 964
adding program logic 974
adding SQL statements 974
allocating z/OS data sets for stored procedure 966
building 390 Stored procedure 978
changing stored procedure 985
checking build errors 982
checking COBOL syntax 979
connections required to z/OS 968
create a COBOL stored procedure 969
creating stored procedure 966
debugging a stored procedure 985
external procedures 965
process for building DB2 stored procedure 980
removing stored procedure 984
running 983
setting up z/OS build properties 967
SQL procedures 965
types 965
External procedures 965
SQL procedures 965
debug console 957
debug console view 947
debug perspective 944
Debug Tool 193, 297
AUTOMONITOR command 53
debugging programs 193
Assembler programs 269
commands supported in remote debug mode 295
IBM Distributed Debugger 292
mainframe interface debugger 291
modifying and using a setup file 286
preparing the program 269
C/C++ programs 194
IBM Distributed Debugger 204
mainframe interface debugger 198
preparing the program 194
COBOL programs 240
IBM Distributed Debugger 265

mainframe interface debugger 259
preparing a COBOL for z/OS and OS/390 application using DTU&AF 246
non LE programs 268
debugging exclusively non-Language Environment programs 268
debugging GuideDebugging CICS non-Language Environment initial programs 269
debugging MVS batch or TSO non-Language Environment initial programs 268
PL/I programs 217
main frame interface 220
preparing PL/I program for debugging 217
remote interface 229
frequently used commands
COMPUTE (COBOL) 51
MONITOR 52
MOVE 52
PLAYBACK BACKWARD command 53
PLAYBACK DISABLE command 54
PLAYBACK ENABLE command 53
PLAYBACK FORWARD command 53
PLAYBACK START command 53
PLAYBACK STOP command 54
subsystems 305
CICS 306
compiling for DB2 and Debug Tool 346
DB2 346
DB2 stored procedures considerations 348
debug CICS programs 343
debugging with IMS V8 and above 349
IMS 349
invoking a DB2 application using the TEST runtime option 346
precedence of the information on the LE Runtime Options panel 362
prepare a CICS program 314
create and store a CADP profile 318
create and store a DTCN profile 315
need to link edit EQADCCXT into your program? 314
start Debug Tool under CICS 335
support for debugging under CICS 306
Activating CICS Non-Language Environment Exits 307
Basic CICS customization 306
Enabling communication between Debug Tool and a remote debugger 310
Enabling the CADP transaction 311
Enabling the CADP Web interface 311
Sharing the CADP repository profile items among CICS systems 312
Sharing the DTCN repository profile items among CICS systems 307
traditional methods of debugging under IMS 349
using DTSU to invoke your DB2 application 346
TEST runtime option
prompt_level 44
UNIX System Services 297
compiling a C/C++ application 300
debugging a USS application 300
invocation 300
setup 300
specifying the TEST runtime option 298
using C/C++ and Debug Tool under USS 299
Unix System Services
common considerations for all languages 298
special considerations for running DLL applications from the shell 299
Debug Tool and Debug Tool Utilities and Advanced Functions 35, 55
advanced functions 62
ALLOCATE 62
CALL %FA 62
DESCRIBE 62
FREE 62
LOADDEBUGDATA 62
PLAYBACK * 62
SET ASSEMBLER 62
SET AUTOMONITOR 62
Debug Tool interfaces 36
batch mode 36
built-in functions 54
%GENERATION (PL/I) 54
%HEX 54
compiler options 43
Assembler 43
C/C++ 43
COBOL 43
PL/I 43
Dynamic Debug Facility 55
finishing Debug Tool session 54
QQUIT 54
QUIT 54
QUIT ABEND 54
QUIT DEBUG 54
frequently used commands 50
AT 50
AT CHANGE 50
CLEAR 51
COMPUTE (COBOL) 51
DESCRIBE 51
DISABLE / ENABLE 51
GO 51
GO BYPASS 53
GOTO 53
LIST 52
MONITOR LIST 52
MOVE (COBOL) 52
playback (COBOL) 53
QUERY 52
RUNTO 53
SET 52
set automonitor (COBOL and PL/I) 53
STEP 53
WINDOW CLOSE/OPEN/SIZE/ZOOM 54
full-screen mode 41
global preferences file enhancement 49
function 50
global preferences file content 50

lobal preferences file location 49
restrictions 49
using EQAOPTS options file 49
IBM Distributed Debugger 39
link-edit options 43
log window 41
monitor window 42
remote debug mode 36
runtime TEST option 43
commands_file 44
preferences_file 44
prompt_level 44
sample runtime options 45
specifying TEST runtime option 46
test_level 44
source window 41
special files 48
WebSphere Studio Enterprise Developer Debugger 37
Debug Tool Utilities 55
code coverage 61
compiling, converting, linking 59
creating and managing setup files 56
preparing IMS run-time environment 62
Debug Tool Utilities 365
COBOL and CICS Command Level Conversion Aid 383
conversion tools requirements 383
convert old 68/74 COBOL to 85 COBOL 388
setting up for convert and compile 383
start conversion process 385
Coverage Utility 366
compiling programs 371
control file 372
customization 366
display statistics 377
link step 377
modifying defaults 370
monitor panel 382
monitor session 374
report 375
requirements 368
running 368
setup JCL 373
setup step 377
start application 377
start monitor session 377
stop monitoring 378
summary and annotated listing report 381
summary report 379
using 369
debug view 946
disable breakpoints 953
display view 948

E

editor view 947
entry breakpoint 950
Examples
File Manager
COBOL copybook 597
COPY REPLACING statement 591
copybook with nested multiple copybooks 593
Workload Simulator
APPCLU and TP statements 798
NTWRK statement 797
TCPIP and DEV statements 798
VTAMAPPL and LU statements 798
exception handling 959
executive summary 3
Application Monitor for z/OS 4
Debug Tool and Debug Tool Utilities & Advanced Functions 5
Fault Analyzer for z/OS 5
File Manager for z/OS 6
Workload Simulator for z/OS and OS/390 7
expressions view 947

F

Fault Analyzer 395
additional information 509
additional requirements 510
hardware requirements 510
healthy installation and customization checklist 512
library names after you finish installing 512
software requirements 510
target system requirements 511
batch reanalysis report 475
batch reanalysis options 476
creating your own batch reanalysis job 479
data sets used 478
initiating batch reanalysis 478
interactive analysis 423
abend job information 447
converting STORE CLOCK values 463
data sets used for interactive reanalysis 467
disassembling object code 462
displaying associated storage areas 448
displaying chained data areas 461
displaying source code 454
displaying storage locations 455
event summary 429
exit 428
expanding messages and abend codes 451
interactive report 428
mapping storage areas 459
options 424
options in effect 448
prompting for compiler listing or side file 465
reanalysis 427
synopsis 428
system-wide information 432
user-specific report formatting 463
real-time analysis report 469
analysis report 471
controlling the read-time analysis with options 470
dump registration processing 472
fault history file selection 470

pointing to listings with JCL DD statements 471
SYSLOG summary 472
using the program SNAP interface 472
settings and customization 395
action-bar pull-down menus 418
Actions command 418
applying an action against a particular fault 416
changing history file or the view displayed 406
fault entry list column configuration 409
history file properties 417
invocation exits 396
invoking the interface 398
ISPF interface 396
matching and selecting fault 412
refreshing fault entry information 418
resetting history file access information 417
setting up views 401
using views 400
subsystems 481
CICS 482
configuring Language Environment for CICS 483
controlling CICS transaction abend analysis 484
defining required program 483
ensuring transaction abend analysis is not suppressed by DUMP 486
installing the MVS post-dump exit IDIXTSEL 488
Language Environment abend considerations 488
preventing LE from causing the CICS trace to wrap 486
specifying data sets through the IDIOPTS DDname 487
storage requirements 489
using a CFA from a CICS terminal 485
using CFA from an MVS console 486
DB2 502
DB2 and Language Environment 502
DB2 stored procedures 503
IMS 505
performing WebSphere or Java dump analysis 505
selecting a WebSphere or Java dump data set 506
type of dumps supported 506
Fault Analyzer for z/OS 63
compiler options 72
Assembler 73
C 72
C++ 72
COBOL compilers other than OS/VS COBOL 73
Enterprise PL/I 73
OS/VS COBOL 72
PL/I compilers other than VisualAge PL/I and Enterprise PL/I 73
VisualAge PL/I 73
Fault history file 67
mechanics of Fault Analyzer 67
preparing your program for Fault Analyzer 72
PTF information 64
summary of real-time analysis 69
supported application environments 68
Assembler C/C++ 68
CICS 68
COBOL PL/I 68
DB2 68
IMS 68
Java 69
Language Environment 68
MQSeries 69
UNIX System Services 68
WebSphere 69
TEST option considerations 73
File Manager 517
APF authorization 595
application programmer ISPF profile alert 595
check the IMS feature installation 575
copybook and Include 597
Data Set Generate (DSG) 532
DB2 561
copy data to a table 572
create a table 567
edit a table 562
external REXX functions used 527
CO 528
DROP 541
DSC 527
EXIT 528
FLD 540
NCO 540
PRINT 528
RETURN 540
STOP IMMEDIATE 528
WRITE 528, 540
generate data using a template 537
global update of JCL 522
how to set up the batch job 525
IMS 575
copybook and include 597
hints and tips 593
navigating an IMS database 576
template processing 586
ISPF skeleton modification 594
modify the JCL for generic use 533
PF keys 595
processing options section on data set edit panel 587
record structure defined in source application program 597
result of a batch FIND command 549
selecting options before creating a template 592
systems programmer notes 594
template processing 586
process COPY REPLACING statements 591
remembers the copybook 586
using the SELECT command in template edit 587
test data 599
copying DB2 data 615
copying MVS data 606

creating DB2 test data 611
extract and load IMS data 627
File Export 638
importing and exporting DB2 data 619
twenty nine different keylists 595
use 518
compare data sets using templates 558
copy selected variably blocked records to another file 541
create one VSAM file using another as a model 528
dynamic template 555
information about a load module 554
initialize a VSAM file with low-value records 531
perform a global find and replace 522
performance guidelines 518
populate this newly created file 533
QSAM allocation using a model 550
replace a string in a specific location in a file 541
selective copying 521
simple copy 519
split a single file into constituent record types 538
work with VTOC 551
using File Manager online 522
using in batch 524
watch out for that bad disposition 597
what to do when a copybook fails to compile 595
working with DB2
copy data to a table 572
edit a table 562
utility function to create a table 567
working with IMS
navigating an IMS database 576
File Manager for z/OS 75
File Manager for DB2 data 77
File Manager for IMS data 77
File Manager for working with MVS data sets 76
major functions 77
batch processing 79
REXX functions 78
support for DFSORT 78
templates 78
overview 76

H

how to process COPY REPLACING statements 591

L

line breakpoint 952
load breakpoint 952

M

modules view 948
monitors view 947–948

O

overview of the problem determination and deployment tools 9
Application Monitor for z/OS 10
use of Application Monitor 10
what Application Monitor can do 11
what are the major functions of Application Monitor 12
Debug Tool for z/OS and Debug Tool Utilities and Advanced Functions 13
functions 16
what Debug Tool can do 14
what Debug Tool provides 15
Fault Analyzer for z/OS 16
File Manager for z/OS 20
Workload Simulator for z/OS and OS/390 23

R

Redbooks Web site 999
Contact us xix
registers view 947
remote development - the debugger 943

S

scenario
Application Monitor and CICSplex configuration 905
Application Monitor and Dynamic Program Link 921
Application Monitor reports 922
Application Monitor set-up 922
system configuration 921
Application Monitor and Function Shipping 920
Application Monitor reports 920
Application Monitor set-up 920
system configuration 920
Application Monitor and Transaction Routing 906
Application Monitor set-up 906
Application Monitors reports 909
system configuration 906
Application Monitor and parallel sysplex 927
Application Monitor reports 929
Application Monitor set-up 928
CICS and DB2 data sharing group 933
Application Monitor reports 936
Application Monitor set-up 935
system configuration 933
system configuration 928
Fault Analyzer and File Manager 887
set up the components 888
tracking an abend in the application 893
walkthrough of the CICS Trader application 889
Is the error in DB2 data or program logic? 869
set up the components 870
tracking a problem in the application 871
Scenario 1 - Invalid VSAM data generating an abend 825
tracking an abend in the application 827
Scenario 2 - Using Debug Tool 843
set up the components 844
walkthrough of the batch Trader application 844
scenarios 807
application software 810

overview 808
system configuration 815
storage mapping view 947
storage view 947

T

TEST runtime option 43
threads view 948
Trader application 817
Trader application in batch 822
Trader application in CICS 818

V

variables 953
variables view 947
Views
Debug Tool
USS
debugger asking for source file location on invocation 301
found source 302
providing the correct absolute address 302
run from the directory it was compiled 303
setting and clearing Language Environment runtime options in the USS shell 299
File Manager
browse entry 588
browse entry (dynamic PSB) 577
cancel command 566
COBOL copy replacing options 592
comparison characteristics 558
compilation errors 596
confirm cancel 567
copy 574
copybook compile listing at the point of error 596
create table 569
database data set 577
database positioning 578, 580, 583, 585
DB2 copy utility 573
DB2 edit 563
DB2 object 573
DB2 object functions 568
deleting rows 565
field selection edit 589
field selection/edit 590
IMS edit/browse options 576
inserting rows 564
object list lines commands 572
object list utility 571
options before creating a template 592
options section 587
primary options 562
roll-back 567
segment layout selection 581
table created 570
table edit 563
tables, views, aliases 571
template edit 588
utility function 568
utility functions 570, 572
view specification 578
Visuals
Debug Tool
debugger control Icons 38
invoking program to be debugged on the mainframe 38
preferences file syntax 45
remote debugger while debugging 40
WebSphere Studio Workbench Debug Perspective 37
WSED initial program status message 38
WSED screen when debugging 39

W

watch breakpoint 951
Workload Simulator 81, 791
APPCLU statement 798–799
attributes of the simulated resources 797
connections between the simulated resources and the system under test 797
DEV statement 798–799
DIST statement 799
FE statement 799
FILE statement 799
general order of network definition statements 799
IF statement 799
INCLUDE statement 799
latest enhancements 92
logical configurations 85
LU statement 797, 799
message generation decks 86, 800
methods for creating message decks 87
using STL and the STL Translator 88
using the Interactive Data Capture Utility 89
using the script generator utility 89
using the SNA 3270 Reformatter Utility 90
writing message generation statements 88
MSGDISK statement 799
network definition statements 86
NTWRK statement 799
NTWRKLOG statement 799
operating WSim 91
overview 82
resources WSim can simulate 83
testing with WSim 83
use of WSim 83
PATH statement 799
physical configurations 85
TCP/IP application configuration 85
VTAM and CPI-C application configuration 85
RATE statement 799
RN statement 799
script preparation 86
scripting 791
coding STL programs 800
network definition statements 797
WSim operator command 804
WTM generated scripts review 792
SIDEEND statement 799

SIDEENT statement 799
SIDEINFO statement 799
special information 797
system configuration 84
Logical unit (LU) 84
Session 84
Transaction program (TP) 84
TCPIP statement 798–799
testing scripts 90
TP statement 798–799
types of the simulated resources on the network 797
UDIST statement 799
UTBL statement 799
UTI statement 799
VTAMAPPL statement 797, 799
WSim output 90
WSim Test Manager 91
Workload Simulator and Workload Simulator Test Manager 667
settings and customization 667
installing WSim 668
installing WTM 669
Workload Simulator Test Manager 675
concepts 676
creating a project 677
creating a testgroup 740
Display Monitor Facility 776
load test 778
organizing test cases 708
regression test 770
scheduling and running a simulation 698
simulation reports 703
SNA trace to create testcase 788
testcases 680
user data tables 749
WSim in batch mode 765
WTM operating modes 676

Redbooks

IBM Application Development and Problem Determination Tools for z/OS and OS/390

(1.5” spine)
1.5”<-> 1.998”
789 <->1051 pages

IBM Application Development and Problem Determination Tools for z/OS and OS/390

Overview of the z/OS problem determination and deployment tools

Coverage of V5 Debug Tool, Fault Analyzer, and File Manager

Using V2 Application Monitor and V1 Workload Simulator

This IBM Redbook introduces the IBM Problem Determination and Deployment Tools for z/OS and OS/390.

The tools covered are:

- IBM Application Monitor for z/OS, Version 2, Release 1
- IBM Debug Tool for z/OS, Version 5, Release 1
- IBM Fault Analyzer for z/OS, Version 5, Release 1
- IBM File Manager for z/OS, Version 5, Release 1
- IBM Workload Simulator for z/OS and OS/390, Version 1, Release 1

This comprehensive suite of powerful yet easy-to-use tools helps developers to efficiently build, test, and service applications while minimizing errors and increasing cost-effectiveness. The tools allow users to quickly identify and resolve problems that occur in batch, IMS, DB2, CICS, and UNIX System Services applications.

For each tool, this book provides step-by-step instructions for installation and customization, a detailed discussion of the features and capabilities, and guidance for using the tool in your environment.

Practical scenarios demonstrate how to apply the tools to monitor, manage, debug, and analyze applications in a real-world environment.

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks

SG24-6492-00 ISBN 0738497762