Explore the new features and functions of IMS 12
Understand advantages and applicability of IMS 12
Plan for installation of or migration to IMS 12

Paolo Bruni
Isabelle Bruneel
Angie Greenhaw
Dougie Lawson
Jorge Alberto Luz Ribeiro
Egide Van Aershot
Note: Before using this information and the product it supports, read the information in “Notices” on page xxiii.

First Edition (October 2011)

This edition applies to Version 12, Release 1 of IBM IMS Transaction and Database Servers (program number 5635-A03) and Version 2, Release 1 of IBM Enterprise Suite (program number 5655-T62).
## Contents

**Figures** .................................................................................. xi

**Examples** .............................................................................. xv

**Tables** .................................................................................. xxi

**Notices** ................................................................................ xxiii
**Trademarks** ........................................................................... xxiv

**Preface** ................................................................................ xxv
The team who wrote this book ................................................... xxv
Now you can become a published author, too! .......................... xxvii
Comments welcome .................................................................. xxvii
Stay connected to IBM Redbooks ............................................. xxviii

**Chapter 1. IMS 12 at a glance** ............................................... 1
  1.1 IMS product positioning ................................................... 2
  1.2 System enhancements .................................................... 2
  1.3 Transaction manager and connectivity enhancements .......... 4
  1.4 Database Manager enhancements ..................................... 5
  1.5 DBRC enhancements ..................................................... 7
  1.6 Main enhancements from IMS 10 to IMS 11 ......................... 8

**Chapter 2. System enhancements** .......................................... 11
  2.1 TSO SPOC ........................................................................ 12
    2.1.1 SPOC background information .................................... 12
    2.1.2 SPOC description ..................................................... 13
    2.1.3 SPOC enhancements with IMS 12 ............................... 14
    2.1.4 Using TSO SPOC .................................................... 14
    2.1.5 Considerations for using the IMPORT command .......... 17
  2.2 Dynamic resource definition .............................................. 17
    2.2.1 Background information about DRD ......................... 17
    2.2.2 Description of DRD ................................................ 19
    2.2.3 DRD enhancement with IMS 12 ................................. 21
    2.2.4 Using DRD ............................................................ 21
    2.2.5 Considerations for using DRD ..................................... 24
  2.3 ACB online change ............................................................ 25
    2.3.1 Background information about online change ............ 25
    2.3.2 Description of online change ..................................... 26
    2.3.3 ACB online change IMS 12 enhancements .................. 27
    2.3.4 Using ACB online change ........................................ 28
    2.3.5 Considerations for the ACB online change ................. 30
  2.4 The /DIAGNOSE command ................................................ 30
    2.4.1 Background on the /DIAGNOSE command ................. 30
    2.4.2 Description of the /DIAGNOSE command .................... 31
    2.4.3 /DIAGNOSE IMS 12 enhancements ........................... 32
    2.4.4 Using the /DIAGNOSE command ............................... 33
    2.4.5 Using the /DIAG SNAP command .............................. 39
    2.4.6 Considerations for the /DIAGNOSE command ............ 41
Chapter 3.  Database enhancements
   3.1  Full function database enhancements ............................... 68
       3.1.1  Overview of the database pool storage enhancements .......... 68
       3.1.2  Dynamic full function database buffer pools .................. 68
   3.2  High availability large database enhancements .................. 72
       3.2.1  HALDB online reorganization ownership release ............... 73
       3.2.2  Parallel migration to HALDB: IMS 10 and IMS 11 small programming enhancement 74
       3.2.3  Reuse of HALDB partition database names ....................... 75
       3.2.4  Reorganization numbers handled differently by timestamp recovery .... 75
   3.3  Fast Path database enhancements .................................. 76
       3.3.1  Reduced logging for DEDB changed data capture ............... 76
       3.3.2  Full segment logging option ................................ 77
       3.3.3  Improved diagnostic message for DEDB data sharing ............ 77
       3.3.4  Fast Path DEDB secondary index support ....................... 78
       3.3.5  DEDB buffer pool enhancements ................................ 97
   3.4  CICS threadsafe support ............................................ 99
   3.5  Miscellaneous enhancements ....................................... 100
       3.5.1  Status messages for DB exit routines .......................... 100
       3.5.2  Lock timeout message and logging ................................ 102
       3.5.3  Batch data sharing abend elimination .......................... 103
       3.5.4  RACF user ID in Data Capture batch log records .............. 104
       3.5.5  Increased VSAM pools ........................................ 104
Chapter 7. IMS repository

7.1 Introducing and explaining the value of the repository  
7.2 Components of the repository  
7.3 Repository Server  
7.4 Repository data sets  
7.5 Common Service Layer  
7.6 Batch utilities  
7.7 Migrating to the repository  
7.8 IMSRSC repository access  
7.9 Using the repository in business operations  
7.10 Security considerations  
7.11 DRD user interface enhancements
Chapter 8. Installation and migration considerations ................................................. 279
  8.1 Prerequisites and coexistence ..................................................................... 280
    8.1.1 Preventive Service Planning ............................................................... 280
    8.1.2 Support procedures .......................................................................... 281
  8.2 Fix Category HOLDDATA ........................................................................... 281
    8.2.1 IMS-specific fix categories ............................................................... 282
    8.2.2 Examples of non-IMS-specific fix categories .................................... 282
  8.3 Coexistence with IMS 12 ......................................................................... 282
  8.4 General coexistence considerations ........................................................... 283
    8.4.1 DBRC coexistence considerations ...................................................... 283
    8.4.2 Exit routine coexistence considerations ............................................. 284
    8.4.3 Fast Database Recovery coexistence considerations ......................... 284
    8.4.4 Common Queue Server coexistence ................................................... 284
    8.4.5 IMSplex coexistence considerations .................................................. 284
    8.4.6 Program specification blocks changes ................................................. 285
    8.4.7 Generalized sequential access method changes .................................... 285
    8.4.8 Log records ....................................................................................... 285
    8.4.9 MSC coexistence considerations ....................................................... 285
    8.4.10 Syntax Checker coexistence considerations ....................................... 285
    8.4.11 IMS utilities coexistence considerations ........................................... 286
    8.4.12 IMS Tools migration and coexistence ............................................... 286
    8.4.13 IBM Information Center .................................................................... 286
  8.5 Installation Verification Program .................................................................. 286
    8.5.1 Enhancements overview .................................................................... 287
    8.5.2 File tailoring ..................................................................................... 300
    8.5.3 IVP jobs and tasks ............................................................................. 300
    8.5.4 Executing IVP tailored jobs and tasks ................................................. 301
    8.5.5 Changes to the IVP for IMS 12 ............................................................ 302
  8.6 Syntax Checker ......................................................................................... 303
    8.6.1 Starting Syntax Checker ..................................................................... 304
    8.6.2 Using Syntax Checker ........................................................................ 305
    8.6.3 IMS Release and Control Region Type entry panel ................................ 306
    8.6.4 Keyword Display panel ..................................................................... 307
    8.6.5 Exiting Syntax Checker ....................................................................... 307
  8.7 Installation and migration tasks .................................................................. 308
    8.7.1 Migration considerations ..................................................................... 308
    8.7.2 Discontinued support in IMS ............................................................... 311
    8.7.3 Fallback considerations ..................................................................... 311
    8.7.4 DBRC fallback considerations ............................................................ 312
  8.8 Review of migration considerations ............................................................ 313
    8.8.1 Migrating to IMS 12 Database Manager ............................................ 313
    8.8.2 Migrating to IMS 12 Transaction Manager ......................................... 314
    8.8.3 Migrating to the IMS 12 system .......................................................... 315

Chapter 9. IMS Enterprise Suite V2.1 ................................................................. 323
  9.1 Enterprise Suite V2.1 contents ................................................................... 324
    9.1.1 Acquiring and installing Enterprise Suite V2.1 .................................... 325
  9.2 Enhancements for Enterprise Suite V2.1 ..................................................... 326
  9.3 IMS Enterprise Suite Connect APIs for C .................................................. 326
    9.3.1 Overview of API for C ....................................................................... 327
Chapter 10. Tools for IMS 12

10.1 IMS Tools general information ................................................................. 386
  10.1.1 Useful links .................................................................................. 386
  10.1.2 IMS Tools portfolio ...................................................................... 387
  10.1.3 IBM Tools Base Pack for/z/OS Version 1 (5655-V93) ................ 388
  10.1.4 IMS Fast Path Solution Pack for/z/OS (5655-W14) .................... 390
  10.1.5 IMS Database Solution Pack for/z/OS (5655-S77) ....................... 394
  10.1.6 IMS Recovery Solution Pack for/z/OS (5655-V86) ...................... 399
  10.1.7 IMS Performance Solution Pack for/z/OS (5655-S42) ................. 401
  10.1.8 Complementary products tools ...................................................... 407

10.2 Examples of specific product updates for IMS 12 support .................. 409
  10.2.1 High Performance Fast Path Utilities updated for IMS 12 ............ 410
  10.2.2 Library Integrity Utilities updated for IMS 12 ............................... 423
  10.2.3 IMS Connect Extension updated for IMS 12 ............................... 424
  10.2.4 IMS Performance Analyzer updated for IMS 12 ......................... 428

Appendix A. Environment description ............................................................. 431
  A.1 Hardware and z/OS configuration ....................................................... 432
  A.2 IMS configuration .......................................................................... 435
    A.2.1 TCP/IP MSC links ...................................................................... 436
    A.2.2 Base Primitive Environment Database Recovery Control .......... 442
  A.3 IMS repository .................................................................................. 443
  A.4 IMS Connect .................................................................................... 444
  A.5 Shared queues .................................................................................. 449

Appendix B. Recent maintenance: IMS 12 APARs ........................................ 455

Abbreviations and acronyms ........................................................................... 457

Related publications ....................................................................................... 461
Contents

IBM Redbooks ................................................................. 461
Other publications ............................................................ 461
Online resources ............................................................... 462
Help from IBM ................................................................. 462

Index .................................................................................. 463
Figures

2-1  IMS Application Menu panel .................................................. 12
2-2  The SPOC architecture ......................................................... 13
2-3  TSO SPOC initial panel .......................................................... 13
2-4  A QUERY DBDESC Command ......................................................... 15
2-5  The IMPORT UPDATE command .................................................. 16
2-6  Online change process .............................................................. 18
2-7  DRD flow .............................................................. 19
2-8  IMS Managed Resources panel for DRD ............................................ 19
2-9  Creating a new transaction called DLRINQ (part 1 of 2) ................. 22
2-10  Creating a new transaction called DLRINQ (part 2 of 2) ............... 22
2-11  Import DLRINQ Transaction to I12C IMS system (part 1 of 2) ....... 23
2-12  Import DLRINQ transaction to I12C IMS system (part 2 of 2) ....... 23
2-13  INITIATE command syntax ...................................................... 27
2-14  The Online Change prepare phase ............................................ 29
2-15  The online change commit command ....................................... 29
2-16  IMS logging process overview ................................................ 42
2-17  Sequential data set striping .................................................. 43
2-18  Shared queue environment ..................................................... 48
2-19  EAV characteristics ................................................................. 53
2-20  BUFPOOLS macro syntax ......................................................... 59
2-21  DFSFIXnn BLOCKS syntax ........................................................ 62
3-1  Overview of dynamic full function buffer pools .................................. 69
3-2  HALDB databases ................................................................. 72
3-3  Parallel unload ................................................................. 74
3-4  Target, source, and pointer segment ......................................... 79
3-5  Secondary Index where root is target ......................................... 80
3-6  Secondary Index where dependent is target .................................. 81
3-7  SENSEG sequence different from full function, but not processing sequence ................................................................. 81
3-8  Secondary indexes on DEDB ...................................................... 84
3-9  ROOT (COURSE) is the target; INSTRUCT is the source .................... 85
3-10  DBD for DEDB with target ROOT, source INSTRUCT .................... 86
3-11  DBD for index with target ROOT, source INSTRUCT ..................... 87
3-12  CLASS is target, STUDENT is source ....................................... 88
3-13  DBD for DEDB with target CLASS, source STUDENT .................... 88
3-14  DBD for index with target CLASS, source STUDENT ..................... 89
3-15  Processing sequence with target class ..................................... 90
3-16  User partitioning for secondary indexes .................................. 91
3-17  Multiple secondary index segments ......................................... 92
3-18  DBD for DEDB with multiple secondary indexes ............................ 93
3-19  Secondary index segment layouts .............................................. 94
3-20  Building Fast Path index with HPFPU ........................................ 96
3-21  TCB switching with threadsafe support for CICS accessing IMS database ................................................................. 100
3-22  Control Area reclaim ............................................................. 106
4-1  SMQ parameters in DFSDDxxx proclib member ................................ 111
4-2  AOS and RRS combinations at a glance ..................................... 113
4-3  General flow of a synchronous message using RRS ........................ 115
4-4  Synchronous SL0 processing with XCF communication ...................... 119
4-5  Synchronous SL1 processing with XCF communication ...................... 120
7-6  Enhanced EXPORT panel in the MR application for target data set selection .......................... 265
7-5  The IMS Manage Resources application menu ................................................................. 264
7-4  The IMS Application Menu .......................................................................................... 264
7-2  IMS repository parameter consistency requirements ......................................................... 215
7-1  Components of the IMS repository .............................................................................. 201
6-5  TCP/IP MSC configuration ............................................................................................ 188
6-4  TCP/IP MSC with IMS shared queues .......................................................................... 184
6-3  MSC using TCP/IP for IMS to IMS connectivity ............................................................. 170
6-2  Remote IMS Connect configuration .............................................................................. 169
6-1  Asynchronous IMS to IMS ............................................................................................ 162
5-9  The CHANGE.RECON syntax ...................................................................................... 158
5-8  The LIST.DB command syntax ..................................................................................... 153
5-7  The INIT.RECON command ......................................................................................... 151
5-6  Syntax of the CHANGE.CAGRP command ................................................................. 148
5-5  The CLEANUP.RECON command syntax .................................................................. 146
5-4  The GENJCL.IC Command syntax ............................................................................. 144
5-3  The NOTIFY.IC command syntax ............................................................................... 141
5-2  The CHANGE.CA command syntax ............................................................................ 140
5-1  RECON records relationship ......................................................................................... 139
4-9  Example of expiration time at MQ message level ........................................................... 135
4-8  DFS2082 CM0 implementation ...................................................................................... 131
4-7  OTMA ACEE aging value ............................................................................................... 128
4-6  OTMA ACEE reduction ................................................................................................. 127
4-5  The U-series of steps in the IVP related to the IMS repository function ..................... 276
4-4  IMS 11 DFSAPPL menu ............................................................................................... 288
4-3  Selecting the suboptions ............................................................................................... 289
4-2  IMS 11 IVP Notices pane (presented the first time only) ............................................. 293
4-1  Starting the IMS 11 IVP dialog ...................................................................................... 287
8-3  Selecting the Use existing tables option ...................................................................... 289
8-2  IMS 11 DFSAPPL menu ............................................................................................... 288
8-1  Selecting the suboptions ............................................................................................... 289
8-4  Selecting the Use existing tables option ...................................................................... 289
8-5  Choosing option A for variable export ...................................................................... 290
8-6  Entering the INSTATBL and export data set names ...................................................... 291
8-7  Invoke the IVP dialog from ISPF ................................................................................. 292
8-8  The IMS 12 IVP logo panel .......................................................................................... 292
8-9  IMS 12 IVP Notices pane (presented the first time only) ............................................. 293
8-10 Choosing the IMS system environment ...................................................................... 293
8-11 Choosing the IVP suboptions ....................................................................................... 294
8-12 Confirming your suboption selections ....................................................................... 295
8-13 Choosing to create or update the working tables ......................................................... 295
8-14 Variable Gathering Table Merge runs as normal ........................................................ 296
8-15 Table Merge process completed ................................................................................ 296
Examples

2-1 The /DIAG SNAP commands ................................................................. 39
2-2 JCL for printing DIAG records with exit DFSERA30 .......................... 41
2-3 JCL to allocate OLDS ................................................................. 46
2-4 DFSVSMxx using the new BUFS TOTOR parameter ......................... 46
2-5 CQS JCL sample ................................................................. 50
2-6 BPE configuration with new trace entries ......................................... 51
2-7 DISPLAY and UPDATE CQS trace command sample ......................... 52
2-8 Allocate OLDS ................................................................. 55
2-9 WADS allocation ................................................................. 56
2-10 RDS allocation ................................................................. 56
2-11 Message queue allocation using EAV ........................................ 57
2-12 D FSVSMxx sample to fix pools ............................................... 63
3-1 VSAM buffer pool specifications in DFSVSMxx ................................ 69
3-2 Changed VSAM specifications in DFSD Fxxx .................................... 70
3-3 OSAM buffer pool specifications in DFSVSMxx ................................ 70
3-4 Changed OSAM specification in DFSDFxxx .................................... 70
3-5 Add and change commands ............................................................ 71
3-6 Type-2 QUERY command .......................................................... 71
3-7 TSO SPOC samples ............................................................... 71
3-8 Ownership release indication ....................................................... 73
3-9 Control parallel unload ............................................................. 75
3-10 DBDGEN exit parameters ......................................................... 76
3-11 DBRC commands for full REPL segment logging .......................... 77
3-12 Acknowledgement from other data sharing IMS ........................... 78
3-13 Secondary indexes for a DEDB .................................................. 82
3-14 Secondary index DBDGEN input ............................................... 83
3-15 Other secondary index-related statements ................................... 83
3-16 PCB for processing by using the secondary index STUDXSEG ......... 90
3-17 DBD additional keywords for user partitioning ............................. 91
3-18 DBDGEN statements for multi secondary index segments ............. 93
3-19 DBD for index database with multiple secondary indices ............. 93
3-20 JCL and control information for INDEXBLD function ................... 96
3-21 IMS 11 Fast Path specifications in DFSDFXXX ............................ 97
3-22 DFS2842I message ............................................................... 100
3-23 DFS2838I message ............................................................... 101
3-24 DFS2406I message ............................................................... 101
3-25 Multiple DFS2291I messages ...................................................... 102
3-26 Short form of the DFS2291 message ........................................ 102
3-27 LOCKTIME definition in DFSVSMxx ......................................... 103
3-28 DF SDFxxx DIAGNOSTIC_STATISTICS section ......................... 103
3-29 LOG specification in DBD of the DBDGEN utility ......................... 104
3-30 DFS0730I message ............................................................... 105
4-1 LIST RECON STATUS and MINVERS V12 .................................... 114
4-2 Startup messages and OTMA CM1 SL0 transaction monitoring messages with RRS 116
4-3 Message flow on the front-end of I12A for an RRS scenario ............ 117
4-4 Message flow on back-end I12C for an RRS scenario ........................ 117
4-5 Startup messages and OTMA CM1 SL0 transaction monitoring messages with XCF 121
4-6 Message flow on front-end I12A for an XCF scenario ..................... 121
6-35 Output from the VIEWMSC ALL command.
6-36 Output from the STOPLINK linkname command.
6-37 Output from the STOPLINK DFSL0001 MSC2A2B command.
6-38 Output from the STARTMSC MSC2A2B command.
6-39 Output from the STOPMSC MSC2A2B command.
6-40 Output from the STARTRTM HWSI12B1 command.
6-41 Output from the STOPRTM HWSI12B1 command.
6-42 Output from the STOPSLN MSCxxxxx command.
6-43 Output from the DELETE LINK NAME(DFSL0002) command.
6-44 Output from the QRY IMSCON TYPE(CONFIG) SHOW(ALL) command.
6-45 Output from the QRY IMSCON TYPE(MSC) NAME(*) SHOW(ALL) command.
6-46 Output from the QRY IMSCON TYPE(MSC) NAME(MSC2B2A) SHOW(LINK) command.
6-47 Output from the QRY IMSCON TYPE(RMTIMSCON) NAME(*) SHOW(ALL) command.
6-48 Output from the QRY IMSCON TYPE(SENDCLNT) NAME(*) SHOW(ALL) command.
6-49 Output from the UPD IMSCON TYPE(LINK) NAME(DFSLxxxx) STOP(COMM).
6-50 Output from the UPD IMSCON TYPE(LINK) NAME(x) MSC(m) STOP(COMM) command.
6-51 Output from the UPD IMSCON TYPE(MSC) NAME(MSC2A2B) START(COMM).
6-52 Output from the UPD IMSCON TYPE(MSC) NAME(MSC2A2B) STOP(COMM).
6-53 Output from the UPD IMSCON TYPE(RMTIMSCON) NAME(rmtinm) START(COMM).
6-54 Output from the UPD IMSCON TYPE(RMTIMSCON) NAME(rmtinm) STOP(COMM).
6-55 Output from the UPD IMSCON TYPE(SENDCLNT) NAME(cl) RMTIMSCON(rmt) STOP(COMM).
6-56 IMS stage 1 macros for a TCP/IP MSC link (shared queues).
6-57 Additional statement in IMS.PROCLIB(DFSDCxxx)
6-58 Output from the UPD MSPLINK NAME(*) START(GENLOGON) command.
6-59 Output from the UPD MSPLINK NAME(*) STOP(GENLOGON) command.
6-60 Modified MSC definitions in the local IMS Connect.
6-61 Modified MSC definitions in the remote IMS Connect.
6-62 Local RACF definitions.
6-63 Remote RACF definitions.
6-64 Changes to the RMTIMSCON definition for security.
6-65 Remove the old recorder trace.
6-66 Updated IMS.PROCLIB(BPEHWSxx) member.
6-67 Defining the GDG base data set.
6-68 BPE command to activate the recorder trace.
6-69 BPE command to terminate the recorder trace.
6-70 IMS Connect configuration member.
6-71 Output from the QUERY MEMBER command.
6-72 New layout for RSM.
6-73 REXX handling for *REQSTS* RSM.
6-74 Output from IMS Connect.
6-75 Output from the VIEWPORT command with a partial read client.
6-76 Output from the QRY IMSCON TYPE(PORT) command with a partial read client.
6-77 The STOPCLNT command.
6-78 The UPD IMSCON TYPE(CLIENT) command to terminate a partial read client.
7-1 JCL for allocating user repository and RS catalog repository data sets.
7-2 BPE configuration member for Repository Server address space.
7-3 FRPCFG configuration member parameter definitions.
7-4 Repository Server startup procedure JCL.
<table>
<thead>
<tr>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7-5</td>
<td>SCI initialization PROCLIB member CSLSI12D</td>
</tr>
<tr>
<td>7-6</td>
<td>SCI startup procedure JCL</td>
</tr>
<tr>
<td>7-7</td>
<td>OM initialization PROCLIB member CSLOI12D</td>
</tr>
<tr>
<td>7-8</td>
<td>OM startup procedure JCL</td>
</tr>
<tr>
<td>7-9</td>
<td>RM initialization PROCLIB member CSLRIRMX</td>
</tr>
<tr>
<td>7-10</td>
<td>RM startup procedure JCL</td>
</tr>
<tr>
<td>7-11</td>
<td>System Definition PROCLIB member DFSDF12D</td>
</tr>
<tr>
<td>7-12</td>
<td>FRP messages displayed upon Repository Server initialization</td>
</tr>
<tr>
<td>7-13</td>
<td>Defining an IMSRSC repository to the RS catalog repository with FRPBATCH</td>
</tr>
<tr>
<td>7-14</td>
<td>RM successfully connecting to IMSRSC repository</td>
</tr>
<tr>
<td>7-15</td>
<td>Populating an IMSRSC repository from an RDDS</td>
</tr>
<tr>
<td>7-16</td>
<td>RDDS to Repository utility (CSLURP10) job output</td>
</tr>
<tr>
<td>7-17</td>
<td>JCL to allocate a non-system RDDS</td>
</tr>
<tr>
<td>7-18</td>
<td>Populating an IMSRSC repository from MODBLKS using DFSURCM0 and CSLURP10</td>
</tr>
<tr>
<td>7-19</td>
<td>Job output from running DFSURCM0 and CSLURP10</td>
</tr>
<tr>
<td>7-20</td>
<td>Batch ADMIN LIST command to display the information of a single IMSRSC repository</td>
</tr>
<tr>
<td>7-21</td>
<td>Output for the batch ADMIN LIST REPOSITORY command</td>
</tr>
<tr>
<td>7-22</td>
<td>The batch ADMIN LIST STATUS command to display all IMSRSC repository information</td>
</tr>
<tr>
<td>7-23</td>
<td>JCL to populate a repository with stored definitions within an RDDS</td>
</tr>
<tr>
<td>7-24</td>
<td>JCL to populate an RDDS with stored definitions within an IMSRSC repository</td>
</tr>
<tr>
<td>7-25</td>
<td>JCL that executes the ADD, START, and LIST functions of the batch ADMIN utility</td>
</tr>
<tr>
<td>7-26</td>
<td>UPDATE RM command syntax</td>
</tr>
<tr>
<td>7-27</td>
<td>UPDATE RM command dynamically updating the AUDITACCESS setting</td>
</tr>
<tr>
<td>7-28</td>
<td>QUERY RM command syntax</td>
</tr>
<tr>
<td>7-29</td>
<td>Displaying the audit access setting using the QUERY RM command</td>
</tr>
<tr>
<td>7-30</td>
<td>Displaying the status of RM using the QUERY RM command</td>
</tr>
<tr>
<td>7-31</td>
<td>Displaying all information associated with RM using the QUERY RM command</td>
</tr>
<tr>
<td>7-32</td>
<td>UPDATE IMS command syntax for DRD-related functions</td>
</tr>
<tr>
<td>7-33</td>
<td>QUERY IMS command syntax</td>
</tr>
<tr>
<td>7-34</td>
<td>Showing repository attributes using the QUERY IMS command</td>
</tr>
<tr>
<td>7-35</td>
<td>Showing the automatic export setting using the QUERY IMS command</td>
</tr>
<tr>
<td>7-36</td>
<td>Syntax for displaying attribute values for resources or descriptors</td>
</tr>
<tr>
<td>7-37</td>
<td>Displaying both repository and local IMS definition information with QUERY</td>
</tr>
<tr>
<td>7-38</td>
<td>Displaying only repository IMS definition information with QUERY</td>
</tr>
<tr>
<td>7-39</td>
<td>Displaying only local IMS definition information with QUERY</td>
</tr>
<tr>
<td>7-40</td>
<td>Displaying a list of IMS systems that have a specific resource defined with QUERY</td>
</tr>
<tr>
<td>7-41</td>
<td>Displaying repository and local IMS definitions with IMSIDs using QUERY</td>
</tr>
<tr>
<td>7-42</td>
<td>EXPORT command syntax</td>
</tr>
<tr>
<td>7-43</td>
<td>Displaying the time a resource was created with QUERY and SHOW(TIMESTAMP)</td>
</tr>
<tr>
<td>7-44</td>
<td>Exporting a resource to the repository</td>
</tr>
<tr>
<td>7-45</td>
<td>Failed export attempt due to a missing program resource</td>
</tr>
<tr>
<td>7-46</td>
<td>Query transaction resource to determine associated program</td>
</tr>
<tr>
<td>7-47</td>
<td>Successful export of resources to repository</td>
</tr>
<tr>
<td>7-48</td>
<td>DELETE DEFN command syntax</td>
</tr>
<tr>
<td>7-49</td>
<td>Command sequence for deleting runtime and stored IMS resource definitions</td>
</tr>
<tr>
<td>7-50</td>
<td>IMPORT command syntax</td>
</tr>
<tr>
<td>7-51</td>
<td>Syntax for the batch ADMIN utility ADD command</td>
</tr>
<tr>
<td>7-52</td>
<td>JCL to add a user repository to the RS catalog repository</td>
</tr>
<tr>
<td>7-53</td>
<td>Syntax for the batch ADMIN utility UPDATE command</td>
</tr>
</tbody>
</table>
7-54  JCL to update an IMSRSC repository in the RS catalog repository 243
7-55  Syntax for the batch ADMIN utility RENAME command 243
7-56  JCL to stop and rename a user repository in the RS catalog repository 243
7-57  Syntax for the batch ADMIN utility DELETE command 244
7-58  JCL to delete a user repository from the RS catalog repository 244
7-59  Syntax for the batch ADMIN DSCHANGE command 244
7-60  Syntax for the batch ADMIN utility LIST command 245
7-61  JCL to show user repository details defined to the RS 245
7-62  Output from the batch ADMIN utility LIST STATUS command 245
7-63  Output from the batch ADMIN utility LIST command when a user repository is specified 246
7-64  Syntax for the batch ADMIN utility START command 246
7-65  Syntax for the batch ADMIN utility STOP command 247
7-66  Sequence showcasing each batch ADMIN command 247
7-67  Syntax for the z/OS modify interface ADMIN command 248
7-68  Output from the z/OS modify interface ADMIN,DISPLAY command 249
7-69  Syntax for the z/OS modify interface AUDIT command 249
7-70  Syntax for the z/OS modify interface SECURITY command 250
7-71  Syntax for the z/OS modify interface SHUTDOWN command 250
7-72  Using the batch ADMIN UPDATE command to designate a SAF security class 257
7-73  Making RACF definitional changes after the security class has been designated 257
7-74  Refreshing RACF in-storage profiles with z/OS modify interface SECURITY command 257
7-75  A batch ADMIN DSCHANGE command setting RDS1 to SPARE disposition 258
7-76  A z/OS modify interface command setting RDS1 to SPARE disposition 258
7-77  Restricting access to a user repository by defining a profile for it in RACF 260
7-78  Restricting access to all user repositories by defining a catch-all profile in RACF 260
7-79  Restricting access to the RS catalog repository by defining a profile for it in RACF 261
7-80  Restricting access to an individual member by defining a profile for it in RACF 261
7-81  Restricting access to modifying audit levels by defining a profile for it in RACF 261
7-82  Granting different levels of repository access to user IDs 261
7-83  Granting alter access to the ADMIN1 user ID 262
7-84  Granting update access to the USRUTL10 user ID for an individual resource 262
7-85  Granting read access to the USRUTL20 user ID for all resources 262
7-86  Granting update access to the USRZOSMI user ID to modify the audit level of REPO1 262
8-1  The MISSINGFIX command 281
8-2  Using FIXCAT as part of an APPLY command 282
8-3  XML variables in the IVP Export file 291
8-4  Additional JCL generated by the IVP file tailoring process 291
8-5  Example of output from job IV_U206J 303
9-1  Excerpt of the DBD metacode in XML format for DBD “AUTODB” 338
9-2  Excerpt of the DBD metacode in XML format for PSB “AUTPSB11” 342
9-3  PCB macro in PSBGEN corresponding to Figure 9-21 346
9-4  SOAPheader with UserNameToken 352
9-5  SOAPheader with SAML11SignedToken 354
9-6  SOAPheader with SAML 2.0 token 357
9-7  RACF definition sample 361
9-8  Changes to the server.xml file for enabling SSL port 8443 367
9-9  Starting the SOAP Gateway 367
9-10  Messages on the console for start of SOAP gateway 368
9-11  START procedure for SOAP Gateway on z/OS 368
9-12  Start of SOAP gateway server on Windows 369
## Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-1</td>
<td>/DIAG valid environments and keywords</td>
<td>31</td>
</tr>
<tr>
<td>2-2</td>
<td>The /DIAGNOSE SNAP AREA() control blocks</td>
<td>33</td>
</tr>
<tr>
<td>2-3</td>
<td>The /DIAGNOSE SNAP DB() control blocks</td>
<td>34</td>
</tr>
<tr>
<td>2-4</td>
<td>The /DIAGNOSE SNAP LINE() control blocks</td>
<td>34</td>
</tr>
<tr>
<td>2-5</td>
<td>The /DIAGNOSE SNAP LINK() control blocks</td>
<td>35</td>
</tr>
<tr>
<td>2-6</td>
<td>The /DIAGNOSE SNAP PGM() control blocks</td>
<td>36</td>
</tr>
<tr>
<td>2-7</td>
<td>The /DIAGNOSE SNAP REGION() control blocks</td>
<td>37</td>
</tr>
<tr>
<td>2-8</td>
<td>Trace tables that contain CQS trace records</td>
<td>50</td>
</tr>
<tr>
<td>2-9</td>
<td>3390 DASD volume characteristics</td>
<td>53</td>
</tr>
<tr>
<td>4-1</td>
<td>Meaning of the combinations of AOS and RRS values</td>
<td>111</td>
</tr>
<tr>
<td>4-2</td>
<td>MINVERS influence on APPC/OTMA SMQ enablement</td>
<td>114</td>
</tr>
<tr>
<td>6-1</td>
<td>IMS Connect additional recorder trace entries</td>
<td>188</td>
</tr>
<tr>
<td>6-2</td>
<td>New commands for user ID caching</td>
<td>191</td>
</tr>
<tr>
<td>6-3</td>
<td>XML converter refresh command</td>
<td>192</td>
</tr>
<tr>
<td>6-4</td>
<td>New TCP/IP MSC link statistics</td>
<td>194</td>
</tr>
<tr>
<td>6-5</td>
<td>IMS Connect type-2 display commands</td>
<td>195</td>
</tr>
<tr>
<td>6-6</td>
<td>IMS Connect type-2 start commands</td>
<td>196</td>
</tr>
<tr>
<td>6-7</td>
<td>IMS Connect type-2 stop or close commands</td>
<td>197</td>
</tr>
<tr>
<td>6-8</td>
<td>IMS Connect type-2 set, reset, and refresh commands</td>
<td>198</td>
</tr>
<tr>
<td>7-1</td>
<td>New type-2 command RACF definitions for OM</td>
<td>240</td>
</tr>
<tr>
<td>7-2</td>
<td>Batch ADMIN utility commands and their functions</td>
<td>241</td>
</tr>
<tr>
<td>7-3</td>
<td>Summary of RS command functions</td>
<td>248</td>
</tr>
<tr>
<td>7-4</td>
<td>Comparison of the batch ADMIN utility commands to the RS commands</td>
<td>250</td>
</tr>
<tr>
<td>7-5</td>
<td>Potential results for IMPORT DEFN command when OPTION(UPDATE) is specified</td>
<td>252</td>
</tr>
<tr>
<td>8-1</td>
<td>IMS 12 PSP Upgrade and Subset ID</td>
<td>280</td>
</tr>
<tr>
<td>8-2</td>
<td>IMS 12 component IDs</td>
<td>281</td>
</tr>
<tr>
<td>8-3</td>
<td>New or changed log records for IMS 12</td>
<td>285</td>
</tr>
<tr>
<td>8-4</td>
<td>IVP naming convention for jobs and tasks</td>
<td>301</td>
</tr>
<tr>
<td>8-5</td>
<td>IMS 10 and IMS 12 file system paths changes</td>
<td>318</td>
</tr>
<tr>
<td>9-1</td>
<td>IMS environments</td>
<td>325</td>
</tr>
<tr>
<td>9-2</td>
<td>Differences between Enterprise Suite V1.1 and V2.1</td>
<td>326</td>
</tr>
<tr>
<td>10-1</td>
<td>Fast Path Solution Pack maintenance to support IMS 12</td>
<td>391</td>
</tr>
<tr>
<td>10-2</td>
<td>Database Solution Pack maintenance to support IMS12</td>
<td>394</td>
</tr>
<tr>
<td>10-3</td>
<td>Recovery Solution Pack maintenance to support IMS12</td>
<td>399</td>
</tr>
<tr>
<td>10-4</td>
<td>Performance Solution Pack maintenance to support IMS12</td>
<td>402</td>
</tr>
<tr>
<td>10-5</td>
<td>Fast Path Solution Pack utilities</td>
<td>410</td>
</tr>
<tr>
<td>B-1</td>
<td>IMS 12 current function-related APARs</td>
<td>455</td>
</tr>
</tbody>
</table>
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 websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites 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 illustrate 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.
Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at http://www.ibm.com/legal/copytrade.shtml

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

- BookManager®
- CICS®
- DB2®
- Distributed Relational Database Architecture™
- DRDA®
- DS8000®
- ECKD™
- Enterprise Storage Server®
- Enterprise Workload Manager™
- FlashCopy®
- GDPS®
- Geographically Dispersed Parallel Sysplex™
- IBM®
- IMSTM
- Language Environment®
- MVS™
- OMEGAMON®
- Parallel Sysplex®
- ProductPac®
- RACF®
- Rational®
- Redbooks®
- Redbooks (logo) ®
- RETAIN®
- RMF™
- S/390®
- System Storage®
- System z9®
- System z®
- SystemPac®
- TotalStorage®
- VTAM®
- WebSphere®
- z/Architecture®
- z/OS®
- z/VM®
- z9®

The following terms are trademarks of other companies:

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

Java, and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

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

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries 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, or service names may be trademarks or service marks of others.
Preface

IBM® Information Management System (IMS™) provides leadership in performance, reliability, and security to help you implement the most strategic and critical enterprise applications. IMS also keeps pace with the IT industry. IMS, Enterprise Suite 2.1, and IMS Tools continue to evolve to provide value and meet the needs of enterprise customers.

With IMS 12, integration and open access improvements provide flexibility and support business growth requirements. Manageability enhancements help optimize system staff productivity by improving ease of use and autonomic computing facilities and by providing increased availability. Scalability improvements have been made to the well-known performance, efficiency, availability, and resilience of IMS by using 64-bit storage.

IBM IMS Enterprise Suite for z/OS® V2.1 components enhance the use of IMS applications and data. In this release, components (either orderable or downloaded from the web) deliver innovative new capabilities for your IMS environment. They enhance connectivity, expand application development, extend standards and tools for a service-oriented architecture (SOA), ease installation, and provide simplified interfaces.

This IBM Redbooks® publication explores the new features of IMS 12 and Enterprise Suite 2.1 and provides an overview of the IMS tools. In addition, this book highlights the major new functions and facilitates database administrators in their planning for installation and migration.

The team who wrote this book

This book was produced by a team of specialists from around the world working at the International Technical Support Organization (ITSO), San Jose Center.

Paolo Bruni is an ITSO Project Leader specializing in IMS with the ITSO and is based in the Silicon Valley Lab in San Jose, CA. Since 1998, Paolo has authored Redbooks publications about IMS, IBM DB2® for z/OS, and related tools and has conducted workshops worldwide. During his many years with IBM in development and in the field, Paulo’s work has been mostly related to database systems.

Isabelle Bruneel is a Technical Sales Specialist in IMS and IMS Tools with IBM Software Group in France. She has 25 years of experience in IT, providing customer services, product support, and education for IBM Global Services in z/VM®, DB2 for z/OS, and IMS. Before joining IBM, she worked for 6 years as a consultant at a large French bank with a large workload in shared queues. From this experience, she gained a deep understanding of customer expectations in high availability environments. She is now involved in several projects that include IMS tools and IMS modernization with SOAP Gateway.

Angie Greenhaw is an IT Specialist in the IBM IMS Advanced Technical Skills group. She is a primary resource in the areas of IMS security, dynamic resource definition, Common Service Layer, and Online Change. Previously, she worked in IMS development, specializing in the Online Change function, contributing to new IMS functionality, and devising solutions as a Level 3 Service Representative. She also spent three years as the IMS Development Representative for SHARE. Angie has written a white paper about global online change implementation and has coauthored three Redbooks publications. Angie joined IBM in 2000 after receiving a bachelor degree in Computer Information Systems from Arizona State University.
Dougie Lawson is a senior software specialist in IMS with IBM Global Services in the United Kingdom. He works in software support for the UK and Ireland and with the European software support teams. He has 30 years of experience in the IT field and 29 years working with IMS. His areas of expertise include IMS, DB2, Linux, and z/OS. Before joining IBM in 1994, Dougie worked as a systems programmer for a large bank in the UK, where he was responsible for IMS and DB2 systems and related products.

Jorge Alberto Luz Ribeiro is an IT specialist in IMS with IBM Global Services in Brazil. He joined IBM in 2009 after providing IT services for some of the largest national and multinational companies in Brazil. He has more than 30 years of experience in the IT field with expertise in support for IMS, IBM CICS®, and z/OS software, in system application development, and in software quality assurance. Jorge is a Six Sigma Black Belt certified professional. He holds a Master of Business Administration degree from the University of São Paulo and a Finance specialization from Methodist University of Piracicaba (UNIMEP) in Brazil.

Egide Van Aershot is a contractor for Zinteg C.V., where he teaches classes about IBM System z® WebSphere® Application Server and WebSphere MQ in Northern Europe. His technical experience began in 1967 when he joined IBM and became responsible for many computer installations related to teleprocessing and database management in Belgium. In 1997, he moved from IBM Belgium to IBM France, where he worked as an architect and consultant at the IBM Program Support Center in Montpellier. Since 1997, he has specialized in Java, SOA, IMS, and WebSphere applications, mainly on z/OS systems, and has participated in many projects related to the Internet. Egide is co-owner of the patent “Methods, systems, program product for transferring program code between computer processes.” Egide holds an engineering degree in Electricity and Nuclear Physics from the University of Leuven, Belgium.

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

Rich Conway
Bob Haimowitz
Emma Jacobs
IBM ITSO
Carlos Alvarado
Jim Bahls
John Barmettler
Marilyn Basanta
Tom Bridges
John Butterweck
Dave Cameron
Kyle Charlet
Cedric Chen
Himakar Chennapragada
Nathan Church
Demetrios Dimatos
Bach Doan
Jeffrey Fontaine
Shirley Gaw
David Hanson
Bill Huynh
Barbara Klein
Terry Krein
Janet Leblanc
Rose Levin
Now you can become a published author, too!

Here is an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

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 books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:
  
  ibm.com/redbooks

- Send your comments in an email to:
  
  redbooks@us.ibm.com

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400

Stay connected to IBM Redbooks

- Find us on Facebook:
  
  http://www.facebook.com/IBMRedbooks

- Follow us on Twitter:
  
  http://twitter.com/ibmredbooks

- Look for us on LinkedIn:
  
  http://www.linkedin.com/groups?home=&gid=2130806

- Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:
  

- Stay current on recent Redbooks publications with RSS Feeds:
  
  http://www.redbooks.ibm.com/rss.html
IBM Information Management System (IMS) 12 provides major enhancements with many new and useful features that improve integration, openness, manageability, and scalability. These enhancements can help address your On Demand Business needs.

With IMS 12, IBM offers enhancements to both major components (transaction manager and database manager) and to other operational areas, to ensure you have the growth, availability, and systems management that newer environments and cost measures require.

This chapter highlights the following major enhancements in IMS 12:

- IMS product positioning
- System enhancements
- Transaction manager and connectivity enhancements
- Database Manager enhancements
- DBRC enhancements
- Main enhancements from IMS 10 to IMS 11

You can find more details about each of these areas in the other chapters of this book.
1.1 IMS product positioning

IMS, the recognized database manager and transaction server in the industry, is critical to your business. Exclusive to the z/OS platform, IMS 12 complements IBM DB2 and WebSphere servers for database and transaction management. All of these are strategic IBM products and continue to be enhanced.

IMS Database Manager uses hierarchical organization technology. DB2 and some other systems use relational database technology. Hierarchical and relational databases have each continued to grow, with their specific characteristics and different roles to play. Hierarchical is best used for mission-critical work and for work that requires the utmost in performance. Relational is best used for decision support.

Hierarchical databases can offer a significant performance edge over relational databases when queries are known beforehand. Query optimization for relational databases ensures good performance where the query is not known in advance. Each type is best at what it does. The products supporting these technologies are enhanced to address different application requirements, and they continue to overlap more and more in their capabilities. However, the product originally designed for a particular capability will inherently be the best.

Relational and hierarchical technologies can work together for optimum solutions. Users can efficiently store operational data in hierarchical form, which can be accessed easily by their favorite relational decision support tools, with minimal impact on the production hierarchical data. IMS data can be accessed directly or propagated and replicated with relational data for summarizing, enhancing, and mining. IBM provides standard application interfaces for accessing IMS and other data. Both relational and hierarchical IMS data can be most efficiently accessed, together or independently, using the IMS Transaction Manager (TM) and WebSphere servers.

IMS Transaction Manager and WebSphere servers are both strategic application managers, and they are enhanced to take advantage of each other. They each have inherently different characteristics. IMS is more efficient in application management, data storage, and data access but applies strict rules for this access. WebSphere makes it easier to serve the web and integrate data that might have been less defined in advance. Thus, they play different roles in the enterprise. Clients use both application managers: WebSphere for newer web-based applications, and IMS for more mission-critical high performance and availability and low-cost and transactional applications and data.

IMS and WebSphere products together have been providing tools to make this combination an optimum environment. Using IMS Connect and IMS TM Resource Adapter, WebSphere development tooling can develop web applications that can serve the web and easily access existing and new critical IMS applications. Using JDBC and IMS Open Database Access, WebSphere applications can also access IMS DB data directly.

IBM demonstrates its commitments to you by continuing to enhance IMS, as highlighted in the following sections.

1.2 System enhancements

This section describes the changes to the IMS system definition and execution parameters that affect functions of the IMS system itself. Definition and execution changes for new functions are provided where the new functions are explained in detail.
Repository
With IMS 12, users of IMS DRD can now use the repository to manage MODBLKS resources in a single location. From this centralized data set, the resource definitions can be stored and retrieved by different IMS systems existing together in an IMSplex by using type-2 commands. They can also be shared among the systems and can be dynamically updated.

The main goal of the IMS repository is to simplify resource management by eliminating the need for multiple resource definition data sets (RDDS) for each IMS system. All IMS systems in the IMSplex share the same IMS repository for their resource definitions. These IMS systems can be cloned or non-cloned. With the IMS repository, users can maintain different attributes for the same resource name for each IMS in the IMSplex.

If you have never implemented DRD, you can use the definitions that exist in your MODBLKS data set as a starting point, eventually porting them to the new repository. Much like the RDDS, a repository contains resource definitions for database, program, routing code and transaction resources and descriptors.

IMPORT UPDATE
IMS 12 brings the new UPDATE option for the type-2 IMPORT command. This enables users to change runtime resources and descriptors with the attributes from an imported definition. If no runtime resource or descriptor exists when the IMPORT OPTION(UPDATE) command is issued, IMS creates the resource or descriptor with the attributes from the imported definition. For the update of an existing runtime resource or descriptor to succeed, the resource or descriptor must not be in use.

LOGGER
IMS 12 improves the functionality of IMS logging by adding Extended Format support for online data sets (OLDS) and system log data sets (SLDS), allowing them to be striped and providing the buffer allocation beyond the 31-bit, among other improvements.

Also, IMS 12 changes the way that write-ahead data set (WADS) writes are done. The concept of track groups is not used with IMS 12. This changes the calculation for the space required for the WADS and changes the data written by log ahead requests. The benefit of this enhancement is to increase logging rates while releasing storage in the extended common service area (ECSA) for other users.

EAV support of non-VSAM data sets
z/OS Version 1 Release 12 supports non-Virtual Storage Access Method (VSAM) data sets residing in the extended addressing space (EAS). IMS 12 adds the ability for IMS to address specific non-VSAM data sets residing in the EAS of an extended address volume (EAV) volume.

With z/OS 1.12, two new data set types allow applications such as IMS to recognize when data sets are located in the EAS of an EAV. Data sets in EAS are addressed using a new 28-bit cylinder/track addressing format. With EAV volumes, it is now possible to have more than 65 K cylinders, and specific IMS overflow sequential access method (OSAM) data sets can benefit from this capability. Support was added for VSAM data sets to reside in EAS in IMS 11.

The /DIAGNOSE command
IMS 12 improves the reliability of diagnostic information by supporting additional control blocks for the /DIAG SNAP command. The /DIAGNOSE command enhancements address the need to improve the reliability of diagnostic information and to streamline the problem
determination process. This line item component enhances the /DIAGNOSE command SNAP function in the following ways:

- Six new SNAP resource types (AREA, DB, LINE, LINK, PGM, and REGION), which support multiple resource name parameters, have been added.
- A display output option has been added.
- An output filtering option has been provided.

The overall benefits of these enhancements are improved problem diagnosis and resolution.

**Common Queue Server trace facilities**

Before IMS 12, all Common Queue Server (CQS) structure events were contained in a single trace table. IMS 12 adds two new trace entries. These two new trace event tables are automatically generated by IMS to contain structure event trace entries and structure overflow trace entries. These new tables retain critical trace entries for a longer period of time, which improves CQS serviceability.

All existing trace entries that are not related to client activity are moved to one of the new trace tables and are expanded to benefit from the larger 64-byte trace entry length. The existing CQS structure trace table (STR) now contains only client activity trace entries. The benefit of these separate traces is to provide the IMS support team with valuable diagnostic data for client problem resolution, with the possibility that shared queues-related problems might be resolved faster.

### 1.3 Transaction manager and connectivity enhancements

This section explains the changes to the IMS system definition and execution parameters that affect functions of the IMS TM and connectivity. Definition and execution changes for new functions are provided where the new functions are explained in detail.

**Asynchronous IMS-to-IMS TCP/IP messages**

IMS 12 provides the ability to send messages from a local IMS to a remote IMS over a TCP/IP network. The messages are sent over the TCP/IP network by a local IMS Connect and received by a remote IMS Connect. The local IMS communicates with the local IMS Connect, and the remote IMS communicates with the remote IMS Connect. IMS can send messages using Open Transaction Manager Access (OTMA) or Multiple Systems Coupling (MSC) protocols. For OTMA messages, one-way asynchronous messaging is supported. IMS OTMA communicates with IMS Connect using cross-system coupling facility (XCF).

**MSC TCP/IP links**

Environments that primarily use TCP/IP networks for communications, but require SNA or IBM VTAM® to sustain their MSC links, now have the option of also migrating their MSC connections entirely to TCP/IP. A new TCPIP link type has been added to the MSPLINK macro. These new links can be used to replace, complement, or back up existing MSC VTAM connections. Operationally, the new support is compatible with other link types, such as CTC, MTM, and VTAM, by using the same command structure.

From an MSC perspective, the operation of TCP/IP physical links and VTAM physical links is similar. Depending on various factors such as network traffic and the distance between the two connected IMS systems, a TCP/IP physical link is likely to provide better performance than a VTAM physical link.
IMS Connect type-2 command
The IMS Connect type-2 command line item introduces new type-2 commands for IMS Connect resources into Version 12. These type-2 commands are equivalents and, in some cases, improvements to the existing write to operator with reply (WTOR) and z/OS commands available to IMS Connect commands. With these commands, you can efficiently manage IMS Connect resources in an IMSplex and view consolidated command output. These new commands can be issued from the Operations Manager (OM) API.

OTMA ACEE reduction
IMS 12 also introduces a new maximum aging value for accessor environment elements (ACEEs). The ACEE aging value triggers when an ACEE is refreshed. This value is specified by the OTMA member client during the client-bid process. The maximum has been reduced from 68 years to 11.5 days. Each cached OTMA ACEE will have an aging value based on the OTMA member client using it that has specified the lowest number. This enhancement is important because some OTMA clients, such as the DB2 Stored Procedure DSNAIMS, set ‘7FFFFFFF’X seconds (68 years) as the ACEE aging value for users sending IMS transactions and commands. The enhancement also detects obsolete ACEEs, thereby improving security.

LU 6.2 Edit exit routine (DFSLUEE0)
The LU 6.2 Edit exit routine (DFSLUEE0) enables you to edit input and output LU 6.2 messages for IMS-managed LU 6.2 conversations. It is also called if a message is inserted from an alternate PCB destined for an LU 6.2 destination. IMS 12 adds to the LU6.2 Input/Output Edit Exit (DFSLUEE0) supports a new return code (RC=2) that requests the dequeuing of an undeliverable asynchronous output message. Previously, IMS requeued the message. The new return code is only valid for asynchronous conversations.

APPC and OTMA shared queues
IMS 12 supports the execution of APPC synchronous conversations and OTMA Commit Mode 1 (CM1) or Send-then-Commit transactions with sync level of NONE or CONFIRM in a shared queues (SQ) environment using XCF communications, eliminating the need for Resource Recovery Services (RRS). The use of Synclevel=syncpoint still requires RRS.

1.4 Database Manager enhancements
IMS 12 provides several improvements for traditional databases, high availability large databases (HALDB), and Fast Path databases.

Dynamic full function database buffer pools
With IMS 12, users can add, change, and delete full function buffer pools. With this support, full function buffer pools can be managed without restarting IMS because IMS can internally quiesce application read and update activity so that the UPDATE POOL command can complete with little disruption to transaction workloads

Parallel migration to HALDB
The IMS Unload utility (DFSURGU0) has been enhanced to allow the unload of key ranges of an hierarchical direct access method (HDAM) or hierarchical indexed direct access method (HIDAM) database when migrating to HALDB. Multiple unloads for the same database can be run in parallel. This can significantly reduce the elapsed time for a migration to HALDB.
Reorganization of number handling by timestamp recovery

In IMS 12, you do not have to run the DFSPREC0 utility to rebuild an ILDS. The reorganization number in the data set matches the number in recovery control (RECON) when timestamp recovery is done. Users can avoid running the HALDB Index/ILDS Rebuild utility (DFSPREC0) to rebuild the ILDS after recovering a database that has a secondary index when reorganization verification is enabled.

Previous IMS versions did not coordinate the numbers in RECON and the partition data set.

- The reorganization number in the data set was updated from the RECON value by the first IMS subsystem that updated the partition.
- The Index Builder tool created index pointers based on the reorganization number in the data set.
- Index entries needed healing when the reorganization number was changed by the updater.

The IMS 12 Database Recovery utility sets the reorganization number of the partition based on the value in RECON.

Fast Path DEDB secondary index support

In IMS 12, you can have a secondary index on a primary data entry database (DEDB) to process a segment type in a sequence other than the one defined by the segment's key. With secondary indexing, you can have an index based on any field in the database, not simply the key field in the root segment. Secondary indices were previously only available for full-function and HALDB databases.

Control Area reclaim support

With the Control Area (CA) Reclaim feature in z/OS 1.12, free CA space can be reused. With CA Reclaim, space fragmentation caused by erasing records from a KSDS will be minimized to reduce the need to reorganize the data set. When the freed CAs are placed in a free chain to be reused, the index structure can be shrunk to facilitate quicker data accesses. CA Reclaim is available for all current IMS releases that use z/OS 1.12.

ACB library enhancements (already in IMS 11)

The application control blocks library (ACBLIB) usability enhancements enable you to cache the application control block (ACB) members into 64-bit storage. Separately, you have to create DFSMDA members for the dynamic allocation of the ACBLIB data sets.

Database pool storage

With IMS 12, you can specify the initial amount of 64-bit storage used for the buffer pool for a DEDB buffer pool in the DBCTL environment. In IMS 12, the storage for certain database pools is now obtained in 31-bit virtual storage backed by 64-bit real storage. IMS 12 expands VSAM full function database buffer pools to be defined for an IMS online system, batch job, or utility to 255.

Lock timeout message and logging

IMS 12 adds optional DFS2291I diagnostic messages for lock timeouts and writes log record x’67D0’ subtype x’1B’ for lock timeouts. If the wait for a lock exceeds the IMS LOCKTIME value when using the internal resource lock manager (IRLM), the lock request is rejected and the waiter is abended with a U3310 or a ‘BD’ status code is returned to the program.
**Batch data sharing abend elimination**
In previous versions of IMS, a batch data sharing job abended with a U3303 when an OSAM or VSAM cache structure access failed. In IMS 12, batch jobs survive when these structure accesses fail. Like online systems, they wait for the resolution to the problem. When the problem is resolved, the batch jobs continue processing. For example, when a rebuild of a structure completes, the batch jobs continue.

**Reduced logging for DEDB changed data capture**
Some users want to use asynchronous changed data capture, but they do not want to write log records for before images. Prior to IMS 12, asynchronous changed data capture writes before and after image log records (x'99'). By using IMS 12, you can specify that these before image log records are not to be written.

**Elimination of OSAM U0080 Open/Close/EOV abends**
Previous versions of IMS had rare problems in OSAM open, close, or end-of-volume processing. When they occurred, the entire IMS system terminated with the U0080 abend. IMS 12 has changed this processing. When such problems occur, the database is closed and marked “recovery needed.” The abend does not occur. Additionally, message DFS0730I is issued for open or close problems.

**Optional Release of HALDB OLR**
IMS 12 provides an option for the release of ownership of a HALDB Online Reorganization when the IMS system on which it is executing terminates. The termination can be either a normal termination or abnormal termination of IMS. If ownership is released, the OLR can be restarted on another IMS system. If ownership is not released, the OLR cannot be restarted on another IMS system.

### 1.5 DBRC enhancements

This section outlines the enhancements for Database Recovery Control (DBRC) for IMS 12.

**New CLEANUP command parameters**
IMS 11 introduced the `CLEANUP.RECON` command to delete old or expired recovery-related information from the RECON data set, including image copy, allocation, reorganization, and recovery records as well the log information. IMS 12 adds deletion of change accumulation (CA) execution records information by the new CAGRANGE, CAONLY, and LASTCA keywords.

**New user information**
To allow clients to add their own control information into some of the important DBRC RECON records related to the data base administration role, you can use IMS 12 to add user data into IC, CA, RECOV, and REORG RECON records by the UDATA optional keyword. The UDATA ('string') is an optional keyword that you use to specify up to 80 bytes of information about the identified record. You can use the variable field of this keyword to describe how the data set was created. The string value must be enclosed in single quotation marks (').

**New change accumulation retention period**
Earlier versions of IMS use the GRPMAX parameter to control how many versions of CA execution are to be kept in RECON data sets. When the number of times you run the Change Accumulation utility for the specified group exceeds the GRPMAX value, the record with the earliest CA stop time for the group is deleted for CA groups. To improve the control of relevant
information stored in the CA group record, IMS 12 provides the RECOVPD optional keyword, which can be used to set the recovery period for a specified CA group records. The recovery period is the amount of time before the current date for which DBRC maintains CA information in the RECON data set.

**GENJCL enhancements**

DBRC uses, along another elements, partitioned data set (PDS) members as templates or skeletal job control language (JCL) to correctly run recovery utilities. Usually, you modify the skeletal JCL to reflect your installation's system configuration.

When you issue a GENJCL command, it uses a skeletal JCL execution member, which contains symbolic keywords. You can define your own symbolic keywords and use the symbolic keywords that already exist. DBRC substitutes current information for the symbolic keywords.

IMS 12 increases the number of user keys in skeletal JCL from 32 to 64, keeping the same conventions and restrictions applied to earlier versions. In addition, the existing %DBTYPE key can be used when selecting database data sets (DBDS) allocation.

**LIST command enhancements**

The most common use of the LIST command is to produce a formatted printout of all or selected information contained in a RECON data set. Normally, we use the information printed to make an analysis and then take a specific course of action regarding DBRC-controlled resources.

IMS 12 supports an output larger than 32 K for the /RMLIST online command, but only when the command is entered through OM API. The output size is restricted by the DBRC private storage available for buffering the output message or OM limitations.

IMS 12 provides the NORCVINF keyword for LIST.DB and LIST.DBDS commands. This keyword suppresses recovery-related information. ALLOC, IC, RECOV, and REORG records are not listed, which reduces command output.

IMS 12 adds full precision timestamps and more information about the HALDB type of databases such as active DBDS, DDNames of inactive DBDS, and the current reorganization number for the partition in the LIST.HISTORY command output.

IMS 12 includes the number of registered databases in the LIST.RECON output so that users can see whether they are nearing the 32, 767 limit of registered databases.

### 1.6 Main enhancements from IMS 10 to IMS 11

This section summarizes the main enhancements in IMS 11, which are new to you if you are migrating from IMS 10. For information about other enhancements that are not explained in this book, see *IMS Version 11 Technical Overview*, SG24-7807.

**IMS Connect**

The biggest change is for Open Database. IMS Connect becomes the entry point for IBM Distributed Relational Database Architecture™ (IBM DRDA®) requests for access to IMS databases. This means that IMS Connect becomes important also for database-only users of IMS. Two Java drivers, named the IMS Universal drivers, provide Open Database Access (ODBA) either directly using ODBA (if running on the same LPAR) or using DRDA with IMS Connect (if not running on the same LPAR). The drivers handle the various protocols so that the programmer does not have to.
IMS Enterprise Suite
Among other features, the Enterprise Suite contains a new version of the DLIModel utility supporting PL/I structures and a number of other enhancements. It also offers IMS Connect APIs to help programmers more easily communicate directly with IMS Connect, and a JMS API for application programs running in a Java-dependent region that want to make synchronous callout requests.

Open Database
The main change introduced by IMS Open Database is the ability to easily access IMS databases from platforms other than z/OS. This access is through TCP/IP by using the DRDA protocol; applications are shielded from the complexity of DRDA by a new set of Java drivers, known as IMS Universal drivers.

User exits
IMS 11 introduces three user exits:
- An initialization/termination exit (invoked at IMS initialization and termination)
- An IMS Common Queue Server (CQS) event exit (invoked when IMS receives a CQS event)
- An IMS CQS structure event exit (invoked when IMS receives a CQS structure event)

Dynamic allocation of ACBLIB
With IMS 11 you can modify the inactive ACBLIB concatenation while IMS is running. This allows you to compress ACB libraries, change the concatenation sequence, and add or remove ACB libraries. This improvement is particularly useful in an IMSplex with Global Online Change, because earlier versions of IMS required an IMSplex outage for such changes.

DBRC
Using IMS 11, you can benefit from Base Primitive Environment (BPE) features for your existing DBRC user exits, such as refreshing them dynamically. You can also collect DBRC performance data by writing a new (or modifying an existing) BPE statistics exit. If you have implemented RECON security, IMS 11 enables you to more easily use copies of the RECON that you might have made for recovery or problem diagnosis. The CLEANUP.RECON new command helps you remove obsolete data from the RECONs, which can be tedious to remove by using other means.

IMS commands
IMS 11 introduced four type-2 commands that help you to manage your OTMA environment. These commands provide information about message volumes and associated transaction instances, so that you can take action to prevent potential problems from arising because of high-message volumes OTMA enhancements.

OTMA
IMS 11 introduced OTMA capabilities that improve consistency in the IMS shared queues environment, reduce overall transaction processing costs, and increase resiliency when certain conditions arise that, if left untreated, can result in delays.

Fast Path 64-bit buffer manager
You can choose to store DEDB data buffers in 64-bit storage. If you do this, the new Fast Path 64-bit buffer manager will manage the buffer pool autonomically for you up to the size you originally specified.
**ACB caching**

IMS 11 introduced the possibility to cache ACB library members in 64-bit storage. This enhancement does not replace the existing program specification block (PSB) and data management block (DMB) pools. Instead, it acts as a buffer for the ACBLIB, reducing ACBLIB I/O for often-used members because these are found in the cache.
System enhancements

This chapter describes the major enhancements for the IMS 12 system, including descriptions of the following main components:

- TSO SPOC
- Dynamic resource definition
- ACB online change
- The /DIAGNOSE command
- IMS logger
- CQS trace facilities
- Extended address volume
- Buffer pools allocation
- Miscellaneous other enhancements
2.1 TSO SPOC

A single point of control (SPOC) is a place where you can handle the operations of all IMS systems within an IMSplex.

IMS 12 adds support to the IMS repository function and enhances the `UPDATE` option of the `IMPORT DEFN` command. The associated benefit simplifies the need to manually coordinate individual resource definition across the IMSplex.

2.1.1 SPOC background information

A SPOC is a collection of programming elements that helps you to handle operations of all IMS systems within an IMSplex instead of using a master terminal for each one. By using a SPOC, you can issue commands to all participant members of an IMSplex at once.

The Time Sharing Option (TSO) SPOC application provides a set of resource management panels that you can use to query IMS resources. Additionally, if dynamic resource definition (DRD) is enabled in your IMS systems, you can also use the TSO SPOC panels to create, delete, export, import, and update IMS resources. You can access the TSO SPOC panels directly from the IMS Application Menu (Figure 2-1).

![IMS Application Menu panel](image)

The TSO SPOC communicates with a single Operations Manager (OM), which in an IMSplex is a Common Service Layer (CSL) component that provides an application programming interface (API) for automated operator programs (AOPs). Through the Structured Call Interface (SCI), which is a CSL component that manages communications between the IMSplex members, OM then communicates with all of the other IMS control regions in the IMSplex. TSO SPOC communicates with a single OM.
Figure 2-2 shows the SPOC architecture.

![Figure 2-2: The SPOC architecture](image)

You can issue both IMS type-1 commands and type-2 commands by using the TSO SPOC interface. You can issue commands in the IMS TSO SPOC application in the following ways:

- By using the command line
- By retrieving a command
- By defining and using command shortcuts

Figure 2-3 shows the SPOC first panel.

![Figure 2-3: TSO SPOC initial panel](image)

### 2.1.2 SPOC description

The existing command interfaces for the write to operator with reply (WTOR), master terminal operator (MTO), and extended multiple console support (EMCS) console are supported only for type-1 commands. If you want to use type-2 commands, you must use a SPOC. You can use a SPOC instead of the z/OS master console, the IMS master terminal, or a user terminal for most command processing.
The following types of SPOCs are available:

- On a workstation, the SPOC is an IMS Control Center with a graphical user interface (GUI).
- On a 3270 TSO terminal, a TSO SPOC is an application with an Interactive System Productivity Facility (ISPF) panel interface.
- The batch SPOC utility uses OM services to submit IMS operator commands to an IMSplex throughout standard job control language (JCL) statements.
- With the REXX SPOC API, automation programs can use SPOC functions.
- A vendor or user-written SPOC is a program that uses or accesses the OM API to perform SPOC functions.

More than one type of SPOC can be in an IMSplex, and any number of SPOCs can be active.

The TSO SPOC provides the following functions to an IMSplex:

- It presents a single system image for an IMSplex by allowing the user to issue commands to all IMS systems in the IMSplex.
- It displays consolidated command responses from multiple IMS address spaces.
- It sends a message to an IMS terminal connected to any IMS control region in the IMSplex by using the IMS /BROADCAST command.
- It helps users to create, query, update, and delete various IMS resources online.
- It displays the OM audit trail log, which records command input, command responses, and selected system messages from across the IMSplex.
- It helps users to define input user exits to modify or reject command parameters.

### 2.1.3 SPOC enhancements with IMS 12

The TSO SPOC application is enhanced in IMS 12 to support the use of the IMS repository function and the UPDATE option of the IMPORT DEFN command.


### 2.1.4 Using TSO SPOC

This section explains the QUERY and the IMPORT DEFN commands under TSO SPOC.

**QUERY commands**

You use the IMS QUERY commands to display information about IMS resources. The QUERY commands are type-2 commands and return information based on the keyword specified.
Figure 2-4 shows an example of the `QUERY DBDESC` command being issued through TSO SPOC.

![Example of `QUERY DBDESC` command](image)

**IMPORT UPDATE commands**

Before you create, delete, export, import, or update IMS resources through TSO SPOC by using the dynamic resource definition commands, be aware of the following points:

- You must enable DRD in the target IMS systems.
- Changes made to IMS resources are not saved across an IMS cold start.

You can create, delete, export, import, and update the following IMS resources and resource descriptors in DRD-enabled systems:

- Application programs
- Databases
- Fast Path routing codes
- Transactions

For more information about DRD, see 2.2, “Dynamic resource definition” on page 17.

You can use the **IMPORT** command to create resource and descriptor definitions or replace existing resource and descriptor definitions in an online IMS system by using the definitions in a resource definition data set (RDDS) or the IMSRSC repository. The **IMPORT** command can be issued through TSO SPOC or the Manage Resources options in the IMS Applications menu. This command can also be issued to an IMSplex by using the batch SPOC utility.

The **UPDATE** option indicates that if the definition being imported is for a resource or descriptor that already exists in IMS, the imported definition should be used to replace the existing runtime resource or descriptor definition. If the definition being imported is for a resource or descriptor that does not exist, the imported definition should be used to create the runtime resource or descriptor definition. If the **UPDATE** option is not specified and a runtime definition already exists for the resource or descriptor, the import of the resource or descriptor definition fails.
In most cases, the affected resource must not be in use when the **IMPORT OPTION(UPDATE)** command is entered. If the resource is in use, the import of the stored definition fails. The following transaction attributes can be updated if the transaction is in use: CLASS, LCT, LPRI, NPRI, MAXRGN, PARLIM, PLCT, PLCTTIME, SEGNO, SEGSZ, and TRANSTAT. The TRANSTAT program attribute can be updated while the program is in use.

To minimize the likelihood of the import of a resource definition failing, complete the following steps before issuing the **IMPORT** command:

1. Stop the resource.
2. Query the resource to check for work in progress.
3. Complete the work, if any.

If the imported definition is for a resource or descriptor that already exists in IMS, the import time stamp in the existing runtime definition is replaced with the time the **IMPORT** command was received by OM. If one or more of the attributes in the existing runtime definition are different from the attributes in the imported definition, the update time stamp is also updated with the time the **IMPORT** command was received by OM. The access and create time stamps in the existing runtime definition are unchanged.

If the imported definition is for a resource or descriptor that does not exist in IMS, the import time stamp in the newly created runtime definition is set to the time the **IMPORT** command was received by OM. The create time stamp is obtained from the imported definition and stored in the new runtime definition.

If the imported definition is for a descriptor that has **DEFAULT(N)** defined and the runtime descriptor is the current default descriptor, the default value is not updated. The runtime descriptor remains the default descriptor. Other attributes are updated, but the default value remains unchanged. To change the default descriptor so that it is no longer the default descriptor, you must update another descriptor to be the default descriptor. If the imported definition has **DEFAULT(Y)** defined, the updated runtime descriptor becomes the current default descriptor. The **DEFNTYPE** of a newly created definition is set to **IMPORT**. The **DEFNTYPE** is set to **IMPORT** when an existing definition is replaced with a new definition.

Figure 2-5 shows the **IMPORT UPDATE** option through TSO SPOC. In this example, the resource already exists.

<table>
<thead>
<tr>
<th>File</th>
<th>Action</th>
<th>Manage resources</th>
<th>SPOC</th>
<th>View</th>
<th>Options</th>
<th>Help</th>
</tr>
</thead>
<tbody>
<tr>
<td>IM12X</td>
<td>IMS</td>
<td>Single Point of Control</td>
<td>Command ===&gt; IMPORTDEFNSOURCE(RDDS) NAME(IVPDB1) RDDSDSN(IMS12Q.RRDS) OPTION(UPDATE)</td>
<td>Plex .</td>
<td>Route .</td>
<td>Wait .</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Response for: <strong>IMPORT DEFN SOURCE(RDDS) NAME(IVPDB1) RDDSDSN(IM...</strong></td>
<td>Name</td>
<td>Type</td>
<td>MbrName</td>
</tr>
<tr>
<td>IVPDB1</td>
<td>DB</td>
<td>I12A</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IVPDB1</td>
<td>DB</td>
<td>I12B</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IVPDB1</td>
<td>DB</td>
<td>I12D</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 2-5** The **IMPORT UPDATE** command
2.1.5 Considerations for using the IMPORT command

The **IMPORT** command imports resource and descriptor definitions from an RDDS or the repository and uses the definitions to create the runtime resource and descriptor definitions in the online IMS system. Therefore, the same rules and restrictions that apply to the **CREATE** commands for databases, programs, routing codes, and transactions also apply to the **IMPORT** command.

- The **IMPORT** command can be issued only through the OM API.
- This **IMPORT** command applies to DB/DC, Database Control (DBCTL), and Data Communications Control (DCCTL) systems.
- The **IMPORT** command is not valid on the extended recovery facility (XRF) alternate, Remote Site Recovery (RSR) tracker, or Fast Database Recovery (FDBR) region.
- The **IMPORT** command is valid only in a DRD environment.

The runtime resource and descriptor definitions that are created by the **IMPORT** command exist in the online system until IMS terminates, unless they are deleted using a **DELETE** command. They are recoverable across an IMS warm start or emergency restart.

To preserve the resource and descriptor definitions across a cold start, export the definitions to an RDDS or the repository before IMS terminates. Then import the stored definitions from the RDDS or the repository back into IMS either during cold start processing by using the automatic import function or, when IMS is up and running, by using the **IMPORT** command.

2.2 Dynamic resource definition

DRD is an IMS function that you can use to manage IMS resources dynamically, without using the traditional IMS system generation. IMS 12 brings several enhanced DRD ISPF panels to support the use of the IMS repository function.

The aggregated benefits of providing support to the IMS Resource Definition Repository (Repository) are to simplify the management of IMS system and to provide a single safe source for information about IMS resources.

2.2.1 Background information about DRD

A request to add or modify the current set of programs, transactions, and database does not necessarily force a complete system definition and an IMS recycle. You can evaluate the request and if does not involve changes to the IMS network or the use of static terminals as defined in the current IMS system definition, then you can use the online change process to make the changes while the IMS system is up and running.

IMS systems uses the online change process to add, change, and delete resource definitions while the system is running. The online change process requires that you perform the following tasks:

- Generate the resource into IMS and store them in an MODBLKS staging library data set.
- Run the Online Change Copy utility (DFSUOCU0) to copy into an inactive library.
- Run a series of online change commands to cause the change to take effect.
Figure 2-6 shows the online change process.

DRD is an IMS function that enables users to create, update, query, and delete IMS resources and their descriptors dynamically, without using the batch system definition or online change processes. Using the DRD, you can handle the following IMS resources:

- Transactions
- Application programs
- Databases
- Fast Path routing codes

With DRD enabled, the **APPLCTN**, **DATABASE**, **RTCODE**, and **TRANSACT** macros are optional in the IMS system generation input deck. You can either use type-2 commands to define resources to IMS dynamically or import resource definitions into IMS from a RDDS.
Figure 2-7 shows the DRD flow.

![DRD Flow Diagram](image)

**Figure 2-7  DRD flow**

### 2.2.2 Description of DRD

With DRD enabled, you can use the CREATE, IMPORT, UPDATE, and DELETE type-2 commands to dynamically create, update, and delete application program, database, routing code, and transaction resource and descriptor definitions.

You can issue these type-2 commands either from a TSO SPOC or by using the IMS Manage Resources application that is available from the IMS Application Menu at option 2.

Figure 2-8 shows the first DRD panel.

![IMS Managed Resources Panel](image)

**Figure 2-8  IMS Managed Resources panel for DRD**
With DRD enabled, you can perform several tasks to manage the resource and descriptor definitions for your IMS system:

- Create resources or descriptors.
- Update resources or descriptors.
- Delete resources or descriptors.
- Query definitional attributes of resources or descriptors.
- Export resource and descriptor definitions.
- Import resource and descriptor definitions.

Resource descriptors are templates that can be used to define new resources and descriptors. IMS supplies four resource descriptors, one for each resource type. The descriptors contain IMS-system default values for each resource attribute. These IMS-supplied descriptors cannot be deleted or modified. The following resource descriptors are IMS-supplied:

- **DFSDSDB1** (database descriptor)
- **DFSDSPG1** (application program descriptor)
- **DBFDSRT1** (Fast Path routing code descriptor)
- **DFSDSTR1** (transaction descriptor)

When you create a resource without specifying a model (that is, you do not specify the LIKE keyword), any attribute values that are not specified on the `CREATE` command are inherited from the default descriptor. When you create a resource that is modeled from a descriptor (using the LIKE keyword), any attribute values that are not specified on the `CREATE` command are inherited from the descriptor.

Similarly, when you create a resource using an existing resource as a model (using the LIKE keyword), any attribute values that are not specified on the `CREATE` command are inherited from the existing resource.

These resource definitions can then be imported from the RDDS during cold start processing. To ensure that your changes are recovered across a cold start, set up your system to automatically export your resource and descriptor definitions at checkpoint time and to automatically import the definitions from the RDDS or the repository during cold start processing.

Specify parameters related to DRD in the DFSDFxxx member of the IMS.PROCLIB data set. The DFSDFxxx member of the IMS.PROCLIB data set contains parameters for the IMS CSL, shared queues, databases, restart exit routines, DRD, the IMSRSC repository, dynamic database buffer pools, the Fast Path 64-bit buffer manager, and the IMS abend search and notification procedure.

The MODBLKS= keyword enables either DRD (MODBLKS=DYN) or the online change process (MODBLKS=OLC). The MODBLKS= keyword can only be changed as part of a cold start. MODBLKS=OLC and MODBLKS=DYN are mutually exclusive. You can specify the MODBLKS= keyword in either the DFSCGxxx member, or in the CSL section of the DFSDFxxx member. If you specify a value for the MODBLKS= keyword in the DFSCGxxx member, that value overrides the value specified for the MODBLKS= keyword in the DFSDFxxx member.

If the online change process is disabled (DRD is enabled), the IMS, DBC, and DCC procedures no longer require the DD statements for the IMS.MODBLKS data sets, IMS.MODBLKSA, and IMS.MODBLKSB.
2.2.3 DRD enhancement with IMS 12

DRD is enhanced for IMS Version 12 in the following areas: the Manage Resources ISPF panels are enhanced, the use of the IMPORT command with a new UPDATE option, and DRD can use the IMS resource definition (IMSRSC) repository.

The following Manage Resources ISPF panels are enhanced in IMS 12:

- DELETE DEFN
- EXPORT DEFN
- IMPORT DEFN
- QUERY DB
- QUERY DBDESC
- QUERY PGM
- QUERY PGMDESC
- QUERY RTC
- QUERY RTCDESC
- QUERY TRAN
- QUERY TRANDESC

For information about the IMPORT command with the new UPDATE option, see 2.1, “TSO SPOC” on page 12. For information about the IMS resource definition (IMSRSC) repository, see Chapter 7, “IMS repository” on page 199.

If you use the DRD function for the first time when running on IMS Version 12, you can use the IMS repository function without ever using a RDDS. Although support for RDDSs continues in IMS Version 12, the IMSRSC repository is a strategic alternative to the RDDS.

2.2.4 Using DRD

With DRD enabled, you can perform several tasks to manage the resource and descriptor definitions for your IMS system:

- Update IMS resources or resource descriptors by using the IMS UPDATE command.
- Delete Language Environment® runtime options, resources, and resource descriptors by using the IMS DELETE command.
- Display information about IMS resources by using the IMS QUERY command.
- Export runtime resource and descriptor definitions from the online IMS system to a RDDS or the IMSRSC repository as stored resource definitions by using the EXPORT command.
- Create resource and descriptor definitions or replace existing resource and descriptor definitions in an online IMS system by using the IMPORT command and by using the definitions in a RDDS or the IMSRSC repository.

Following are examples of creating and updating resources in DRD enabled IMS environment.
Creating resources or descriptors

Use the IMS CREATE commands to create resources and resource descriptors. Figure 2-9 shows the panel to introduce the definition of the new transaction resource, called DLRINQ, for the I12A IMS of the IMSplex.

**Figure 2-9 Creating a new transaction called DLRINQ (part 1 of 2)**

Figure 2-10 shows the panel to define the properties of the DLRINQ transaction.

**Figure 2-10 Creating a new transaction called DLRINQ (part 2 of 2)**
Importing resources

Import resource and descriptor definitions from an RDDS or a repository into IMS. Figure 2-11 shows the panel to identify the type of resource to be imported as a transaction, DLRINQ, in this case from I12A IMS system to I12B IMS system.

Figure 2-11  Import DLRINQ Transaction to I12C IMS system (part 1 of 2)

Figure 2-12 shows the panel to create the imported copy.

Figure 2-12  Import DLRINQ transaction to I12C IMS system (part 2 of 2)
2.2.5 Considerations for using DRD

Unless you have a simple system and the online change process fully meets your requirements, use DRD with repositories or DRD with RDDSs rather than the IMS.MODBLKS data set.

**Attention:** DRD changes are not necessarily made across all IMS systems in an IMSplex. Changes might be successful on some IMS systems but fail on others. You must verify that changes have been made across all systems.

If you plan to use DRD in your IMS system, be aware that several restrictions apply to your use of DRD, as explained here:

- When DRD is enabled, you cannot perform online change for the resources that are typically defined in the IMS.MODBLKS data set. If you have a simple system such as a single IMS and online change meets your requirements, consider continuing to use online change and not enabling DRD.

- In an IMSplex with only one IMS and DRD enabled, the `/MODIFY PREPARE MODBLKS` and `INITIATE OLC PHASE(PREPARE) TYPE(MODBLKS)` commands are rejected. The `/MODIFY PREPARE ALL` and `INITIATE OLC PHASE(PREPARE) ALL` commands do not apply to the IMS.MODBLKS data set.

- When DRD is disabled, `CREATE`, `DELETE`, `IMPORT`, and most `UPDATE` commands that change the definitional attributes of a resource are rejected. The following parameters for the `UPDATE TRAN` command are permitted regardless of whether DRD is enabled:
  - `CLASS(class)`
  - `CPRI(value)`
  - `LCT(value)`
  - `LPRI(value)`
  - `MAXRGN(number)`
  - `MSNAME(name)`
  - `NPRI(value)`
  - `PARLIM(value)`
  - `PLCT(value)`
  - `SEGNO(number)`
  - `SEGSZ(size)`
  - `TRANSTAT(Y | N)`

- You cannot delete an IMS-supplied descriptor. The only attribute you can update on an IMS-supplied descriptor is the DEFAULT attribute.

- Because of the default OM routing in a sysplex, DRD commands are routed to all the IMS systems, unless you specify otherwise. However, the commands are not coordinated across the sysplex, so a command can succeed on some IMS systems and fail on others.

- The Multiple Systems Verification utility (`DFSUMSV0`) can verify resources defined only by the batch system definition process; it cannot verify resources that were created using DRD. Use the `/MSVERIFY` command to verify resources that are created dynamically.

- You can update a resource or a descriptor definition, but the changes to that resource or descriptor definition are not propagated to the resource or descriptor definitions that were built from the updated resource or descriptor definition.

- You can import a resource or descriptor and it affects any resource or descriptor explicitly specified in the `IMPORT DEFN` command or RDDS, but it does not affect any resources or descriptors built from the specified resource or descriptor.
2.3 ACB online change

The global online change function modifies resources online for all IMS systems in an IMSplex. With IMS, you can perform an online change only for the DBDs and PSBs that are specified in the appropriated command.

The benefit is that in an IMSplex environment, you can use the application control block (ACB) member online change (OLC) function with `OPTION(NAMEONLY)` specified on the appropriated command to process only the DBDs and PSBs that are specified.

2.3.1 Background information about online change

The global online change function changes resources online for all IMS systems in an IMSplex, bringing the following benefits:

- Simplified process for changing resources online for all IMS systems in the IMSplex
- Continuous processing because each IMS in the IMSplex is not manually coordinated
- Avoidance situations where online change is committed just for some IMS systems

With global online change, the master IMS control region coordinates online changes of the following resources:

- Databases (data management block (DMB) in the ACB library (ACBLIB))
- Database directories (DL/I database directory (DDIR) in MODBLKS)
- MFS formats (FMTLIB)
- Programs (PSB in ACBLIB)
- Program directories (PSB directory (PDIR) in MODBLKS)
- Transactions (scheduler message blocks (SMBs) in MODBLKS)
- Routing codes (Fast Path routing codes (RCTEs) in MODBLKS)

Use the `INITIATE OLC` command to initiate the global online change process.

The `INITIATE OLC` command master usually performs the online change phase locally. If it fails locally, the command master usually skips sending the online change phase to the other IMS systems, sets a completion code for each other IMS indicating that the online change phase was not attempted, and terminates command processing. However, if the `INITIATE OLC PHASE(COMMIT)` command fails on the local IMS because of work in progress for resources that are directly affected by the online change, the command master still sends the commit phase 1 to the other IMS systems. The purpose is to report work in progress for all the IMS systems in the IMSplex, to facilitate completion of the work in progress.

Before you can use online change, you must create three copies of each of the following libraries:

- **IMS.MODBLKS**: This library contains the control blocks to support online change of databases, programs, transactions, and MFS formats.
- **IMS.ACBLIB**: This library contains database and program descriptors.
- **IMS.FORMAT**: This library contains your MFS maps produced by the MFS Language and Service utilities.

These libraries are for the exclusive use of IMS offline functions and are called the staging libraries. Two copies are made of each library, producing data sets with a data set name suffixed with A and B, for example, IMS.MODBLKSA and IMS.MODBLKSB. These two copies of each library are used by the IMS online system.

Figure 2-6 on page 18 shows the online change process.
2.3.2 Description of online change

The global online change process operates through a master IMS control region, which uses the Resource Manager (RM) to coordinate the phases of online change with the other IMS systems in the IMSplex. If you are running an IMSplex that does not use RM, you can specify the CSL global parameter RMENV=N in the DFSCGxxx IMS.PROCLIB member. If this parameter is specified, IMS does not require RM and will use type-2 command support.

The following components are required for a global online change in the IMSplex:

- An IMSplex defined with CSL and at least one OM
- Common Queue Server (CQS), if there is a resource structure in the IMSplex
- OLCSTAT data set, which must be initialized by the Global Online Change utility (DFSUOLC0)
- OLC=GLOBAL and OLCSTAT= parameters in the DFSCGxxx PROCLIB member data set
- CSLG= parameter in the IMS EXEC statement

If you exclude RM from the IMSplex by specifying RMENV=N, each IMS system must have its own OLCSTAT data set and that OLCSTAT data set must contain the IMSID of the IMS system that owns it and no other IMSIDs.

At least one RM and a resource structure are recommended for global online change, but they are not required. Take into consideration that, if you do not use an RM in your IMSplex, the OLCSTAT data set can contain only the IMSID of the IMS system that owns that OLCSTAT data set. Any attempt to restart an IMS system that contains an OLCSTAT data set with a different or multiple IMSIDs results in an abend. IMS rejects INITIATE OLC and TERMINATE OLC commands that are issued by an IMS system other than the IMS system that owns the OLCSTAT data set.

To enable an IMSplex for global online change, you must perform the following tasks:

- Run the Global Online Change utility (DFSUOLC0) to initialize the OLCSTAT data set.
- For each IMS in the IMSplex, perform these steps:
  - Remove the MODSTAT DD and MODSTAT2 DD statements from the IMS control region JCL.
  - Define DFSCGxxx IMS.PROCLIB member parameters related to global online change, or in the CG section of DFSDFxxx member.

The INITIATE OLC (initiate online change) command is provided to support global online change, where online change is coordinated across IMS systems in the IMSplex. The INITIATE OLC command is similar to the /MODIFY PREPARE and /MODIFY COMMIT commands, except that it applies to an IMSplex-wide global online change.

If the INITIATE OLC PHASE(COMMIT) command fails for any IMS before the OLCSTAT data set is updated, either correct the errors and try the commit again or terminate the online change with a TERMINATE OLC command. If the INITIATE OLC PHASE(COMMIT) command fails for any IMS after the OLCSTAT data set has been updated, correct the errors and try the commit again. The online change cannot be terminated.

If you want to fall back to the previous version of the changed resource, perform a full online change process with a full library switch.
2.3.3 ACB online change IMS 12 enhancements

IMS 10 added, in an IMSplex environment, the possibility to use the ACB member OLC function to add or change individual members of the ACB library, or the entire ACB library, and bring these new or changed members online without quiescing the IMSplex or refreshing the active ACB library. Members that are not affected by the member OLC are not quiesced.

With IMS 12, the ACB member online change function is enhanced to process only the DBDs and PSBs that are specified in NAME() keyword of the INIT OLC TYPE(ACBMBR) command.

Figure 2-13 shows the syntax of the INITIATE command.

![Figure 2-13 INITIATE command syntax](image)

The NAME(acbmbr) parameter
The NAME(acbmbr) parameter specifies that one or more ACB library members are to be copied from the staging ACB library to the active ACB library. The acbmbr parameter can be a DBD or PSB that is being changed or added. Wildcard characters are not supported in the member names.

If a DBD is being changed and OPTION(NAMEONLY) is not specified, you do not have to specify the associated PSBs on the command because all of the PSBs that are associated with the changed DBD are copied automatically from the staging ACB library to the active ACB library.

If a DBD that is being changed or added has external references and OPTION(NAMEONLY) is not specified, the secondary index DBD does not have to be specified on the acbmbr parameter. The INIT OLC TYPE(ACBMBR) command processing copies all externally referenced members of the DBD from the staging ACB library to the active ACB library.
The **OPTION NAMEONLY** parameter

The **OPTION NAMEONLY** parameter specifies that only the DBDs and PSBs that are specified in the **NAME** keyword are to be processed.

If a PSB is specified in the **NAME** keyword, the following rules apply:

- Regardless of whether a PSB is new to the active ACB library or not, if the DBDs in the intent list have not been modified, this PSB is copied to the active ACB library.
- If a PSB is not new to the active ACB library and the DBDs in the intent list have been modified, this PSB is not copied to the active ACB library and a completion code 12B is returned.

If a DBD is specified in the **NAME** keyword, the following rules apply:

- If a DBD is new to the active ACB library and the DBDs in the reference list have not been modified, this DBD is copied to the active ACB library. After an **INITIATE OLC PHASE(COMMIT)** command, when the **TYPE(ACBMBR)** online change completes successfully, the active and inactive ACB library suffixes and the MODID are not updated in the OLCSTAT data set or in the IMS control blocks because ACB library member online change processing changes or adds ACB members directly into the active ACB library.
- If a DBD is new to the active ACB library and the DBDs in the reference list have been modified, this DBD is not copied to the active ACB library and a completion code 12C is returned.
- If a DBD is not new to the active ACB library and has been modified, this DBD is not copied to the active ACB library and a completion code 12A is returned.

### 2.3.4 Using ACB online change

The ACB member online change uses IMS type-2 commands only. The following commands are involved in performing an ACB library member online change:

- **INITIATE OLC PHASE(PREPARE)**
- **INITIATE OLC PHASE(COMMIT)**
- **TERMINATE OLC**
- **QUERY MEMBER TYPE(IMS)**
- **QUERY OLC SHOW(RSCLIST)**

The **INIT.OLC PHASE(PREPARE) TYPE(ACBMBR)** command

The **INIT.OLC PHASE(PREPARE) TYPE(ACBMBR)** command specifies that an ACB library member online change process is to be performed for the specified ACB library member name or names. The command response lists all the members that are to be copied from the staging ACB library to the active ACB library during the **COMMIT** command processing.
Figure 2-14 shows the **INITIATE OLC** command using a TSO SPOC in an IMSplex.

**The INIT.OLC PHASE(COMMIT) command**

The **INIT.OLC PHASE(COMMIT)** command performs the online change commit phase on each IMS listed in the OLCSTAT data set, which commits the online changes by bringing all the newly defined resources online, updating the changed resources, and removing the deleted resources.

The commit phase consists of commit phase 1, the OLCSTAT data set update, commit phase 2, and commit phase 3. The OLCSTAT data set is updated with the new current online change libraries and the list of IMS systems that are current with the current online change libraries. The commit phase 2 switches the online environment from the active ACBLIB, FORMAT, or MODBLKS libraries to the inactive libraries containing the new or changed resource descriptions.

Figure 2-15 shows the sequence to perform a online change in an IMSplex.
2.3.5 Considerations for the ACB online change

Use the Global Online Change utility with caution, so that you do not inadvertently destroy valid OLCSTAT data set contents. If you do destroy valid OLCSTAT data set contents, global online change and initialization of additional IMS systems fail until the OLCSTAT data set is re-initialized.

Establish an OLCSTAT data set recovery procedure to deal with the loss of the OLCSTAT data set. After every successful global online change, record the modify ID, the active online change library suffixes, and the list of IMS systems that are current with the online change libraries. If the OLCSTAT data set is destroyed, run the initialize function of the Global Online Change utility with the saved data to re-initialize the OLCSTAT data set.

Consider the following points:

- The INITIATE OLC command is not supported if local online change is enabled.
- This command is invalid on XRF alternate, RSR tracker, and FDBR systems.

2.4 The /DIAGNOSE command

The /DIAGNOSE command retrieves diagnostic information for system resources such as IMS control blocks and user-defined resources at any time without creating a disruptive situation.

IMS 12 improves the reliability of diagnostic information by supporting additional control blocks for /DIAG SNAP command. The overall benefit of these enhancements is an improved reliability of diagnostic information and, ultimately, to streamline the problem determination process.

2.4.1 Background on the /DIAGNOSE command

Users should have the ability to gather diagnostic information without affecting the normal business operations. In most cases, gathering a console dump in a productive environment is disruptive and can negatively impact the service level agreement (SLA). Therefore, users are obligated to produce these dumps off the business prime hours.

The /DIAGNOSE command alleviates this situation by allowing you to take a snapshot of IMS system resources at any time without affecting system availability. The /DIAGNOSE command enables users to retrieve diagnostic information for system resources such as IMS control blocks, user-defined nodes, or user-defined transactions at any time without taking a console dump.

The /DIAGNOSE command SNAP function takes a current snapshot of system resources and displays the response into the issuing LTERM. Optionally, the resource information can be sent to either an online log data set (OLDS) or trace data sets as type X'6701' log records.

The /DIAGNOSE command SNAP function captures information for the following resources:

- A specific IMS control block.
- A user-defined resource.
- Primary control blocks for a dependent region.
- Any area of storage within the region address space.
- Prolog information for an IMS load module.
- A user-defined shared queues structure.
2.4.2 Description of the /DIAGNOSE command

The /DIAGNOSE command SNAP function provides a non-intrusive way to capture IMS resources and control blocks. Using this command can decrease the time needed to generate problem determination data.

The /DIAGNOSE command SNAP function takes a current snapshot of system resources at any time without negatively impacting the IMS system. The SNAP function of the /DIAGNOSE command captures storage information and shows information about the issuing LTERM. Optionally, the resource information can be sent to either an OLDS or trace data sets as type X'6701' log records.

The /DIAGNOSE command is a standard type-1 command. It can be issued from an IMS terminal, a console WTOR, APPC and OTMA clients, an AOI program, MCS/EMCS consoles, and any OM command clients including SPOC.

The /DIAGNOSE command SNAP function captures information for the following areas:

- A specific IMS control block. The /DIAGNOSE SNAP BLOCK(CSCD) command captures storage information for the APPC/OTMA SMQ SCD Extension control block.
- A user-defined database, communication line, logical link, node, program, transaction, logical terminal (LTERM), or USER.
- Primary control blocks for a dependent region.
- Any area of storage within the control region address space (by specifying the address of that storage area).
- Prolog information for an IMS load module. The /DIAGNOSE SNAP MODULE(modname) command identifies the entry point address and captures prolog information for the specified IMS module. The prolog information contains the current maintenance level for a module on your system, which can help you to determine if any maintenance is missing.
- A user-defined shared queues structure. The /DIAGNOSE SNAP STRUCTURE(structurename) command captures storage information for the DFSSQS control block storage for the specified shared queues structure.

The SNAP function of the command captures storage, both addresses and raw data, for the requested IMS control blocks and resources. The information in the blocks is copied to a copy storage area to avoid holding enqueues, locks, latches, and others. The environment is further protected by a separate ESTAE routine that protects the copy process and also prevents an IMS failure.

You can also use the /DIAGNOSE command SNAP function to perform these tasks:

- Show filtered resource information captured by the SNAP function.
- Specify a limit for the number of lines to display.
- Specify the control blocks to be captured by the SNAP function.

Table 2-1 shows the valid environments for commands and keywords.

<table>
<thead>
<tr>
<th>Commands or keywords</th>
<th>DB/DC</th>
<th>DBCTL</th>
<th>DCCTL</th>
</tr>
</thead>
<tbody>
<tr>
<td>/DIAGNOSE</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>ADDRESS</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>AOSLOG</td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>
2.4.3 /DIAGNOSE IMS 12 enhancements

The /DIAGNOSE SNAP command is enhanced in IMS 12 to improve the reliability of diagnostic information and to streamline the problem determination process.

The /DIAGNOSE command supports the following additional control blocks:

<table>
<thead>
<tr>
<th>Commands or keywords</th>
<th>DB/DC</th>
<th>DBCTL</th>
<th>DCCTL</th>
</tr>
</thead>
<tbody>
<tr>
<td>AREA(areaname)</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BLOCK</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>DB</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JOBNAME</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>LINE(linenumber)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LINK(linknumber)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LTERM</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>MODULE</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>NODE</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPTION</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>PGM(pgmname)</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>REGION(regionnumber)</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>SET</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>SHOW</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>SNAP</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>STRUCTURE</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TRAN</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>USER</td>
<td>X</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The /DIAGNOSE command SNAP function has the following new options:
- A DISPLAY option to route output back to the issuing LTERM
- A LIMIT option to restrict the number of lines of output going to LTERM
- A SHOW parameter to control the type and amount of output produced
2.4.4 Using the /DIAGNOSE command

Use the /DIAGNOSE command to retrieve diagnostic information about system resources at any time, and to enable and disable diagnostic features such as the facility to capture the events related to APPC and OTMA synchronous transactions in a shared-queues environment.

The following new keywords are introduced by IMS 12:

The /DIAGNOSE SNAP AREA command
The /DIAGNOSE SNAP AREA command captures control block information for the Fast Path area specified in the areaname parameter. The areaname parameter specified must be alphanumeric, no longer than 8 characters, and identify a currently defined Fast Path area. Multiple areaname parameters can be specified with each parameter separated by a comma or a blank.

The /DIAGNOSE SNAP AREA() command is available in a DB/DC or DBCTL environment where Fast Path is defined. The DEDB extended area control block (EMAC) is available only in an RSR tracker environment.

Table 2-2 shows the /DIAGNOSE SNAP AREA() control blocks.

Table 2-2   The /DIAGNOSE SNAP AREA() control blocks

<table>
<thead>
<tr>
<th>Name</th>
<th>Block description</th>
<th>Macro</th>
<th>Primary</th>
<th>Optional</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADSC</td>
<td>DEDB area data set control block</td>
<td>DBFADSC</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>ALDS</td>
<td>DEDB area name list entry</td>
<td>DBFAREA</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>DDIR</td>
<td>Database directory block</td>
<td>DFSDDIR</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>DMAC</td>
<td>DEDB area control block</td>
<td>DBFDMAC</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>DMAX</td>
<td>DMAC ERE extension block</td>
<td>DBFDMHV</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>DMCB</td>
<td>DEDB master control block</td>
<td>DBFDMCB</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>DMHR</td>
<td>DEDB buffer header (SDEP)</td>
<td>DBFDMHR</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>DMSL</td>
<td>Data space map list</td>
<td>DBFDMSL</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>DSME</td>
<td>Data space mapping entry</td>
<td>DBFDSME</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>EMAC</td>
<td>DEDB extended area control block</td>
<td>DBFEMAC</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>MRMB</td>
<td>DEDB randomizing module block</td>
<td>DBFDMRMB</td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

The /DIAGNOSE SNAP DB command
The /DIAGNOSE SNAP DB command captures control block information for the database specified in the dbname parameter. The dbname parameter specified must be alphanumeric, no longer than 8 characters, and identify a currently defined database. Multiple dbname parameters can be specified with each parameter separated by a comma or a blank.
The /DIAGNOSE SNAP DB() command is available in a DB/DC or DBCTL environment. If the /DIAGNOSE SNAP DB() command is issued in a DCCTL environment, a DFS110I error message is issued in response.

Table 2-3 shows /DIAGNOSE SNAP DB() control blocks.

### Table 2-3 The /DIAGNOSE SNAP DB() control blocks

<table>
<thead>
<tr>
<th>Name</th>
<th>Block description</th>
<th>Macro</th>
<th>Primary</th>
<th>Optional</th>
</tr>
</thead>
<tbody>
<tr>
<td>DDIR</td>
<td>Database directory block</td>
<td>DFSDDIR</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>DMB</td>
<td>Data management block</td>
<td>DFSDMB</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>DMCB</td>
<td>DEDB master control block</td>
<td>DBFDMCB</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>BHDR</td>
<td>MSDB header</td>
<td>DBFDMDB</td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

The /DIAGNOSE SNAP LINE command

The /DIAGNOSE SNAP LINE command captures control block information for the communication line specified in the line# parameter. The line# parameter specified must be numeric, in the range 1–1000, and identify a currently defined communication line. Multiple line# parameters can be specified with each parameter separated by a comma or a blank.

The /DIAGNOSE SNAP LINE() command is available in a DB/DC or DCCTL environment. If the /DIAGNOSE SNAP LINE() command is issued in a DBCTL environment, a DFS110I error message is issued in response. The extended communication name table (ECNT) is available only in an IMS system where Fast Path is defined.

Table 2-4 shows /DIAGNOSE SNAP LINE() control blocks.

### Table 2-4 The /DIAGNOSE SNAP LINE() control blocks

<table>
<thead>
<tr>
<th>Name</th>
<th>Block description</th>
<th>Macro</th>
<th>Primary</th>
<th>Optional</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECB</td>
<td>Event control block</td>
<td>IEPF</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>CLB</td>
<td>Communication line block</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>CTB</td>
<td>Communication terminal block</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>CTT</td>
<td>Communication translate table</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>CRB</td>
<td>Communications restart block</td>
<td>ICLI</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>SPQB</td>
<td>Subpool queue block</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>SPQBEXT</td>
<td>Subpool queue extension block</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>EMHB</td>
<td>Expedited message handler block</td>
<td>DBFEMHB</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>CNT</td>
<td>Communication name table</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>ECNT</td>
<td>Extended communication name table</td>
<td>DBFECNT</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>CCB</td>
<td>Conversational control block</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>CIB</td>
<td>Communication interface block</td>
<td>ICLI(CIB)</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>INBUF</td>
<td>Input line buffer</td>
<td>ICLI</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>OUTBUF</td>
<td>Output line buffer</td>
<td>ICLI</td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>
The /DIAGNOSE SNAP LINK command

The /DIAGNOSE SNAP LINK() command captures control block information for the logical link specified in the link# parameter. The link# parameter specified must be numeric, in the range 1–675, and identify a currently defined logical link. Multiple link# parameters can be specified with each parameter separated by a comma or a blank.

The /DIAGNOSE SNAP LINK() command is available in a DB/DC or DCCTL environment. If the /DIAGNOSE SNAP LINK() command is issued in a DBCTL environment, a DFS110I error message is issued in response.

Table 2-5 shows the /DIAGNOSE SNAP LINK() control blocks.

Table 2-5  The /DIAGNOSE SNAP LINK() control blocks

<table>
<thead>
<tr>
<th>Name</th>
<th>Block description</th>
<th>Macro</th>
<th>Primary</th>
<th>Optional</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECB</td>
<td>Event control block</td>
<td>IEPF</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>LLB</td>
<td>Link link block</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>LTB</td>
<td>Link terminal block</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>CTT</td>
<td>Communication translate table</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>CRB</td>
<td>Communications restart block</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>SPQB</td>
<td>Subpool queue block</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>SPQBEXT</td>
<td>Subpool queue extension block</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>EMHB</td>
<td>Expedited message handler block</td>
<td>DBFEMHB</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>LNB</td>
<td>Link name table</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>ECNT</td>
<td>Extended communication name table</td>
<td>DBFECNT</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>CCB</td>
<td>Conversational control block</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>CIB</td>
<td>Communication interface block</td>
<td>ICLI(CIB)</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>INBUF</td>
<td>Input line buffer</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>OUTBUF</td>
<td>Output line buffer</td>
<td>ICLI</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>LCB</td>
<td>Link control block</td>
<td>LCB</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>LXB</td>
<td>Link extension block</td>
<td>LXB</td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>

The /DIAGNOSE SNAP PGM command

The /DIAGNOSE SNAP PGM command captures control block information for the program specified in the pgmname parameter. The pgmname parameter specified must be alphanumeric, no longer than 8 characters, and identify a currently defined program. Multiple pgmname parameters can be specified with each parameter separated by a comma or a blank.
Table 2-6 shows /DIAGNOSE SNAP PGM() control blocks.

<table>
<thead>
<tr>
<th>Name</th>
<th>Block description</th>
<th>Macro</th>
<th>Primary</th>
<th>Optional</th>
<th>Work</th>
<th>No work</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDIR</td>
<td>Program directory block</td>
<td>DFSPDIR</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RSCX</td>
<td>Resource extension block</td>
<td>DFSRSCX</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTLIST</td>
<td>Intent list</td>
<td>INTLIST</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PSB</td>
<td>Program specification block</td>
<td>DFSPSB</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PST</td>
<td>Partition specification table</td>
<td>IPST</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CNT</td>
<td>Communication name table</td>
<td>ICLI</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMB</td>
<td>Scheduler message block</td>
<td>IAPS</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DMBL</td>
<td>Data management block list</td>
<td>DFSDMBL</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>XPCB</td>
<td>Program communication block index</td>
<td>DFSPCBS</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCB</td>
<td>Program communication block</td>
<td>DFSPCBS</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCBX</td>
<td>Program communication extension block</td>
<td>DFSPCBS</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EPCB</td>
<td>Program communication block extension</td>
<td>DBFEPCBS</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MP62</td>
<td>Message prefix (LU62)</td>
<td>DFS62PRE</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MPOT</td>
<td>Message prefix (OTMA)</td>
<td>DFSYPRE</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PSBPRM</td>
<td>User parameter list block</td>
<td>IDLI</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WK CDS</td>
<td>Data capture segment work area</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WKN DX</td>
<td>Index maintenance work area</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WXXIO</td>
<td>Index I/O work area</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WKS E G</td>
<td>Segment work area</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WKO A</td>
<td>I/O work area</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WKS S A</td>
<td>Segment search argument work area</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W K I F P</td>
<td>Fast Path control block work area</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The /DIAGNOSE SNAP REGION command

The /DIAGNOSE SNAP REGION command captures control block information for the dependent region specified in the region# parameter. The region# parameter specified must be numeric, in the range 1–999, and identify a currently active dependent region. Multiple region# parameters can be specified with each parameter separated by a comma or a blank.

The dependent region might also be identified using the SNAP JOBNAME(jobname) format of the SNAP REGION() resource type. The jobname parameter specified must be alphanumeric, no longer than eight characters, and identify a currently active dependent region. Multiple jobname parameters can be specified with each parameter separated by a comma or a blank.

The REGION(region#) and JOBNAME(jobname) formats can both be specified on the same command.
The SNAP JOBNAME(jobname) format of the SNAP REGION() resource type cannot be used to identify a CCTL thread. All CCTL threads have the same job name and the CICS region, and therefore it is impossible to identify the correct thread by the job name. If multiple regions are started with the same job name, only the first region will be found using the SNAP JOBNAME(jobname) format of the SNAP REGION() resource type.

Figure 2-7 shows the /DIAGNOSE SNAP REGION() control blocks.

Table 2-7  The /DIAGNOSE SNAP REGION() control blocks

<table>
<thead>
<tr>
<th>Name</th>
<th>Block description</th>
<th>Macro</th>
<th>Primary</th>
<th>Optional</th>
<th>System</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>VTD</td>
<td>SVC vector table directory entry</td>
<td>DFSVDIR</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ASCB</td>
<td>MVS™ address space control block</td>
<td>IHAASCB</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ASSB</td>
<td>MVS address space secondary block</td>
<td>IHAASBB</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DPDIR</td>
<td>Dependent region directory block</td>
<td>DFSDPDIR</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IWALE</td>
<td>Internal work area list elements block</td>
<td>DFSIQALE</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LESEP</td>
<td>Local external entry table prefix block</td>
<td>DFSLESEP</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DRAT</td>
<td>DRA thread control block</td>
<td>DFSDRAT</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IDT</td>
<td>Identify table entry</td>
<td>DFSIDT</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCB</td>
<td>MVS task control block (IDT)</td>
<td>IKJTCB</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCB</td>
<td>Task control block (PST)</td>
<td>IKJTCB</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SAP</td>
<td>Save area prefix block</td>
<td>ISAP</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DSPWRK1</td>
<td>Dispatch block: work area part 1 (current)</td>
<td>IDSPWRK</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DSPWRK2</td>
<td>Dispatch block: work area part 2 (current)</td>
<td>IDSPWRK</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCB</td>
<td>MVS task control block (CDSP)</td>
<td>IKJTCB</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RB</td>
<td>MVS associated request block (CDSP)</td>
<td>IHARB</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RBP</td>
<td>MVS associated request block prefix (CDSP)</td>
<td>IHARB</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>XSB</td>
<td>MVS extended status block (CDSP)</td>
<td>IHAXSB</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DSPWRK1</td>
<td>Dispatcher block: work area part 1 (home)</td>
<td>IDSPWRK</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DSPWRK2</td>
<td>Dispatcher block: work area part 2 (home)</td>
<td>IDSPWRK</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DSPPST</td>
<td>PST dispatching control block</td>
<td>IDSPWRK</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>XMCI</td>
<td>Cross-memory control block, ITASK level</td>
<td>DFSXMC</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SSVPL</td>
<td>System service parameter list block</td>
<td>DFSSSVPL</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The `/DIAGNOSE SET AOSLOG` command

The `/DIAGNOSE SET AOSLOG` command updates the attributes of the specified facility. The SET keyword is mutually exclusive with the SNAP keyword.

**The AOSLOG parameter**

The AOSLOG parameter enables or disables logging of events related to APPC and OTMA synchronous transactions in a shared queues environment. When enabled, the events are written to the OLDS as type X'6701' records.

If AOSLOG(ON) is specified in a non-shared-queues environment or when AOS=N is specified in the DFSDCxxx PROCLIB member, the command is rejected with a DFS2859I message.

<table>
<thead>
<tr>
<th>Name</th>
<th>Block description</th>
<th>Macro</th>
<th>Primary</th>
<th>Optional</th>
<th>System</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>DMIB</td>
<td>Directed message manager interface block</td>
<td>DFSDMIB</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CULE</td>
<td>Common use list element block</td>
<td>DFSCULE</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLLE</td>
<td>Common latch list element block</td>
<td>DFSCLLE</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LSMB</td>
<td>Logging secondary master block</td>
<td>DFSLSMB</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SSIDX</td>
<td>Subsystem status index entry</td>
<td>DFSSSIE</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>LCRE</td>
<td>Local current recovery entry</td>
<td>DFSLCRE</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>RRE</td>
<td>Residual recovery element block (LCRE)</td>
<td>DFRSRE</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIB</td>
<td>APPC transaction instance block</td>
<td>DFSTIB</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>YTIB</td>
<td>OTMA transaction instance block</td>
<td>DFSYTIIB</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCENTRY</td>
<td>Protected conversation task table entry</td>
<td>DFSRRSIB</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RRE</td>
<td>Residual recovery element block (PC)</td>
<td>DFRSRE</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PST</td>
<td>Partition specification table</td>
<td>IPST</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>CNT</td>
<td>Communication name table</td>
<td>ICLI</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMB</td>
<td>Scheduler message block</td>
<td>IAPS</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SQPST</td>
<td>Scheduler queue element</td>
<td>ISQPST</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UOW</td>
<td>Unit of work value (QMGR)</td>
<td>DFSUOWE</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UOWE</td>
<td>Unit of work table entry</td>
<td>DFSUOWE</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

If AOSLOG(ON) is specified in a non-shared-queues environment or when AOS=N is specified in the DFSDCxxx PROCLIB member, the command is rejected with a DFS2859I message.

**ON** Enables AOS logging
**OFF** Disables AOS logging
2.4.5 Using the /DIAG SNAP command

Example 2-1 shows the /DIAG SNAP LINE() command being issued and its results.

**Example 2-1 The /DIAG SNAP commands**

```
DIAG SNAP LINE(3) SHOW(ALL) I12A
44I DISPLAY FROM ID=I12A 982
/DIAGNOSE SNAP STORAGE DISPLAY

Resource: LINE(3)
```

```
<table>
<thead>
<tr>
<th>ECB</th>
<th>Event Control Block</th>
<th>Loc: 0004B700</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>809A60BB 04000000D 0004C988 00000000</td>
<td>[......].Ih...]</td>
</tr>
<tr>
<td>0010</td>
<td>0016D698 080000016 00100000 C1E500D3</td>
<td>[.0q........AV.L]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CLB</th>
<th>Communication Line Block</th>
<th>Loc: 0004B720</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>00000000 00000000 0004C988 0304C930</td>
<td>[........].Ih...I.</td>
</tr>
<tr>
<td>0010</td>
<td>0082200 0004CEF0 00000000 063C0023</td>
<td>[.........0......]</td>
</tr>
<tr>
<td>0020</td>
<td>2011221F 19241539 1356016D 03000000</td>
<td>[........].*...0.0...</td>
</tr>
<tr>
<td>0030</td>
<td>010000000 001AF5C 0004CEF0 00000000</td>
<td>[........].0q.......</td>
</tr>
<tr>
<td>0040</td>
<td>00000000 001AF998 00000001 00000001</td>
<td>[.........Ih.........]</td>
</tr>
<tr>
<td>0050</td>
<td>00000000 00000000 00000000 00000000</td>
<td>[.........0q........]</td>
</tr>
<tr>
<td>0060</td>
<td>00000000 00000000 00000000 00000000</td>
<td>[.........0q........]</td>
</tr>
<tr>
<td>0070</td>
<td>00000000 00000000 00000000 00000000</td>
<td>[.........0q........]</td>
</tr>
<tr>
<td>0080</td>
<td>00000000 00000000 00000000 00000000</td>
<td>[.........0q........]</td>
</tr>
<tr>
<td>0090</td>
<td>00000000 00000000 00000000 00000000</td>
<td>[.........0q........]</td>
</tr>
<tr>
<td>00A0</td>
<td>00000000 00000000 00000000 00000000</td>
<td>[.........0q........]</td>
</tr>
<tr>
<td>00B0</td>
<td>00000000 00000000 00000000 00000000</td>
<td>[.........0q........]</td>
</tr>
<tr>
<td>00C0</td>
<td>00000000 00000000 00000000 00000000</td>
<td>[.........0q........]</td>
</tr>
<tr>
<td>00D0</td>
<td>00000000 00000000 00000000 00000000</td>
<td>[.........0q........]</td>
</tr>
<tr>
<td>00E0</td>
<td>00000000 00000000 00000000 00000000</td>
<td>[.........0q........]</td>
</tr>
<tr>
<td>00FF</td>
<td>00000000 00000000 00000000 00000000</td>
<td>[.........0q........]</td>
</tr>
</tbody>
</table>
| 0100    | 00C4E2D7 00000000 00000000 00000000 | \[.DSP...........8...]
| 0110    | 000A0000 20FF4820 8004FC88 00000000 | \[.........8...]

<table>
<thead>
<tr>
<th>CTB</th>
<th>Communication Terminal Block</th>
<th>Loc: 0004CEF0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>0004E888 004B700 F0F0F040 000B0000</td>
<td>[.Yh....000 ...]</td>
</tr>
<tr>
<td>0010</td>
<td>00004000 00000004 01350000 25C5F989</td>
<td>[.......].E.q</td>
</tr>
<tr>
<td>0020</td>
<td>00000000 00000000 00000000 00007FFF</td>
<td>[.].0. T........</td>
</tr>
<tr>
<td>0030</td>
<td>F007FF F0040E3 00000000 00000000</td>
<td>[.0. T........</td>
</tr>
<tr>
<td>0040</td>
<td>00000000 25C5F989 00000000 00000000</td>
<td>[.E.q........</td>
</tr>
<tr>
<td>0050</td>
<td>00000000 00000000 00000000 259D0386</td>
<td>[.E.q........</td>
</tr>
<tr>
<td>0060</td>
<td>00000000 00000000 00000000 00000000</td>
<td>[.E.q........</td>
</tr>
<tr>
<td>0070</td>
<td>00000000 00000000 00000000 00000000</td>
<td>[.E.q........</td>
</tr>
<tr>
<td>0080</td>
<td>00000000 00000000 00000000 00000000</td>
<td>[.E.q........</td>
</tr>
<tr>
<td>0090</td>
<td>00000000 00000000 00000000 00000000</td>
<td>[.E.q........</td>
</tr>
<tr>
<td>00A0</td>
<td>00000000 00000000 00000000 00000000</td>
<td>[.E.q........</td>
</tr>
<tr>
<td>00B0</td>
<td>00000000 00000000 00000000 00000000</td>
<td>[.E.q........</td>
</tr>
<tr>
<td>00C0</td>
<td>00000000 00000000 00000000 00000000</td>
<td>[.E.q........</td>
</tr>
<tr>
<td>00D0</td>
<td>00000000 00000000 00000000 00000000</td>
<td>[.E.q........</td>
</tr>
<tr>
<td>00E0</td>
<td>00000000 00000000 00000000 00000000</td>
<td>[.E.q........</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CTT</th>
<th>Communication Translate Table</th>
<th>Loc: 0004E888</th>
</tr>
</thead>
</table>
```
CRB      Communications Restart Block       Loc:      00000000
   Storage unavailable

SPQB     Subpool Queue Block                Loc:      00000000
   Storage unavailable

SPQBEXT  Subpool Queue Extension Block      Loc:      00000000
   Storage unavailable

EMHB     Expedited Message Handler Block    Loc:      00000000
   Storage unavailable

CNT      Communication Name Table           Loc:      25C5FB98
0000 0004B75C 00000000 00000000 00000000 |...*............|
0010 00000000 00820171 00220022 E2D4C1E2 |.....b......SMAS|
0020 E3C5D940 40000001 0004CEF0 25C5F328 |TER  ......0.E3.|
0030 00000000 00004400 00000000 00000000 |................|
0040 25C5FC10 00000000 0004B7B0 00000000 |.E..............|
0050 00010001 00000000 00000000 00000000 |................|
0060 00000000 00000000 00000000 00000000 |................|

CNT      Communication Name Table           Loc:      25C5F328
0000 00000000 00000000 00000000 00000000 |................|
0010 00000000 00820059 00000000 C9E5D7E2 |.....b......IVPS|
0020 D7D3F140 00000001 0004CEF0 00000000 |PL1 .......0....|
0030 00000000 00004400 00000000 00000000 |................|
0040 25C5F3A0 00000000 0004B7B0 00000000 |.E3.............|
0050 00010001 00000000 00000000 00000000 |................|
0060 00000000 00000000 00000000 00000000 |................|

CCB      Conversational Control Block       Loc:      00000000
   Storage unavailable

CIB      Communication Interface Block       Loc:      00000000
   Storage unavailable

INBUF    Input Line Buffer                   Loc:      00000000
   Storage unavailable
Example 2-2 shows the sample JCL to extract these DIAG records with DFSERA10 and print them with exit DFSERA30.

Example 2-2  JCL for printing DIAG records with exit DFSERA30

```
//P010 EXEC PGM=DFSERA10
//STEPLIB DD DISP=SHR,DSN=IMS12Q.SDFSRESL
//SYSUT1 DD DISP=SHR,DSN=IMS12Q.IMS12A.OLP05
//SYSPRINT DD SYSOUT=*  
//SYSIN DD *
CONTROL CNTL
OPTION PRINT O=5,V=6701,L=2,C=M,E=DFSERA30
OPTION PRINT O=9,V=DIAG,L=4,T=C,C=E,E=DFSERA30
END
/*
```

2.4.6 Considerations for the /DIAGNOSE command

To capture data from a series of /DIAGNOSE commands in a trace data set, issue the commands in the following order:

```
/TRACE SET ON TABLE DIAG OPTION LOG
/DIAGNOSE
/TRACE SET OFF TABLE DIAG
```

2.5 IMS logger

IMS creates logs of activity that are performed during online executions. These log records are initially written to an OLDS, and then copied to the system log data set (SLDS). The log records that reflect completed operations not yet written to an OLDS is recorded on a write-ahead data set (WADS).

IMS 12 improves the functionality of IMS logging by adding Extended Format support and providing the buffer allocation beyond the 31-bit, among other improvements. The benefits of this enhancement is to increase logging rates while releasing storage in the extended common service area (ECSA).

2.5.1 Background on IMS logger

Logs are the list of all activities being performed by the system such as work that has been done and changes that have been made. By knowing the state of the system when things were functioning properly and knowing what the system has done since then, you can recover it in the event of a failure. Logs make recovery and restart possible. As IMS operates, it constantly writes its event information in log records. The log records contain information about:

- When IMS starts and shuts down
- When programs start and terminate
- Changes made to the database
- Transaction requests received and responses sent
- Application program checkpoints
- System checkpoints

Figure 2-16 illustrates the logging process in an IMS online environment.

Extended format data set is a type of SMS-managed physical sequential data set that, externally, has the same characteristics as sequential data sets. However, records are not necessarily stored in the same format or order as they appear. Sequential data striping is implemented through the use of extended format data sets.

Data striping is the technique of segmenting logically sequential data, such as a file, in a way that accesses of sequential segments are made to different physical storage devices.
2.5.2 Description of IMS logger

IMS externalizes log records by writing them to an OLDS.

OLDS
IMS uses a set of OLDSs in cyclical processing, which enables IMS to continue logging after an individual OLDS is filled. Also, if an I/O error occurs while writing to an OLDS, IMS can continue logging by isolating the defective OLDS and switching to another one. After an OLDS is used, it is available for archiving to a SLDS on DASD or tape by the IMS Log Archive utility.

The utility can be executed automatically through an IMS startup parameter (ARC=). When IMS is close to filling the last available OLDS, it warns you so that you can ensure that archiving completes for used OLDSs or add new OLDSs to the system. You can also manually archive the OLDSs to SLDSs using the IMS Log Archive utility. An SLDS can reside on DASD or tape. After an OLDS is archived, it can be reused for new log data. You use SLDSs as input to the database recovery process.

IMS uses the OLDS only in the online environment. The OLDS contains all the log records required for restart, recovery, and both batch and dynamic backout. The OLDS holds the log records until IMS archives them to the SLDS. Define all of the OLDSs in the IMS procedure library (IMS.PROCLIB) using the OLDSDEF statement. The OLDS must be preallocated on a direct-access device. You can also dynamically allocate additional OLDSs while IMS is running by using the /START OLDS command.

IMS uses the Basic Sequential Access Method (BSAM) to write log records to the OLDS, and the Overflow Sequential Access Method (OSAM) to read the OLDS when IMS performs dynamic backout. Although referred to as a data set, the OLDS is actually made up of multiple data sets that wrap around, one to the other. You must allocate at least three, but no more than 100 data sets for the OLDS.

You can specify that the OLDS use dual logging, which is the duplication of information for two logs. When you use dual logging, an I/O error on the primary or secondary data set causes IMS to close the non-error OLDS and mark the error OLDS in the recovery control (RECON) data set as having an I/O error and a close error. IMS then continues logging with the next
available pair. For dual logging, the minimum number of data sets is three pairs, and the maximum number is 100 pairs.

IMS uses as many OLDSs as you allocate. IMS issues a message each time it changes the current OLDS. This message identifies the OLDS being closed and the next OLDS to be used.

Write-ahead data set
To enhance performance and to optimize space on OLDSs, incomplete or partially filled buffers are written to a WADS when necessary for recoverability. The WADSs are high speed DASD data sets with a high write rate. Only complete log buffers are written to OLDSs. When the log data is on an OLDS, the equivalent WADS records are ignored.

IMS continually reuses WADS space after writing the appropriate log data to the OLDS. The log write-ahead function ensures that all log records are on the log before IMS writes changes to a database. IMS updates a database in any of the following situations:
- When IMS needs to reuse the database buffer (if this is before commit)
- During commit
- During VSAM background write

If IMS fails, you use the log data in the WADS to complete the content of the OLDS and then close the OLDS as part of an IMS emergency restart or as an option of the Log Recovery utility. If you close the OLDS during emergency restart, you must include the WADS in use at the time of the failure.

You can change any of the following specifications for the WADS during an IMS restart:
- Number of WADSs
- Sequence of WADSs
- WADS names
- Use of single or dual WADSs

The DFSVSMxx PROCLIB member contains the log data set definition information, and it specifies the allocation of OLDS and WADS and the number of buffers to be used for the OLDS. It also specifies the mode of operation of the OLDS (single or dual).

Attention: To eliminate a potential resource contention, place the WADS on a low-use device that is different from the device you use for the OLDS. If you place the WADS on the same device as one of your OLDSs and use full-track blocking for the OLDS (in which a block is equal to a full track), the device should be able to handle infrequent OLDS seeks. Contention can still occur. If the WADS and OLDS are on the same device, the Log Archive utility (DFSUARC0) or dynamic backout can cause severe contention between an OLDS being archived and an active WADS.

Data striping
Usually, sequential access processing does not allow for any type of parallelism for I/O operations. This means that when an I/O operation is executed for an extent in a volume, no other I/O activity from the same task or same data set is scheduled. In a situation where I/O is the major bottleneck, and there are available resources in the channel subsystem and controllers, it is a waste of these resources. Data striping addresses this sequential access performance problem by adding two modifications to the traditional data organization:
- The records are organized in stripes along the volumes.
- Parallel I/O operations are scheduled to sequential stripes in different volumes.
Sequential data striping can be used for physical sequential data sets that are subject to intensive I/O for critical applications. Sequential data striping uses extended-format sequential data sets that SMS can allocate over multiple volumes, preferably on different channel paths and control units, to improve performance. These data sets must reside on volumes that are attached to IBM 9340 or RAMAC Array Subsystems, to IBM 3990 Storage Subsystems with the extended platform, ESS, or DS8000®.

Sequential data striping can reduce the processing time required for long-running batch jobs that process large, physical sequential data sets. Smaller sequential data sets can also benefit because of the improved buffer management of DFSMS for QSAM and BSAM access methods for striped extended-format sequential data sets. By striping a data set, the access method can spread simultaneous I/Os across multiple devices.

With this format, a single application request for records in multiple tracks and records can be satisfied by concurrent I/O requests to multiple volumes. The result is improved performance by achieving data transfer into the application at a rate greater than any single I/O path.

Data striping distributes data for one data set across multiple volumes.

- A data set can have a maximum of 59 stripes.
- Each stripe must reside on one volume and cannot be extended to another volume.
- I/O can be done in parallel with striped data for better performance.
- To specify extended format, data set type (DSNTYPE) in data class needs to be set to “Extended” and sustained data rate in storage class needs to be set to the number of stripes needed.

### 2.5.3 IMS 12 enhancements for logging

IMS 12 improves the functionality of IMS logging in the following ways:

- It offers Extended Format support for OLDS and SLDS, allowing them to be striped.
- The IMS log buffers can be moved into 64-bit virtual storage to free ECSA storage space for other uses. Log buffers are obtained in 64-bit storage when all the following are true:
  - The new BUFSTOR parameter of the DFSVSMxx PROCLIB member specifies BUFSTOR=64.
  - The OLDS block size is a multiple of 4096.
  - The OLDS are on extended format data sets.
- WADS management is changed to be more efficient.
  - Track groups are no longer used.
  - WADS are written in a simple wrap-around fashion.
- Archive (DFSUARC0) is enhanced to write “undo” records for non-recoverable databases.

### 2.5.4 Using IMS logger

The following sections show examples of how to use the new enhancements.

**Striping**

An OLDS can be defined as a DFSMS extended-format, striped data set. Set the data type of the OLDS data class to EXT to define it as an extended-format data set, and set the storage class SDR to a value that results in multiple stripes. In JCL allocation, the data class is
specified by the DATACLAS parameter and the storage class is specified by the STORCLAS parameter of the DD statement.

Example 2-3 shows the sample JCL to allocate OLDS as a striped data set.

Example 2-3  JCL to allocate OLDS

```
//P010 EXEC PGM=IEFBR14
//OLDS00 DD DSN=IMS12.OLDS00,DISP=(,CATLG,DELETE),
    //      SPACE=(CYL,(100)),
    //      DATACLASS=STRCLAS,STORCLASS=STRSTOR
/*
```

64-bit virtual storage

The IMS log buffers can be moved into 64-bit virtual storage to free ECSA storage space for other uses. Log buffers are obtained in 64-bit storage when all the following are true:

- The new BUFSTOR parameter of the DFSVSMxx PROCLIB member specifies BUFSTOR=64.
- The OLDS block size is a multiple of 4096.
- The OLDS are on extended format data sets.

Example 2-4 shows a sample of the DFSVSMxx using the new BUFSTOR parameter.

Example 2-4  DFSVSMxx using the new BUFSTOR parameter

```
VSRBF=4096,5
VSRBF=2048,5
VSRBF=1024,5
VSRBF=512,5
IOBF=(8192,5,Y,Y)
IOBF=(2048,5,Y,Y)
SBONLINE,MAXSB=10
OPTIONS,BGWRT=YES,INSERT=SKP,DUMP=YES,DUMPIO=YES
OPTIONS,VSAMFIX=(BFR,IOB),VSAMPLS=LOCL
OPTIONS,DL/I=OUT,LOCK=OUT,DISP=OUT,SCHD=OUT,DLOG=OUT,LATC=ON,SUBS=ON
OPTIONS,STRG=ON
OLDSDEF OLDS=(00,01,02,03,99),BUFNO=005,MODE=DUAL,BUFSTOR=64,BLKS=8192
WADSDEF WADS=(0,1,8,9)
ARCHDEF ALL MAXOLDS(1) JOB(JOBJCLA)
```

WADS management

IMS 12 changes the way that WADS writes are done; it no longer uses the concept of track groups. This changes the calculation for the space required for the WADS and also changes the data written by log ahead requests.

In IMS 12, the WADS should be sized to provide enough space for the data in the OLDS buffers that have not yet been written to disk at any, plus one track. In previous versions, the WADS was sized by using the WADS track group concept.

A track group was the OLDS block size/WADS segment size plus 1. A WADS segment size was 2 K for OLDS buffers below the bar in real storage and 4 K for OLDS buffers above the bar in real storage.
The maximum WADS size was \((\text{OLDS block size/WADS segment size}) + 1\) x (number of OLDS buffers) tracks. For example, an installation defined with 200 OLDS buffers of 24 K size above the bar used a WADS with 1400 tracks. With IMS 12, it requires no more than enough tracks to hold \(200 \times 24\) K plus one track, which is approximately 101 tracks.

In previous versions of IMS, the WADS was written in segments from the OLDS buffers. Successive writes were to different tracks. The scheme is much simpler in IMS 12. Each WADS write is to the next block in the data set. The data written to the WADS includes the data that was not previously written up to the last record in the buffer.

### 2.5.5 Considerations for IMS logger

This is a list of possible migration steps to implement striping with OLDS and buffers in 64-bit storage:

1. Define new OLDS data sets with extended formatting:
   - Use a data class in which the data set type is "EXT".
   - Use a storage class with SDR >= 5 MBps.
   - Define MDA members for dynamic allocation.
2. Start a new OLDS data sets with `/START OLDS` commands.
   - Striping will be used for these data sets
   - A mixture of format in OLDS is acceptable when buffers are below the 2 GB bar
3. Stop old OLDS data sets with `/STOP OLDS` commands.
4. Specify `BUFSTOR=64` in the OLDSDEF statement. Also specify `BLKSZ=` in the OLDSDEF statement.
5. Terminate and restart IMS to get buffers above the bar.

Using striped data sets for OLDS and moving log buffers above the 2 GB boundary increases the log rate and frees ECSA storage for other users, respectively. Striping can be beneficial when your IMS system is experiencing a significant number of Wait-for-Writes or Wait-for-Buffers. For Geographically Dispersed Parallel Sysplex (GDPS®) users, keep the number of stripes low.

### 2.6 CQS trace facilities

The CQS is the interface between IMS and the shared queues. IMS can experience problems that must be diagnosed and corrected quickly. Usually, IMS displays symptoms that can help you with your diagnosis, but often you need to gather more information such as the IMS trace.

IMS 12 adds three separate trace tables for shared queues. In addition, you can move Base Primitive Environment (BPE) tracing to an external data set. The benefit of these separate traces is to provide the IMS support team with valuable diagnostic data for client problem resolution, which can help to resolve shared queues-related problems more quickly.
2.6.1 Background on CQS

IMS shared queues were introduced in an earlier version of IMS to provide dynamic workload balancing among multiple IMS subsystems. Using the shared queues, multiple IMS subsystems can share just one set of message queues, which provides the following benefits:

- **Workload balancing**
  Messages in the shared queues can be processed by any IMS subsystem in the sysplex.

- **Increased capacity**
  Any number of IMS subsystems can be used to handle the transaction workload.

- **Improved availability**
  If an IMS subsystem fails, other IMS subsystems can continue to process.

The CQS handles elements on the shared queue structures for its clients. CQS is a general purpose programming facility that provides clients such as IMS with an API for accessing the shared queues.

All IMS subsystems register and communicate with the CQS to receive messages from or send messages to the shared queues. CQS notifies registered clients when there is work for them on the shared queue structures. In an IMS shared-queues environment, the IMS online subsystem identifies itself to the CQS subsystem during IMS initialization or restart.

The BPE configuration parameter member of the IMS PROCLIB data set can be used whenever you are using an IMS address space that uses BPE, such as CQS.

Figure 2-18 shows the shared queues environment.
2.6.2 Description of CQS

To enable shared queues, you must define a CFRM policy that contains structure names and attributes for all message queues. If you use Fast Path and share EMH queues, you must also include structure names and attributes for EMHQ structures. As many as 32 different clients on the same z/OS image can connect to coupling facility structures through a single CQS by using the CQSCONN request. IMS starts the CQS address space if it is not currently active. If CQS is already active, IMS registers with the active CQS address space.

Several different resources can be stored in the resource structure. It is important to create a resource structure of sufficient size to accommodate these resources. An IMS system with a defined resource structure stores transaction names in the resource structure. Additional resources can be stored in the resource structure, depending on the IMS functions that you enable.

Each resource is stored on the resource structure using a 128-byte entry, and either zero, one, or more 512-byte data elements. The 128-byte entry contains 64 bytes for z/OS control information, and 64 bytes for user data in an adjunct area. Both IMS and CQS use a portion of the 512-byte data element as a prefix. The remaining bytes are available for client data.

Use the entry-to-element ratio when allocating the resource structure to reserve portions for entries and data elements. The more accurate the ratio is for actual resources stored on the resource structure, the less storage is wasted. The number of entries is equal to the number of resources. The number of data elements depends on the number of resources for each resource type.

CQS produces SDUMPs for internal errors. The CQS dumps are in the SYS1.DUMP data sets. CQS can also produce LOGREC data set entries for errors.

CQS-related problems
For a CQS environment, related problems might include:

- IMS WAIT problems
- CQS WAIT or HANG problems
- CQS checkpoint problems
- CQS restart problems
- CQS structure rebuild problems

Trace record eye-catchers in a formatted dump provide clues about which functions resulted in errors. You might be able to correct environmental problems immediately. Refer internal IBM problems to IBM with appropriate documentation, such as system console logs and dumps.
CQS trace records are written to one or more of the trace tables shown in Table 2-8.

<table>
<thead>
<tr>
<th>Table name</th>
<th>Number of tables</th>
<th>Table description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERR</td>
<td>1</td>
<td>Errors</td>
</tr>
<tr>
<td>CQS</td>
<td>1</td>
<td>CQS activity, including errors</td>
</tr>
<tr>
<td>INTF</td>
<td>1</td>
<td>CQS interface events</td>
</tr>
<tr>
<td>OFLW</td>
<td>1 per structure</td>
<td>Structure overflow events</td>
</tr>
<tr>
<td>SEVT</td>
<td>1 per structure</td>
<td>Structure event activity</td>
</tr>
<tr>
<td>STR</td>
<td>1 per structure</td>
<td>Client activity for this structure</td>
</tr>
</tbody>
</table>

Each CQS trace record is 32 bytes long, except records in the SEVT and OFLW tables. Those tables use an expanded 64-byte format. In a standard 32-byte trace entry, the first byte is the trace code and the second byte is the trace subcode. The expanded 64-byte format used by trace records in the SEVT and OFLW tables contains 16 words of trace data.

Use the BPE configuration parameter member of the IMS PROCLIB data set to define the BPE execution environment, as explained here:

- Whether trace entries are written to an external data set
- The external data set to which trace entries are written
- The language used for BPE and IMS component messages
- The trace level settings for BPE and IMS component internal trace tables
- The name of a BPE exit list member of the IMS PROCLIB data set where configuration information for IMS component user exit routines is stored
- The time interval between calls to the BPE statistics exit routines

Specify the member name by coding BPECFG=member_name on the EXEC PARM= statement in the address space startup JCL (Example 2-5).

**Example 2-5  CQS JCL sample**

```plaintext
//IEFPROC  EXEC PGM=CQSINIT0,REGION=3000K,
// PARM=('BPECFG=BPECONFG','CQSINIT=12A','SSN=C12A')
//*
//STEPLIB  DD  DSN=IMS12Q.SDFSRESL,DISP=SHR
//PROCLIB  DD  DSN=IMS12Q.PROCLIB,DISP=SHR
//SYSPRINT DD  SYSOUT=*  
//SYSUDUMP DD  SYSOUT=*  
//*
```
2.6.3  CQS enhancements with IMS 12

Before IMS Version 12, all CQS structure events were contained in a single trace table. The following two new trace tables are added:

- The structure event trace table (SEVT) contains all structure event trace entries except entries that are related to overflow events.
- The structure overflow event trace table (OFLW) contains only trace entries that are related to overflow events.

These two new trace event tables are automatically generated by IMS to contain structure event trace entries and structure overflow trace entries. These new tables retain critical trace entries for a longer period of time, which improves CQS serviceability.

All existing trace entries that are not related to client activity are moved to one of the new trace tables and are expanded to take advantage of the larger 64-byte trace entry length. The existing CQS structure trace table (STR) now contains only client activity trace entries.

The following IMS components are updated to support the new trace event table:

- The BPE configuration PROCLIB member
- The CQS BPE EXTERNAL TRACE FORMATTING MENU panel of the IMS Dump Formatter
- The BPE DISPLAY TRACETABLE and UPDATE TRACETABLE commands

The BPE DISPLAY TRACETABLE and UPDATE TRACETABLE commands are enhanced to display information about the trace tables for the Repository Server address space and the new Repository Server DIAG trace table.

TIP: Request response processing for authorized CQS clients in IMS 12 is executed under enclave service request blocks (SRBs). In IMS 12 and subsequent releases, IMS will request z/OS to process such work on an available System z9 Integrated Information Processors (zIIP).

2.6.4  Using CQS

BPE trace records can be written to internal (memory only) trace tables and to external data sets. The default is to write to internal trace tables (EXTERNAL=NO). To write to an external data set, you must set EXTERNAL=YES on the TRCLEV statement and specify the EXTRACE parameter in the BPE configuration parameter member. You must also define a generation data set group (GDG) for BPE to trace into. If you specify EXTERNAL=YES, trace data is written to both an external data set and internal trace tables.

Example 2-6 shows the BPE configuration with new trace entries.

Example 2-6  BPE configuration with new trace entries

```plaintext
# DEFINITIONS FOR BPE SYSTEM TRACES
#
TRCLEV=(AWE,LOW,BPE)  /* AWE SERVER TRACE */
TRCLEV=(CBS,LOW,BPE)  /* CONTROL BLK SRVCS TRACE */
TRCLEV=(DISP,LOW,BPE) /* DISPATCHER TRACE */
TRCLEV=(LATC,LOW,BPE) /* LATCH TRACE */
TRCLEV=(SSRV,LOW,BPE) /* GEN SYS SERVICES TRACE */
TRCLEV=(STG,LOW,BPE) /* STORAGE TRACE */
```
TRCLEV=(USRX,LOW,BPE)    /* USER EXIT TRACE */
#
# DEFINITIONS FOR CQS TRACES
#
TRCLEV=(CQS,LOW,CQS)    /* CQS GENERAL TRACE */
TRCLEV=(ERR,LOW,CQS)    /* CQS ERROR EVENTS */
TRCLEV=(INTF,LOW,CQS)   /* CQS INTERFACE TRACE */
TRCLEV=(OFLW,LOW,CQS)   /* CQS STRUCTURE OVERFLOW TRACE */
TRCLEV=(SEVT,LOW,CQS,EXTERNAL=YES) /* CQS STRUCTURE EVENTS TRACE */
TRCLEV=(STR,LOW,CQS)    /* CQS CLIENT ACTIVITIES TRACE */

You can dynamically change the external trace data set specification in the BPE configuration
PROCLIB member and refresh the member while an address space is running. For example,
if you are running without an external trace data set, you can edit your BPE PROCLIB
member, add an external trace data set specification, and start using external trace without
having to restart the address space.

You can dynamically modify BPE tracing by using the z/OS MODIFY command with the UPDATE
TRACETABLE command as shown in Example 2-7.

Example 2-7   DISPLAY and UPDATE CQS trace command sample

/F IM12ACQS,DISPLAY TRACETABLE NAME(SEVT) OWNER(CQS)
BPE0030I TABLE  OWNER  LEVEL  #PAGES  EXT  #ENTRIES   #CYCLES  C12ACQS
BPE0000I SEVT   CQS    LOW        12  NO          8         0  C12ACQS
BPE0032I DISPLAY TRACETABLE COMMAND COMPLETED C12ACQS

/F IM12ACQS,UPDATE TRACETABLE NAME(OFLW) OWNER(CQS) LEVEL(HIGH)
BPE0032I UPDATE TRACETABLE COMMAND COMPLETED C12ACQS

2.6.5 Considerations for CQS

The overflow (OFLW) event trace table includes activity related to CQS structure overflow
events. CQS defines one OFLW trace table for each structure pair defined to CQS. Trace
entries in this table are 64 bytes long. The default number of pages for this table is 12.

The SEVT includes activity related to CQS structure events. CQS defines one SEVT trace
table for each structure pair defined to CQS. Trace entries in this table are 64 bytes long. The
default number of pages for this table is 12.

The client (STR) event trace table includes CQS client activity events. CQS defines one STR
trace table for each structure pair defined to CQS. The default number of pages for this table
is 8. You cannot set the level for the ERR trace table. BPE forces the level to HIGH to ensure
that error diagnostics are captured.

2.7 Extended address volume

The largest DASD volume was restricted to 65,520 cylinders when EAV was first introduced
to support VSAM data sets. Now, the z/OS operating system supports also non-VSAM data
sets residing in the extended addressing space (EAS).
IMS 12 takes advantage of that enhancement. Now it can address specific non-VSAM data sets residing in the EAS of an EAV. The actual benefits are that users can manage fewer numbers of larger volumes and have less need for multivolume OSAM.

### 2.7.1 Background on EAV

Before EAV was introduced, the largest DASD volume was restricted to approximately 54 GB (65,520 cylinders), with the track–address restricting DASD volume growth.

Table 2-9 shows the 3390 DASD volume characteristics.

<table>
<thead>
<tr>
<th>Model</th>
<th>Model 1</th>
<th>Model 2</th>
<th>Model 3</th>
<th>Model 9</th>
<th>Model 27</th>
<th>Model 54</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tracks</td>
<td>16,695</td>
<td>33,390</td>
<td>50,085</td>
<td>150,255</td>
<td>491,400</td>
<td>982,800</td>
</tr>
<tr>
<td>Cylinders</td>
<td>1,113</td>
<td>2,226</td>
<td>3,339</td>
<td>10,017</td>
<td>32,760</td>
<td>65,520</td>
</tr>
<tr>
<td>Bytes</td>
<td>946 MB</td>
<td>1.89 GB</td>
<td>2.84 GB</td>
<td>8.51 GB</td>
<td>27.84 GB</td>
<td>55.68 GB</td>
</tr>
</tbody>
</table>

IBM raised the limit on volume sizes introducing a new DASD family with more than 65,520 cylinders in capacity and increasing the amount of addressable DASD storage per volume beyond 65,520 cylinders by changing how tracks on extended count key data (ECKD™) volumes are addressed.

Figure 2-19 shows the EAV characteristics.

![EAV keeps 3390 track format](image)

- **EAV keeps 3390 track format**
  - Track-managed space:
    - Area on EAV within the first 65,520 cyls
    - Space allocated in track or cyl increments
    - Storage for “small” data sets
  - Cylinder-managed space:
    - Area on EAV located above first 65,520 cyls
    - Space is allocated in multicylinder units
    - Storage for “large” data sets
  - New DSCB format types identify EAS data sets
  - New formats (Format 8 and 9) in VTOC

Exploiting the capabilities of a new 3390 device model A on IBM System Storage® DS8000 storage subsystems, EAV is designed to provide a new architectural limit of hundreds of TB per volume, while keeping the fully compatible access to data residing on cylinders below 65,520.

In z/OS V1.10, support is for SMS and non-SMS managed VSAM data sets (entry-sequenced data set (ESDS), key-sequenced data set (KSDS), relative record data set (RRDS), and linear data set (LDS)) at any location on an EAV. In z/OS, Version 1 Release 12 supports is for non-VSAM data sets.
2.7.2 Description of EAV

An EAV is a volume with more than 65,520 cylinders. An EAV increases the amount of addressable DASD storage per volume beyond 65,520 cylinders by changing how tracks on ECKD volumes are addressed.

A track address is a 32-bit number that identifies each track within a volume.

The address is in the format hexadecimal CCCCcccH where:

- **CCCC** The low order 16-bits of the cylinder number.
- **ccc** The high order 12-bits of the cylinder number.
- **H** The four-bit track number.

The combination of the 16-bits and 12-bits for the low order and high order cylinder number represents a 28-bit cylinder number.

For an EAV, the EAS is cylinders whose addresses are equal to or greater than 65,536. The ccc portion is non-zero for the cylinders of EAS. These cylinder addresses are represented by 28-bit cylinder numbers. For compatibility with older programs, the ccc portion is hexadecimal 000 for tracks in cylinders whose addresses are below 65,536. These cylinder addresses are represented by 16-bit cylinder numbers. This is the base addressing space on an EAV.

The cylinder-managed space is space on the volume that is managed only in multicylinder units (a multi-cylinder unit is a fixed unit of disk space that is larger than a cylinder). Cylinder-managed space begins at cylinder address 65,520. Each data set occupies an integral multiple of multicylinder units. Space requests targeted for the cylinder-managed space are rounded up to the next multicylinder unit. The cylinder-managed space only exists on EAVs.

The track-managed space is space on a volume that is managed in tracks and cylinders. Track-managed space ends at cylinder address 65,519. Each data set occupies an integral multiple of tracks. Track-managed space exists on all volumes.

2.7.3 EAV support in IMS 12

IMS 12 provides support for OSAM data sets to be allocated in the EAS of an EAV. The following types of data sets can be allocated:

- **OSAM data sets**
  - OSAM database data sets
  - Restart data set (RDS)
  - Message queue blocks data set
  - Long and short message data set
- **IMS OLDS**
- **IMS log WADS**
- **IMS Spool and Disk Data Sets (that is, **UNITYPE=SPOOL** and **UNITYPE=DISK** on the IMS TERMINAL macro)**
- **BPE external trace data sets**
  - Specified on **EXTTRACE** in BPE configuration
### 2.7.4 Using EAV

For all data set types a new data set attribute, EATTR, has been added so that a user can control whether a data set can have extended attribute DSCBs and can be allocated in EAS.

- **EATTR of NO** indicates that the data set cannot have extended attributes or reside in EAS. This is the default for non-VSAM data sets.
- **EATTR of OPT** indicates that the data set can have extended attributes and can optionally reside in EAS. This is the default for VSAM data sets.

Additionally, the user must specify an EAV on the appropriate volume specification using:

- **AMS DEFINE CLUSTER (VOL (xxxxxx))**
- **ALLOCATE**
- **JCL (VOL=SER=xxxxxx)**
- **Dynamic allocation**
- **Data class**

The following sections describe how some IMS sequential data sets can benefit from EAV support.

#### Online log data sets

IMS creates logs of activity that are based on online executions. Log records are initially written to an OLDS, and then archived into the SLDS. When allocating space for OLDSs, consider how many data sets your system needs, the track size of the storage device, and whether you intend to use dual logging. You can allocate OLDSs as DFSMS extended-format data sets to improve logging performance.

Define the initial set of OLDSs to be acquired by restart initialization in the OLDSDEF control statement in the DFSVSMxx member of IMS.PROCLIB. You can dynamically allocate this set of OLDSs, or specify them through DD statements.

DASD space for each OLDS must be contiguous, and secondary extents are not permitted. Pairs of OLDSs (primary and secondary) must have the same space allocation.

Example 2-8 shows an OLDS allocation using EAV.

```
Example 2-8  Allocate OLDS

//*************
//* ALLOCATE OLDS
//*************
//OLDS    EXEC PGM=IEFBR14
//SYSPRINT DD  SYSOUT=* 
//DFSOLP00 DD  DSN=IMS12Q.IMS12X.OLDSP0,UNIT=SYSDA,VOL=SER=EAS001,
//             DISP=(,CATLG),EATTR=OPT,SPACE=(CYL,(50)),
//             DCB=(RECFM=VB,BLKSIZE=22528,LRECL=22524,BUFNO=15)
/*/```

#### Write-ahead data set

The WADS is a small DASD data set containing a copy of log records reflecting committed operations in the OLDS buffers that have not yet been written to the OLDS. WADS space is continually reused after the records it contains are written to the OLDS. The recommended size for your WADS can vary by storage device.
You can define up to 10 WADSs to use as spares. IMS automatically switches to a spare WADS if the current WADS becomes unusable after an error occurs. If a write error occurs, logging to the WADS continues if at least one WADS is available (for single logging) or two WADSs are available (for dual logging). For additional resiliency, define each WADS on a different hardware device.

You can enable your WADS to use EAVs that are available in z/OS V1.12 or later by specifying an EAV volume on the VOLSER parameter of the DFSWADSnn DD statement when you allocate the data set. In addition, you can specify the attribute EATTR to indicate whether the data set supports extended attributes.

Example 2-9 shows a WADS allocation using EAV.

```
Example 2-9  WADS allocation

//*************
//*   ALLOCATE WADS
//*************
//WADS    EXEC PGM=IEFBR14
//SYSPRINT DD  SYSOUT=*  
//WADS00   DD  DSN=IMS12Q.IMS12X.WADS0,UNIT=SYSDA,VOL=SER=EAS001,
//             DISP=(,CATLG),EATTR=OPT,SPACE=(CYL,(3)),
//             DCB=(RECFM=F,BLKSIZE=4096,LRECL=4096)
/*
```

**Restart data set**

While IMS is running and logging its activities, it takes periodic system checkpoints and writes the checkpoint notification to another data set called the RDS. IMS uses the RDS when it restarts to determine the correct checkpoint from which to restart.

A minimum of five tracks must be allocated to the restart data set because it contains the checkpoint-ID table and other control information.

Example 2-10 shows an RDS allocation using EAV.

```
Example 2-10  RDS allocation

//*************
//*   ALLOCATE RESTART DATA SET
//*************
//RDS     EXEC PGM=IEFBR14
//SYSPRINT DD  SYSOUT=*  
//LGMSG    DD  DSN=IMS12Q.IMS12X.RDS,UNIT=SYSDA,VOL=SER=EAS001,
//             DISP=(,CATLG),EATTR=OPT,SPACE=(CYL,(1)),
//             DCB=(RECFM=FBS,BLKSIZE=4096,LRECL=4096)
/*
```

**Message queue data set**

Message queue data sets are allocated in DB/DC and DCCTL environments. They are the amount of DASD space allocated to the message queue, and this depends on how many transaction codes and logical terminal names you specify during system definition, and how many short and long messages are to be held by the system during any period.

Allocate message queue data set space in terms of contiguous cylinders for most efficient operation. Secondary allocation is ignored unless the secondary space has been preallocated (that is, multiple volume data sets with preallocated space on both volumes).
You can change the amount of direct access storage space allocated to the message queue data sets before a cold start of IMS. Reallocation of the message queue data sets with a warm start requires the use of the FORMAT and BUILDQ parameters with either the /NRESTART or /ERESTART command. Allocating less space (than in the previous execution) before an /NRESTART or /ERESTART BUILDQ can cause the restart to abend.

You can allocate up to 10 data sets for the long message queue and 10 data sets for the short message queue. Each data set requires an additional DD statement. Ensure that all data sets of a given message queue type are the same size. If the data sets have different sizes, the smallest size is used for all.

Example 2-11 shows a message queue allocation using EAV.

Example 2-11   Message queue allocation using EAV

```
//*************
//*   ALLOCATE MESSAGE QUEUE
//*************
//WADS    EXEC PGM=IEFBR14
//SYSPRINT DD  SYSOUT=*  
//LGMSG    DD  DSN=IMS12Q.IMS12X.LGMSG,UNIT=SYSDA,VOL=SER=EAS001,
//           DISP=(,CATLG),EATTR=OPT,SPACE=(CYL,(25)),
//           DCB=(RECFM=FBS,BLKSIZE=3360,LRECL=6720)
/*
```

2.7.5 Considerations for EAV

Non-VSAM EAV support has the following requirements:

- z/OS Version 1 Release 12 or later
- IBM System Storage DS8000 devices or DS8700 devices configured as 3390 Model A devices

**Important:** Even if you are not going to use the non-VSAM support, IMS 12 requires that you install DFSMS APAR/PTF OA33409/UA55338 on z/OS V1R11.

Data sets with EATTR=OPT specified cannot be shared with an IMS Version 10 or IMS Version 11 system because those IMS versions do not support extended attributes.

2.8 Buffer pools allocation

IMS performance is directly impacted by several variables such as hardware, operating system, application design or even IMS definitions. Virtual storage and the way you allocate buffer pools is one of these variables.

In IMS 12, the storage for certain database pools is now obtained in 31-bit virtual storage backed by 64-bit real storage. Also, IMS 12 has changed the storage requirements for OSAM data extent blocks (DEBs).

As an immediate benefit, large database pools can now be fixed. Previously, these pools were unable to be fixed due to a shortage of 31-bit real storage.
2.8.1 Background on buffer pools

Generally speaking, the way you tune your buffer pools depends directly on how much real storage is available in the environment. In most cases, increasing the size of the pool means increases paging activity. However, reducing pool sizes reduces paging of those areas.

For an IMS system, each application program requires its PSB, the PSBW, the intent list, and the DMBs to be scheduled. These control blocks are loaded into some of the buffer pools existing in the IMS system.

The PSB pool holds all PSBs for scheduled application programs. A specific scheduled PSB remains in the pool until space is needed to load another PSB. Then, if all space is used, the least-referenced inactive PSB is freed.

PSBs that are made resident and are not parallel scheduled are exceptions to this rule. These PSBs do not take from the pool allocation and are usually designated for highly referenced (usually, preloaded) programs. If the DL/I address space option is used, two resident PSB pools exist, one in the z/OS common area and one in the DL/I address space.

A portion of the PSB work pool is required by each active PSB, but the total size does not need to be larger than the maximum. When you use the technique of starting large and reducing to some value greater than the maximum amount used, scheduling stops when any pool space failure occurs.

The DMB pool holds the DMB that describes and controls a physical database. We recommend that you make the DMBs for all databases resident. If all the DMBs are made resident, the inactive DMBs can be allowed to page out without significantly affecting the active DMBs, but paging them back into storage can suspend processing in the control region or the DL/I secondary address space (SAS). Opening and closing databases suspends either the IMS control region or the DL/I SAS, depending on the specification of the LSO= option, and increases DL/I call elapsed time.

Allocate sufficient buffers in the IMS buffer pools to prevent I/O. If the IMS pools are subject to paging, you can consider page fixing the significant buffers, if necessary.

The principal IMS pools that can be page fixed are:
- Message queue pool (QBUF)
- DMB pool (DLDP)
- PSB pool (DLMP)
- Message format buffer pool (MFBP)
- Database subpools

If the DL/I address space option is used, two PSB pools exist: DLMP in the z/OS common area, and DPSB in the DL/I address space.

2.8.2 Description of buffer pools allocation

The BUFP00LS macro statement specifies default storage buffer pool sizes for the DB/DC and DBCTL environments. The storage buffer pool sizes that are specified on the BUFP00LS macro are used unless otherwise expressly stated for that buffer or pool at control program execution time for an online system.
Figure 2-20 shows the syntax of BUFPOOLS macro. For more information, see IMS Version 12 System Definition, GC19-3021.

![BUFPOOLS macro syntax](image)

**DMB**
This specifies the size of the DMB control block pool. The default is 10,000 bytes. The maximum allowable specification is 9,999,000 bytes. The minimum allowable specification is 8 bytes.

**PSB**
This specifies the size of the PSB control block pool if the DL/I address space option is not used. The SASPSB parameter specifies the size of the PSB control block pool when the DL/I address space option is used. The default is 10000 bytes, with a maximum of 9,999,000 bytes. The minimum allowable specification is 8 bytes.

**PSBW**
This specifies the size of the PSB work area pool. The default is 10,000 bytes. The maximum allowable specification is 9,999,000 bytes. The minimum allowable specification is 8 bytes.

**SASPSB**
This is used only if the DL/I separate address space option is selected. If you are not using this option, the size of the single PSB control block pool is specified with the PSB parameter. With the DL/I address space option, two PSB control block pools exist. Size1 is the size of the pool in the Z/OS common storage area (CSA). Size2 is the size of the pool in DL/I local storage. The maximum allowable for either is 9,999,000 bytes.

Using EXEC parameters, you can override definitions for data communication that were initially set during IMS system definition. Table 2-10 shows the cross reference between BUFPOOLS macro definitions and EXEC parameters.

<table>
<thead>
<tr>
<th>BUFPOOLS macro</th>
<th>EXEC parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBWP</td>
<td></td>
</tr>
<tr>
<td>DMB</td>
<td>DMB</td>
</tr>
<tr>
<td>PSB</td>
<td>PSB</td>
</tr>
<tr>
<td>PSBW</td>
<td>PSBW</td>
</tr>
<tr>
<td>SASPSB(size1)</td>
<td>CSAPSB</td>
</tr>
<tr>
<td>SASPSB(size2)</td>
<td>DLIPSB</td>
</tr>
</tbody>
</table>
This specifies the amount of subpool 231 storage to be allocated to the database work area pool. The value can be specified as either one- to six-numeric characters, or one- to five-numeric characters followed by either K (kilobyte), M (megabyte), or G (gigabyte). If K, M, or G is not specified, K is the default. The maximum value is 2 G-1. If the value specified exceeds 2 G-1, the default is 2 G-1. The pool size specified is rounded up to the nearest page boundary.

The DBWP size must be specified as the largest logical record length parameter size of any database you might run.

This specifies the amount of subpool 231 storage to be allocated to the DMB pool. The value can be specified as either one- to six-numeric characters, or one- to five-numeric characters followed by either K (kilobyte), M (megabyte), or G (gigabyte). If K, M, or G is not specified, K is the default. The maximum value is 2 G-1. If the value specified exceeds 2 G-1, the default is 2 G-1. The pool size specified is rounded up to the nearest page boundary. The output of the ACBGEN utility indicates the size of each DMB processed. Examine this output before you specify the DMB= parameter.

This specifies the amount of subpool 231 storage to be allocated to the PSB pool. The value can be specified as one- to six-numeric characters or one- to five-numeric characters followed by K (kilobyte), M (megabyte), or G (gigabyte). If K, M, or G is not specified, K is the default. The maximum value that can be specified is 2 G-1. If the value specified exceeds 2 G-1 the default is 2 G-1. The value specified is rounded up to the nearest page boundary. The output of the ACBGEN utility indicates the maximum PSB size and the size of each PSB processed. Examine this output before you specify the PSB= parameter.

This specifies the amount of subpool 231 storage to be allocated to the PSB work area pool. The value can be specified as one- to six-numeric characters or one- to five-numeric characters followed by either K (kilobyte), M (megabyte), or G (gigabyte). If K, M, or G is not specified, K is the default. The maximum value that can be specified is 2 G-1. If the value specified exceeds 2 G-1 the default is 2 G-1. The pool size specified is rounded up to the nearest page boundary. The output of the ACBGEN utility indicates the maximum work area size among the work area sizes required by each PSB. Depending on the execution environment, the work area size for a given PSB can be increased to the size of the long message queue buffer or to the size specified plus the maximum segment size of the segments processed by this PSB. Examine the ACBGEN output before you specify the PSBW= parameter. This parameter is not needed for an RSR tracking subsystem. For CSAPS and PSBW, DFSIINS0 obtains contiguous space in the z/OS common area. If this storage is unavailable in the z/OS common area, ABENDU0717 occurs.

When the DL/I address space option (LSO=S) is specified, two program specification block pools are used. CSAPS and PSBW, DLIPS and DFSIINS0 obtains contiguous space in the z/OS common area. If this storage is unavailable in the z/OS common area, ABENDU0717 occurs. In an LSO=S system, the PSB= parameter is ignored.
The output of the **ACBGEN** utility indicates maximum CSA space, average CSA space, and the CSA space required by each PSB in the pool. This output is useful in determining what value to specify for the **CSAPSB**= parameter. Before specifying the **DLIPSB**= parameter, consider the SAS size required for each PSB, maximum SAS space, and average SAS. Neither parameter should be zero. Normally, **DLIPSB** should be larger than **CSAPSB**.

The values for the pool sizes can be specified as either one- to six-numeric characters, or one- to five-numeric characters followed by either K (kilobyte), M (megabyte), or G (gigabyte). If K, M, or G is not specified, K is the default. The maximum value is 2 G-1.

The upper limit defaults to 2 G-1 if any of the following statements are true:

- The value specified is not large enough to hold the largest primary and secondary storage allocations that are defined.
- The value specified exceeds 2G-1.
- No value is specified.

The sizes specified are rounded up to the nearest page boundary.

The sizes of the two pools can also be specified during IMS system definition, using the **SASPSB** parameter in the **BUFPOOLS** macro. The sum of the values for the **CSAPSB** and **DLIPSB** defines the PSB pool size. If PSB is also specified, the larger value (PSB or the sum of **CSAPSB** and **DLIPSB**) is used.

For **CSAPSB** and **PSBW**, **DFSIIINS0** obtains contiguous space in the z/OS common area. If this storage is unavailable in the z/OS common area, **ABENDU0717** occurs.

When the DL/I address space option (**LSO=S**) is specified, two program specification block pools are used. **CSAPSB** specifies the size of pool in the z/OS CSA, and **DLIPSB** specifies the size in the DL/I address space. In an **LSO=S** system, the **PSB**= parameter is ignored.

The output of the **ACBGEN** utility indicates maximum CSA space, average CSA space, and the CSA space required by each PSB in the pool. This output is useful in determining what value to specify for the **CSAPSB**= parameter.

Before specifying the **DLIPSB**= parameter, consider the SAS size required for each PSB, maximum SAS space, and average SAS space. Neither parameter should be zero. Normally, **DLIPSB** should be larger than **CSAPSB**. The values for the pool sizes can be specified as either one- to six-numeric characters, or one- to five-numeric characters followed by either K (kilobyte), M (megabyte), or G (gigabyte). If K, M, or G is not specified, K is the default. The maximum value is 2 G-1.

The upper limit defaults to 2 G-1 if any of the following statements are true:

- The value specified is not large enough to hold the largest primary and secondary storage allocations that are defined.
- The value specified exceeds 2G-1.
- No value is specified.

The sizes specified are rounded up to the nearest page boundary.
The sizes of the two pools can also be specified during IMS system definition, using the SASPSB parameter in the BUFPOOLS macro. For the FDR procedure, the sum of the values for the CSAPSb and DLIPSb defines the PSB pool size. If PSB is also specified, the larger value (PSB or the sum of CSAPSb and DLIPSb) is used.

Use the DFSFIXnn member of the IMS PROCLIB data set to specify that portions of the control region (for example, certain control blocks, buffer pools, loaded modules, and part of the IMS nucleus) are to be fixed in address space during.

Figure 2-21 shows the BLOCKS specification syntax. For more information, see IMS Version 12 System Definition, GC19-3021.

**Figure 2-21  DFSFIXnn BLOCKS syntax**

**POOLS**

The dynamic area acquired by IMS during initialization and used for various buffer pools.

**DBWP**

DMB work pool.

**DLDp**

DMB pool.

**DLMP**

With the DL/I address space option, that portion of the PSB pool in the z/OS common area.

**DPSB**

With the DL/I address space option, the portion of the PSB pool in DL/I local storage. Specification is ignored if the DL/I address space option is not in effect.

**GLBP**

The WKPL, DLMP, PSBW, DLDp, and DBWP storage pools are included. If the DL/I address space is used, DLDp and DBWP are not included.
The QBUF and MFBP storage pools are included. If the DL/I address space is used, the DPSB, DLDP, and DBWP storage pools are also included.

**MFBP**
Message format buffer pool.

**PSBW**
PSB work pool.

**QBUF**
Message queue buffer pool.

**WKPL**
General working pool.

### 2.8.3 Buffer pools allocation with IMS 12

In IMS 12, the storage for certain database pools is now obtained in 31-bit virtual storage backed by 64-bit real storage. If you page fix any of these pools and have 64-bit real storage on your processor, you might see a reduction in the use of 31-bit fixed real frames.

The 31-bit virtual storage for the following database pools is now backed by 64-bit real storage:

- DBWP DB work pool
- DLDP DMB pool
- DLMP PSB CSA pool
- DPSB DLI PSB pool
- PSBW PSB work pool

Large database pools that were unable to be page fixed previously due to 31-bit real storage constraints, might now be able to be fixed because the fixed pages are backed by 64-bit real storage.

### 2.8.4 Using buffer pools allocation

To page fix the PSB pool (keep the pool in storage for the long term), specify **POOLS=DLMP** and **POOLS=DPSB** in the DFSFIXxx member in the IMS.PROCLIB data set. To page fix the resident PSB pool, include the module names DFSPSBRS and DFSDLIRS in the DFSFIXxx member.

Example 2-12 shows a sample to fix pools.

**Example 2-12 DFSVSMxx sample to fix pools**

```
NUC=033,POOLS=DBWP,DLDP,DLMP,DPSB,PSBW
```

### 2.8.5 Considerations for buffer pools allocation

The pools listed here are moved from 31-bit real storage to 64-bit real storage. They remain in 31-bit virtual storage.

### 2.9 Miscellaneous other enhancements

IMS 12 also adds other enhancements to improve product serviceability. In this publication we cover the following topics:

- IMS Dump Formatter utility
- EOM/EOT trace
2.9.1 IMS Dump Formatter

Use the Offline Dump Formatter utility (DFSOFMD0) to format internal IMS control blocks in a dump that is both independent of a failure and independent of the dumping process. With this utility, you can tailor the dump to print and format only the data areas needed to analyze a particular problem. Use the Offline Dump Formatter utility to perform the following tasks:

- Establish the environment needed for offline dump formatting
- Read and check the dump format control statements
- Relocate or load the dump formatting modules
- Direct the offline dump formatting process

The Offline Dump Formatter utility is invoked as a verb exit from the Interactive Problem Control System (IPCS).

The Offline Dump Formatter utility modules are included in the dumped storage to ensure that the modules used for formatting the dump match the level of the dumped IMS control blocks. These modules can be relocated from the dumped storage, or a fresh copy can be loaded from the program library.

Using the IMS Dump Formatter gives you a menu-driven way to run the Offline Dump Formatter utility without complicated editing of the DFSFRMAT file. IPCS uses menus to run the IMS Dump Formatter. With these menus, you can specify the information to be contained in the dump. The IMS Dump Formatter calls the Offline Dump Formatter utility to perform the required formatting tasks. The output is returned in a format that you can read on the terminal.

IMS 12 adds support to IMS Dump Formatter for the following areas:

- The Repository Server address space by using the Other IMS Components (6) section of the Dump Formatter
- The Repository Client address space by using the Other IMS-Related Products (7) section of the Dump Formatter
- The OTMA C/I by using the Other IMS Components (6) section of the Dump Formatter

2.9.2 EOM/EOT trace

There are certain situations where an IMS dependent region that was executed as a batch job ended abnormally due to storage constraints. The severe storage conditions within the address space caused the normal IMS dependent region termination clean-up services to fail. In earlier IMS versions, there was no diagnostic information to determine whether the end-of-memory (EOM) service had been invoked or whether the service failed or had been successful. EOM service diagnostic information was introduced in IMS 10 and has been helpful in resolving this problem.

IMS 12 adds support for the end-of-task (EOT) process similar to that employed by the EOM process. This function traces the step-by-step flow through the EOT process by updating a nibble trace double-word in the IDT (IDTEOTTR) of the region with a unique marker for each successful step processed or for each decision point reached and evaluated.
The EOM/EOT Tracing Facility produces a step trace of the actions taken by the EOM service each time it is invoked for a dependent region. Trace data is stored in the region's IDT entry in the following fields:

- **IDTASCBT** - EOM trace: ASCB address stack
- **IDTASIDT** - EOM trace: ASID number stack
- **IDTEOMTR** - EOM trace: 16 nibble stack
- **IDTEOTTR** - EOT trace: 16 nibble stack

The EOM/EOT trace information captured by the service is recorded, using WTO, in message **DFS0798I**, which is issued at the end of the End of Memory call (no action is required):

```
DFS0798I eee PROCESSING COMPLETE FOR jjjjjjjjj RC=0000 RSN=00000000 ASCB=aaaaaaaa ASID=dddd TRC=ttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttt
2.9.3 OSAM data extent blocks

Previous IMS versions created two DEBs for each OSAM data set. The DEB contains information about the physical characteristics of the data set and other information that the control program uses. For database data sets, these DEBs were in the DLISAS region. For system data sets, these DEBs were in the control region.

IMS 12 has changed the storage requirements for OSAM DEBs. IMS 12 eliminates one of the DEBs for each OSAM data set. Because each DEB requires approximately 300 bytes, this reduction might be significant for users with many OSAM database data sets.

2.9.4 Reduced aliases in RESLIB

Various IMS load modules contain aliases, which make them error-prone when service affects these modules. Common practice when implementing selected service is to individually copy modules affected by the service. When modules being copied contain aliases it is easy to overlook this and not propagate the aliases.

IMS 12 starts to remove as many aliases from IMS load modules as possible.
Chapter 3. Database enhancements

Databases in IMS are hierarchical databases. They are fast, reliable, and can take advantage of the IBM z/Architecture®. They can be split into three domains:

- Traditional full function databases (hierarchical indexed sequential access method (HISAM), simple hierarchical sequential access method (SHISAM), hierarchical direct access method (HDAM), and hierarchical indexed direct access method (HIDAM))
- High available databases (partitioned HDAM (PHDAM), partitioned HIDAM (PHIDAM), and partitioned secondary index (PSINDEX))
- Fast Path databases (data entry database (DEDB) and main storage database (MSDB))

IMS 12 has both functional and performance enhancements in the database support area. The performance enhancements benefit from the z/Architecture (64 real and virtual bit support, new instructions), and also offer new implementations that are transparent for existing applications. Functional enhancements introduce additional exploitation capabilities of the existing databases.

This chapter includes the following sections:

- Full function database enhancements
- High availability large database enhancements
- Fast Path database enhancements
- CICS threadsafe support
- Miscellaneous enhancements

For reference documentation that relates to this chapter, see the following publications:

- IMS Version 12 Database Administration, SC19-3013
- IMS Version 12 Exit Routines, SC19-3016
- IMS Version 12 System Utilities, SC19-3023
3.1 Full function database enhancements

IMS 12 provides several improvements for traditional databases. Within this category, the database types HIDAM, HDAM, HISAM, and SHISAM are classified, with or without secondary indexes and logical relationships.

3.1.1 Overview of the database pool storage enhancements

In IMS Version 12, the storage for certain database pools is now obtained in 31-bit virtual storage backed by 64-bit real storage. If you page fix any of these pools and have 64-bit real storage on your processor, you might see a reduction in the use of 31-bit fixed real frames, which can influence system paging.

The 31-bit virtual storage for the following database pools is now backed up by 64-bit real storage:
- DBWP: DB work pool
- DLDP: DMB pool
- DLMP: PSB CSA pool
- DPSB: DLI PSB pool
- PSBW: PSB work pool

Large database pools that were unable to be page fixed before, due to 31-bit real storage constraints, might be able to be fixed because the fixed pages are backed by 64-bit real storage.

You can use the DFSFIXnn member of the IMS PROCLIB data set to specify the buffer pools to be fixed in address space during initialization.

3.1.2 Dynamic full function database buffer pools

With IMS 12, users can add, change, and delete full function buffer pools. This support is provided using new specifications in the DFSDFxxx proclib member in conjunction with the UPDATE POOL command. With this support, full function buffer pools can be managed without restarting IMS.
IMS can internally quiesce application read and update activity so that the **UPDATE POOL** command can complete with little disruption to transaction workloads (Figure 3-1).

![Diagram of IMS Proclib and IMS Control Region](image)

**Figure 3-1  Overview of dynamic full function buffer pools**

In IMS 11 and earlier releases, the Virtual Storage Access Method (VSAM) and overflow sequential access method (OSAM) buffer pool definitions were only stored in the DFSVSMxx proclib member. This member is loaded only one time during IMS initialization. No facility is available to change the buffer pool definitions without first changing the DFSVSMxx member in proclib and then restarting IMS.

IMS 12 offers a new feature for dynamically adding, updating, and deleting VSAM and OSAM buffer pools. The initial VSAM and OSAM buffer pool specifications still exist in the DFSVSMxx proclib member and they are loaded during normal restart. However, new VSAM and OSAM buffer pools can be added and existing buffer pools can be changed using specifications in one or more DFSDFxxx proclib members with the type-2 **UPDATE POOL** command.

Example 3-1 shows the initial and pool specifications for VSAM.

**Example 3-1  VSAM buffer pool specifications in DFSVSMxx**

```plaintext
POOLID=VCCC
VSRBF=2048,8000,I
VSRBF=2048,26000,D
VSRBF=4096,32767,I,HSO,HS40000
VSRBF=8192,32767,I,HSR,HS40000
DBD=PVHDJ5B(B,VCCC,ERASE=YES,FREESPACE=YES)
```
The specifications in Example 3-1 on page 69 can be dynamically changed by using buffer pool specifications in the DFSDFxxx proclib member section (Example 3-2).

Example 3-2 Changed VSAM specifications in DFSDFxxx

```
<SECTION=VSAMEXM>
POOLID=(VCCC,
    VSRBF=(2048,12000,I),
    VSRBF=(2048,12000,D),
    VSRBF=(4096,32767,I,HS0,HS80000),
    VSRBF=(8192,32767,I,HSR,HS80000))
DBD=(PVHDJ5B,B,VCCC,ERASE=YES,FREESPACE=YES)
```

For VSAM the specifications in DFSDFxxx are located behind a `<SECTION=xxxxxxx>` indication. The information is similar to that in DFSVSMxx, except that the “VSRBF” definitions are part of the POOLID statement. The name of the section can be chosen, as it will be indicated during the update process.

OSAM buffers can be treated the same way. Example 3-3 shows the initial pool specifications for OSAM.

Example 3-3 OSAM buffer pool specifications in DFSVSMxx

```
IOBF=(8192,8000,N,N,OCCC)
DBD=POHIDKA(B,OCCC)
```

The specifications in Example 3-3 can be dynamically changed using buffer pool specifications in the DFSDFxxx proclib member section (Example 3-4).

Example 3-4 changed OSAM specification in DFSDFxxx

```
<SECTION=OSAMEXM>
IOBF=(8192,12000,N,N,OCCC)
DBD=(POHIDKA,B,OCCC)
```

For OSAM the specifications in DFSDFxxx are also located behind a `<SECTION=xxxxxxx>` indication. The information is similar to that in DFSVSMxx.

The VSAM and OSAM buffer pool specifications can be placed into different DFSDFxxx members in proclib, because with the `UPDATE POOL` command, the user can specify the MEMBER keyword identifying the suffix of the DFSDFxxx proclib member in the proclib data set.

Alternatively, the user can specify multiple VSAM and OSAM sections within one or more DFSDFxxx members.

The update of the pool specifications is done with the type-2 `UPDATE POOL` command identifying the statement sections. The `UPDATE POOL` command can be issued individually for specific VSAM and OSAM sections, or the command can be issued for both VSAM and OSAM sections in the same command. The `UPDATE POOL` command can also reference a specific DFSDFxxx proclib member in the proclib data set using the MEMBER(yyy) keyword.

In this case, yyy is the suffix used in DFSDFyyy. The default for yyy is 000.
Example 3-5 shows **UPDATE** command examples.

**Example 3-5  Add and change commands**

```sql
#Add or Change VSAM or OSAM buffer pool definitions:
UPDATE POOL TYPE(DBAS) SECTION(OSAMxxx)
UPDATE POOL TYPE(DBAS) SECTION(VSAMxxx)

#VSAM and OSAM buffer pool change in one command:
UPDATE POOL TYPE(DBAS) SECTION(OSAMxxx,VSAMxxx)

#Add or Change definitions in an alternate DFSDFyy proclib member
# notice the “member” keyword
UPDATE POOL TYPE(DBAS) SECTION(OSAMxxx) MEMBER(yyy)
```

It is possible to delete a VSAM buffer pool by specifying a **POOLID** in a VSAM section with the **VSRBF** statement for the size of the buffer and a “0” for the number of buffers. The **UPDATE POOL** command is needed to complete the deletion of the VSAM buffer pool.

The database data set association with a subpool is established when the database data set is opened. If a database data set using a subpool is to be deleted, the **UPDATE POOL** command must wait until the access to the subpool is completed before it can delete the subpool.

The **QUERY POOL** command (Example 3-6) can be used to query information about the new and changed VSAM and OSAM buffer pools. The user can specifically limit the output to:

- OSAM or VSAM buffer pools
- Buffers of a particular size
- Specific pool IDs

**Example 3-6  Type-2 QUERY command**

```sql
QUERY
POOL
  TYPE(DBAS)
  SUBTYPE(OSAM,VSAM) SIZE() POOLID()
  SHOW( ALL/MEMBER/STATISTICS)
```

By using the options for **SHOW**, you can show only statistical information that is similar to the current **/DIS POOL DBAS** command. Alternatively, you can show the proclib member information used to add or update a buffer pool specification. It is also possible to show both statistical and member information using the **ALL** parameter. Example 3-7 shows a Time Sharing Option (TSO) single point of control (SPOC) example. The command can also be issued by an **OM API** command, in which case the output is XML tagged.

**Example 3-7  TSO SPOC samples**

**TSO SPOC input:**

```sql
QUERY POOL TYPE(DBAS) SUBTYPE(OSAM,VSAM) SHOW(MEMBER)
```

**TSO SPOC output:**

<table>
<thead>
<tr>
<th>Subpool</th>
<th>MbrName</th>
<th>CC BufSize</th>
<th>PoolId</th>
<th>NBuf</th>
<th>ProcMbr</th>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>OSAM</td>
<td>I12A</td>
<td>0 2048</td>
<td>5</td>
<td>DFSVSM2A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OSAM</td>
<td>I12A</td>
<td>0 8192</td>
<td>5</td>
<td>DFSVSM2A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VSAM-D</td>
<td>I12A</td>
<td>0 512 XXXX</td>
<td>5</td>
<td>DFSVSM2A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VSAM-D</td>
<td>I12A</td>
<td>0 1024 XXXX</td>
<td>5</td>
<td>DFSVSM2A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VSAM-D</td>
<td>I12A</td>
<td>0 2048 XXXX</td>
<td>5</td>
<td>DFSVSM2A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VSAM-D</td>
<td>I12A</td>
<td>0 4096 XXXX</td>
<td>5</td>
<td>DFSVSM2A</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
When VSAM or OSAM buffers are updated or deleted, the system is obliged to execute quiescing activity to implement the new values. This differs for VSAM and OSAM, as explained here:

- For VSAM, when the **UPDATE POOL** command is issued to add, change, or delete one or more VSAM buffer pools, IMS must internally quiesce application activity against the affected buffer pools. The **UPDATE POOL** command must wait until all update activity has been committed and hardened to the database data sets.

- For OSAM, when the **UPDATE POOL** command is issued to add, change, or delete one or more OSAM buffer pools, IMS must wait for the use of the affected buffers by applications to go down to zero. Before the usage count is zero, applications can use the buffers affected by the **UPDATE POOL** command. After the usage count goes to zero, the buffer can be reconfigured.

The buffer pool statistics are handled differently for VSAM and OSAM following an **UPDATE POOL** command. For VSAM, the buffer pool statistics are reset and the old statistics are not carried over. Use the **QUERY POOL** command for the VSAM buffer pool statistics before issuing the **UPDATE POOL** command. The OSAM statistics are carried over and are not reset with the **UPDATE POOL** command.

Committed buffer pool changes are written to restart data set (RDS).

- Emergency restart restores buffer pools using RDS.
- Normal restart initializes buffer pools from DFSVSMxx.

The **UPDATE POOL** command logs information in the x’22’ log record for informational purposes only. The **UPDATE POOL** command itself is non-recoverable.

### 3.2 High availability large database enhancements

Within the group of high availability large databases (HALDBs), the PIHDAM, PHDAM and PSINDEX database are classified. This group gained specific improvements in IMS 12. Figure 3-2 shows an overview of HALDBs, although they are not new in IMS 12.

![Figure 3-2 HALDB databases](image-url)
Each partition must have a name of maximum 7 characters, so that appending the suffixes as indicated in Figure 3-2 makes it possible to distinguish the data sets within the partition. The data sets in a partition have generated data set names and DDNAMEs. The letters are used to distinguish the data sets as shown here:

- **Y/X**: PHIDAM index
- **L**: Indirect list data set (ILDS)
- **A through J**: Database data sets
- **M through V**: Database data sets
- **A**: PSINDEX

For HALDB, no dynamic allocation members are used; instead, all HALDBs must be registered in the recovery controls (RECONs). The dynamic allocation is based on information in the RECON and the partition ID that is stored in the RECON and in the “A” data set of the partition for verification. This partition ID is defined by the creation pattern of the HALDBs and its value has no sequence meaning.

DDNAMEs are defined by the partition name appended with the functional letters as shown in Figure 3-2. Data set names contain a prefix, chosen by the user, followed again by the functional letter and a partition ID. All allocations are dynamic, except for a few areas where the DDNAMEs must be used.

HALDB databases can be reorganized online, and have a HALDB self-healing pointer process.

### 3.2.1 HALDB online reorganization ownership release

IMS 12 adds the capability to release ownership of an online reorganization (OLR) when IMS terminates. The IMS termination can be normal or abnormal. In previous versions, OLR ownership was kept by a terminated IMS system.

In IMS 12, the `RELOLROWNER=Y/N` parameter in the `<DATABASE>` section of the DFSDFxxx PROCLIB member indicates whether OLR is restarted on the non-owning IMS system (Example 3-8).

**Example 3-8 Ownership release indication**

```plaintext
<SECTION=DATABASE>
RELOLROWNER=Y|N
```

`RELOLROWNER=N` is the default and does not release ownership when the IMS system terminates. The `RELOLROWNER=` value can be overridden by specifying one of the following values on the `INIT OLREORG`, `/INIT OLREORG`, `UPD OLREORG`, or `/UPD OLREORG` command:

- `OPTION(REL)`
- `OPTION(NOREL)`

When `RELOLROWNER=Y` or `OPTION(REL)` is not specified, OLR is automatically restarted when the terminated IMS system is restarted.

When `RELOLROWNER=Y` is specified, OLR is not automatically restarted unless it was overridden with `OPTION(NOREL)` on the command. If the OLR is not automatically restarted by IMS restart, it must be restarted with the `INIT OLREORG` or `/INIT OLREORG` command.
3.2.2 Parallel migration to HALDB: IMS 10 and IMS 11 small programming enhancement

The IMS Unload utility (DFSURGU0) has been enhanced in IMS 10, IMS 11, and IMS 12 to allow unloads of key ranges of an HDAM or HIDAM database when migrating to HALDB. Multiple unloads for the same database can be run in parallel. This can significantly reduce the elapsed time for a migration to HALDB.

An execution of HD Unload can be restricted to a range of keys when the MIGRATE=YES control statement is included for migration to HALDB. The low (FROMKEY) and high (TOKEY) values must be specified in hexadecimal. These values can be obtained from the output of a LIST.DB DBRC command for the partition.

Before this enhancement the migration unload of a logically related database often took a long time. When a logical child segment is unloaded, its logical parent must be read in most cases. This is a random read. These random reads account for almost all of the elapsed time of the unload.

The logically related database must be read unless the unload is for the physical logical child of a bidirectional logical relationship with virtual pairing when the PHYSICAL option was specified the database description (DBD) to include the concatenated key in the physical logical child. The logical related database is always read with the following logical relationships:

- Unidirectional logical relationships
- Physically paired bidirectional relationships
- Virtually paired bidirectional logical relationships when reading the virtual logical child
- Virtually paired bidirectional logical relationships when reading the physical logical child and the VIRTUAL option has been specified in the DBD (the concatenated key of the logical parent is not stored in the physical logical child).

Figure 3-3 illustrates the improved process.

![Parallel unload diagram](image)

**Figure 3-3  Parallel unload**

The four HD Unload jobs process different key ranges in the non-HALDB database. They are run in parallel. Their individual outputs are fed to four different HD Reload jobs which load the four partitions in the new HALDB database. The reload jobs are also run in parallel.
Multiple executions of HD Unload can be run in parallel. The key ranges must correspond to the key ranges for HALDB partitions so that the subsequent HD Reload executions can be run in parallel. Use “cut and paste” for the hexadecimal key values from a RECON listing.

Example 3-9 shows the control input by using SYSIN DD.

Example 3-9  Control parallel unload

| MIGRATE=YES  |
| KEYRANGE FROMKEY(key value) TOKEY(key value) KEYLEN(key length) |
| Recommendation: Cut and paste key values from RECON listing, Example: |
| MIGRATE=YES |
| KEYRANGE FROMKEY(F0F0F0F0F0) TOKEY(D2F2F0F0F0) KEYLEN(5) |

MIGRATE=YES must be specified.

The new capability cannot be used for unloading HALDB databases. Also, MIGRATX=YES cannot be used with the KEYRANGE statement. MIGRATX=YES is used only for databases with secondary indexes. It cannot be used with the KEYRANGE statement because the key ranges appropriate for a secondary index would not be those used for the unload of the indexed database. If your database has secondary indexes, you can use MIGRATE=YES and create the secondary indexes with a tool such as IBM IMS Index Builder.

The records unloaded must match the HALDB partition boundaries. This is true because only one output data set from HD Unload can be used as input to HD Reload. Similarly, all of the records for a partition must be unloaded with one execution of HD Unload.

The job control language (JCL) must specify DISP=SHR for the unloads to run in parallel. If the data sets are VSAM, you must also specify share options that allow concurrent reads. SHAREOPTIONS(3 3) can be used for this purpose.

This capability was retrofitted into IMS 10 (APAR PM06635) and IMS 11(APAR PM06639).

3.2.3 Reuse of HALDB partition database names

The improvement regarding the reuse of HALDB partition database names is about partition names, not database names. Each partition of a HALDB has a name. The name must be unique among all partitions and database names in a set of RECONs. Describing the data in the partition name can cause problems because the name can lose significance if the volume of data grows.

Before IMS Version 12, when a HALDB partition was removed from the HALDB master and information about the partition was removed from the RECON data set, IMS retained information about the deleted partition name and prevented the name from being reused for a non-HALDB database.

IMS 12 now discards the residual information about the partition name so that it can be reused for a non-HALDB database. When a HALDB master is deleted, all of its partition names become available for reuse.

3.2.4 Reorganization numbers handled differently by timestamp recovery

Reorganizations of HALDB databases with logical relationships and secondary indexes do not require the execution of utilities to update pointers. Instead, HALDB uses a self-healing pointer process to correct logical relationship and secondary index pointers.
This process is implemented by placing a target key and an extended pointer set (EPS) in the secondary index or logically related database and by using an ILDS in each partition of PHDAM and PHIDAM databases.

The partition reorganization number is used to ensure that secondary index and logical relationship pointers are accurate. The reorganization number for a partition is stored in the partition data set. It is incremented by each reorganization of the partition. The reorganization number is also stored in the EPS of secondary index entries and logical children. If the value in the EPS does not match the value in the partition data set, the pointer is healed by updating it from the ILDS. The reorganization number is also stored in the partition database record in the RECON.

When a timestamp recovery is done to a time before the last reorganization, the reorganization number in partition data set is returned to its previous value by the actions of the Database Recovery utility (DFSURDB0) in previous versions of IMS. IMS 12 changes this. The IMS 12 Database Recovery Utility takes the reorganization number from the RECONs, increments it, and stores it in both the RECONs and the partition data set. This makes the reorganization numbers in the partition data set and the RECON match. Previously, a mismatch occurred until the first update job for the partition was executed. An update batch job or online system takes the value from the RECON and writes it to the partition data set.

Two problems can occur with previous releases of IMS:

- If a user created indexes by using the Index Builder tool after a timestamp recovery, the EPSs in the index entries might contain the reorganization values from the partition data set. These do not match the values in the RECON. After the first update job, the reorganization number in the partition data set is updated, which caused a mismatch with the values in the EPSs. When the index entries were used, IMS healed the pointers, which was unnecessary overhead. The new process eliminates this healing because the values in the EPSs match the values in the partition data set.
- The second problem occurred with the Pointer Checker tool. When the values in the pointers do not match those in the partition data set, the Pointer Checker produces error or warning messages. The new process eliminates these messages.

### 3.3 Fast Path database enhancements

Fast Path databases also received a lot of attention in IMS 12, the most important enhancement is the availability of secondary indexes for DEDBs.

#### 3.3.1 Reduced logging for DEDB changed data capture

Some users want to use asynchronous changed data capture, however they do not want to write log records for before images. Prior to IMS 12, asynchronous changed data captures writes “before” and “after” image log records (x’99’). With IMS 12, you can specify that these before image log records are not to be written.

This is specified with new values on the EXIT= parameter of the DBD or SEGM macro of DBDGEN (Example 3-10).

**Example 3-10  DBDGEN exit parameters**

```
EXIT=((exitname),LOG/NOLG,BEFORE/NOBEBEFORE,DLET/NODLET)
```
The following values are new:

- **DLET**  
  Writes the before image log record for DLET calls. This is the default. It is also the action taken by previous versions of IMS.

- **NODLET**  
  Does not write the before image log record for DLET calls.

- **BEFORE**  
  Writes the before image log record for REPL calls. This is the default. It is also the action taken by previous versions of IMS.

- **NOBEFORE**  
  Does not write the before image log record for REPL calls.

### 3.3.2 Full segment logging option

ISRT always logs the entire segment. DLET logs the data which cleared out the data. REPL logs the changed data.

IMS 12 provides new options to log entire DEDB segments when a REPL call replaces some of the data in a segment. Previously, only the changed data was logged in the x'5950' log record. The option to log the entire segment is specified for either the DEDB database or the area.

Example 3-11 shows the new keywords for the `INIT.DB`, `CHANGE.DB`, `INIT.AREA`, and `CHANGE.AREA` DBRC commands.

**Example 3-11   DBRC commands for full REPL segment logging**

```plaintext
INIT.DB DBD(name) NOFULLSEG|FULLSEG …
CHANGE.DB DBD(name) NOFULLSEG|FULLSEG …
INIT.DB DBD(name) AREA(aname) NOFULLSEG|FULLSEG …
CHANGE.DB DBD(name) AREA(aname) NOFULLSEG|FULLSEG …
```

- **FULLSEG** indicates that the entire segment is to be logged.
- **NOFULLSEG** indicates that only changed data is logged. The default is to log only the changed data.

If `FULLSEG` is specified on the `INIT.DB` or `CHANGE.DB` command without the `AREA(aname)` specified, the entire segment will be logged for all segments in the database unless `NOFULLSEG` is specified for an area with either the `INIT.DB` or `CHANGE.DB` command with `AREA(aname)` specified.

`AREA FULLSEG` takes precedence over `DB FULLSEG`.

The new option is especially useful for users of the Log Archive exit who want access to the entire segment for REPL calls. Without this enhancement, users must invoke Asynchronous Changed Data Capture. It writes x’99’ log records in addition to the x’5950’ log records.

### 3.3.3 Improved diagnostic message for DEDB data sharing

IMS 12 includes a new diagnostic message for data sharing. The new `DFS0066I` message is issued when a data sharing system responds to a notify message.

Notify messages are used to send messages between data sharing IMS systems. For example, they are sent to synchronize the initialization of the use of unit of work (UOW) locks. These messages are sent by one IMS system to all of its data sharing partners. All partners must respond before the processing can continue. IMS sends a `DFS3770W` message if all partners do not respond within a time limit. The `DFS3770W` tells the user that the timeout situation has occurred but does not identify which IMS system has not responded.
IMS 12 uses the new DFS0066I message to assist the user (Example 3-12). This message is sent by all systems responding to the notify message.

Example 3-12  Acknowledgement from other data sharing IMS

DFS0066I A NOTIFY RESPONSE HAS COME BACK FROM imname

If a DFS3770W message is issued, the user can identify the system that has not responded by determining for which system the DFS0066I message has not been sent.

All DFS0066I messages are issued by the IMS system that sent the original notify message. This new message makes it easier for users to identify the failing system and to resolve the problem more quickly. You might need to cancel the IMS that has not issued this message.

3.3.4 Fast Path DEDB secondary index support

Secondary indexing provides a way to meet the various processing requirements of various applications. With secondary indexing, you can have an index based on any field in the database, not simply the key field in the root segment. Secondary indices were previously only available for full-function and HALDB databases.

In IMS 12, you can have a secondary index on a primary DEDB database to process a segment type in a sequence other than the one defined by the segment’s key. A DEDB database can be composed of 1 (optional) sequential dependent segment (SDEP) and up to 127 direct dependent (DDEP) segments, spread over up to 15 levels.

You can have a maximum of 32 secondary indexes per segment and 255 secondary indexes per DEDB. Secondary indexes are only supported for DDEPs, and not for the SDEP. The secondary index databases must be “root-only” VSAM-based HISAM or SHISAM databases. HISAM databases must be used if you are confronted with duplicate secondary index values. A SHISAM database is composed of 1 KSDS data set. A HISAM database can have overflow in an ESDS data set.

Having duplicate secondary index values, if too many, can provoke performance problems. Techniques can be used to avoid the creation of duplicate keys by concatenating the logical key value with generic /CK values, thereby making the real stored physical key value unique.

Creation of the secondary index pointers can be suppressed by using an exit routine or a NULLVAL parameter; this is known as “sparse” secondary indexing.

When designing a secondary index on a database, you must deal with three segments:

**Target segment** This is a segment in the primary DEDB, regardless of whether it is the root, that you try to access by using the secondary key.

**Source segment** This is a segment in the primary DEDB that is the source for building the search (secondary) key. The source segment can be the target, or it can be a dependent of the target.

**Index (pointer) segment** This is the root segment of the secondary index; the starting point for the alternative processing access.
Figure 3-4 illustrates these aspects.

The secondary index relationship must be described by LCHILD/XDFLD macro statements in the database descriptors of the primary and the index databases. The secondary index key is built from up to five fields in the source segment. These fields do not have to be adjacent in the source segment. They can be placed in any order in the secondary index key.

The pointer from the secondary index to the DEDB is a symbolic pointer. That is, it is the concatenated key of the target segment. By using a symbolic key the secondary index is not affected by reorganizations of the DEDB.

Indices are updated when a source segment is inserted, deleted, or replaced. For the creation of secondary indexes on an existing DEDB, however, a tool (such as IBM IMS Index Builder) or program is required to add a secondary index to an existing database.

**Indexing when target is root segment**

When the target segment is the root segment, all segments in the DEDB are accessible through the secondary index. This includes SDEP segments.

When accessing a segment through the secondary index, the key feedback area uses the secondary index key for the key of the target segment (the root) and uses the dependent segment keys when they are accessed. The concatenated key in the key feedback area is composed of the secondary index key and the keys of the dependent segments.

Figure 3-5 on page 80 illustrates the physical structure of a database and the structure as viewed when accessing the database through the secondary index. In this case, they are the same because the root segment is also the target segment.
The key feedback area is composed of the secondary index key and the keys of the dependent segments. The key feedback area for segment C is composed of the secondary index key, the key of segment B and the key of segment C.

**Indexing when target is dependent segment**

If the target segment is not a root segment in the primary DEDB database, the hierarchy in the database record is conceptually restructured as an inverted structure. The DEDB inverted structure access is limited to a subset of segments.

For DEDB inverted structure access, the target segment, its direct parent segments from the target segment to the root segment, and all its child segments from the target segments are accessible. SDEPs are not accessible through the secondary index. When the target segment is not the root, SDEPs are dependents of the root. They have no children and cannot be source segments. This means that they cannot be target segments.

**Important:** Fast Path is different from full function. With full function, you have access to all children of the root segment, even those that are not direct parents or dependents of the target segment. The DEDB inverted structure access is limited to a subset of segments.

Figure 3-6 on page 81 illustrates the physical structure of a database and the structure as viewed when accessing the database through the secondary index. In this case, the target is not the root segment. The target is segment G.

The structure as seen from the secondary index path has the target, segment G, as the root. Segment D is the parent of G in the physical structure, so it is the first parent in the secondary structure. Its parent is A, so it is a dependent of D. Segments H and I are the only children of segment G in the physical structure. They are also children in the secondary structure.

The illustration shows the key feedback areas for segments A and H. The key feedback area for A includes the secondary index key, the key of segment D, and the key of segment A. The key feedback area for H includes the secondary index key and the key of segment H.
SENSEG statements in program specification block

When accessing a database through the secondary index, the SENSEG statements in program specification block (PSB) must be in the physical structure order. This is different from the requirements for full function databases.

Attention: With full function databases, the SENSEG statements must be in the secondary structure order.

In Figure 3-7, the secondary structure order is G, D, A, H, and I. Nevertheless, the SENSEG statement order is A, D, G, H, and I. This matches the physical structure order for these segments.

When accessing a database through the secondary index, the SENSEG statements in PSB must be in the physical structure order of the DEDB.

This is different from the requirements for full function databases. With full function databases, the SENSEG statements must be in the secondary structure order. In this
illustration the secondary structure order is G, D, A, H, and I. Nevertheless, the SENSEG statement order is A, D, G, H, and I. This matches the physical structure order for these segments.

**Attention:** Even though the SENSEG statements in the PSB are in physical structure order, the segments are accessed in the secondary structure order when accessing the database through the secondary index. For example, a program using unqualified GN calls accesses segments in the order G, D, A, H, and I, which does not match the order of the SENSEG statements of A, D, G, H, and I.

The order of the segments in the PCB is different, and the order in which the programmer accesses them is unique with DEDB secondary indexes.

**DBDGEN for DEDB with secondary index**

To include secondary indexes for a DEDB you must specify an LCHILD and XDFLD statement for each secondary index (Example 3-13).

**Example 3-13  Secondary indexes for a DEDB**

```
LCHILD NAME=(sisegname,sidbname),PTR=SYMB
XDFLD NAME=searchname,SEGMENT=segmname, x
   SRCH=(fldname1,fldname2,), x
   DDATA=(fldnamea,fldnameb,), x
   SUBSEQ=(fldnamex,fldnamey,), x
   NULLVAL=value,EXTRTN=rtnname
```

- The LCHILD statement follows the SEGM statement for the segment that is the target of the secondary index. The NAME= parameter specifies the segment in the secondary index and the secondary index database name. The PTR= parameter must specify SYMB. This indicates that the pointer is symbolic; the pointer is the concatenated key of the target segment.
- The XDFLD statement follows the LCHILD statement. The XDFLD statement defines the field or fields on which the secondary index is built. If the source segment is not the target segment, it is specified in the SEGMENT= parameter. The source segment can be either the target segment or one of its dependents. If SEGMENT= is not included, the source segment is the target segment. The field or fields on which the secondary index is built are defined in the SRCH= parameter. Up to five fields can be specified. All fields must be in the same segment. The DDATA=, SUBSEQ=, and NULLVAL= parameters are optional.
  - DDATA= defines duplicate data fields. These are fields in the source segment that are also included in the secondary index segment. They are available to programs that process the secondary index as a database.
  - SUBSEQ= defines subsequence fields. These fields are used to sequence secondary index segments that have the same secondary index key. Concatenated key fields (/CKxxxxx) fields can be specified as SUBSEQ= fields, which is explained in the following section.
  - Index suppression is optional. It can be specified in two ways, with the NULLVAL= parameter and with the EXTRTN= parameter. Either, neither, or both can be specified. NULLVAL= is a one-byte field. If it is specified, a SRCH= value that has this value in all of its bytes will cause index suppression. Index suppression causes no index entry to be created for the source segment. If multiple fields are specified in SRCH=, all must have this value for index suppression to be invoked.
EXTRTN= specifies an Secondary Index Database Maintenance exit routine. The exit routine is passed the source segment and determines if an index entry should be built for the segment.

- The NAME= parameter specifies the name that can be used to segment search arguments (SSAs) to qualify calls on the value of the secondary index.

DBDGEN for secondary index

The DBD for the secondary index requires ACCESS=(INDEX, VSAM) for a HISAM secondary index or ACCESS=(INDEX, SHISAM) for a SHISAM secondary index (Example 3-14).

Example 3-14 Secondary index DBDGEN input

| ACCESS=(INDEX, VSAM) is used when HISAM database is index |
| DBD NAME=indexname, ACCESS=(INDEX, VSAM), FPINDEX=YES |
| ACCESS=(INDEX, SHISAM) is used when SHISAM database is index |
| DBD NAME=indexname, ACCESS=(INDEX, SHISAM), FPINDEX=YES |
| Non-unique keys require overflow data set specification on DATASET statement |
| DATASET DD1=ddname1, OVFLW=ddname2 |
| SHISAM supports only unique index keys |
| Overflow data set is never specified on DATASET statement |
| DATASET DD1=ddname1 |

**Tip:** You do not specify ACCESS=(HISAM, VSAM) or ACCESS=(SHISAM, VSAM) for HISAM and HISAM secondary indexes.

- The secondary index has only one SEGM statement. This statement specifies the segment name, PARENT=0, and the size (BYTES=) of the segment. The segment size includes the size of the secondary index key plus the size of the concatenated key of the target segment. It also includes the size of any subsequence and duplicate data fields. It can be larger than the sum of these fields. If it is larger, the remaining space can be used for user data.

- One or more FIELD statements must be included. The secondary index key is defined with SEQ included in the NAME= parameter. If the key including the subsequence field is unique, specify U. If it is not unique, specify M. The BYTES= parameter specifies the size of the sequence field. This includes the search fields and subsequence fields. Always specify START=1.

- The LCHILD statement NAME= parameter specifies the target segment name and the target database. The INDEX= parameter value must match what is specified in the NAME= parameter on the XDFLD statement in the target database. PTR=SYMB must be specified. See Example 3-15.

Example 3-15 Other secondary index-related statements

```
* SEGM NAME=segmname, PARENT=0, BYTES=segsize
* FIELD NAME=(fldname, SEQ, U|M), BYTES=fldsize, START=1
* LCHILD NAME=(targetsegm, targetdb), INDEX=xdfldname, PTR=SYMB
```

Secondary indexes can have either unique or non-unique keys.

- HISAM supports both unique and non-unique keys.
- SHISAM requires unique keys.
Subsequence fields (\texttt{SUBSEQ=} on the XDFLD statement) can be used to create unique keys. One way to create unique keys is by using the concatenated key of the source segment as a subsequence field. This is done by including a FIELD statement in the DEDB segment whose name begins with /CK and including this field in the \texttt{SUBSEQ=} parameter of the XDFLD statement. The /CK field does not guarantee uniqueness if the keys of the source segment and its parents are not unique.

The /SX field: Secondary indexes for full function databases can use a /SX field to create unique keys. The /SX field is either the four-byte relative byte address (RBA) of the source segment or the 8-byte indirect list key (ILK) of the segment. The ILK is used by HALDB. The /SX field cannot be used with DEDBs. DEDB segments do not have an ILK, and they can be changed by a replace (REPL) call.

Duplicate data can be included in a secondary index by including \texttt{DDATA=} on the XDFLD statement. Duplicate data is only available to application programs when they process the secondary index as a \textit{database}. It is not available to them when they use the secondary index to process the DEDB.

SHISAM does not support CI reclaim with data sharing. Therefore, when all index entries from a large range of keys are deleted, data sharing users might prefer to choose HISAM even when the secondary index has unique keys.

- HISAM secondary indexes support non-unique keys. They require an overflow data set. When using non-unique keys, the DATASET statement includes both the \texttt{DD1=} parameter and the \texttt{OVFLW=} parameter. They specify the DD names for these data sets. If unique keys are used, the DATASET statement specifies only the \texttt{DD1=} parameter. Unique keys are supported with both HISAM and SHISAM.

When accessing a segment through the secondary index, the key feedback area uses the secondary index key for the key of the target segment. The key feedback area uses the keys of the other segments when they are accessed. The concatenated key in the key feedback area is composed of the secondary index key and the keys of the other segments in the path from the secondary index.

Figure 3-8 illustrates this concept.
COURSE is the root segment of a DEDB, and course number is a unique key for this segment. This is the primary access. Consider a scenario where we want to add two secondary entries:

- On the root segment, COURSE, we want a secondary entry by using the INSTRUCT segment as entry point with a phonetic search key INSTPHNO. The COURSE segment is the target and segment INSTRUCT is the source. We eliminate duplicates in the secondary index built by including the concatenated key of INSTRUCT, by using a /CK field.

- We implement an access to the CLASS segment, which is the target segment, by using the student segment, which is the source segment. The student key is used to access the courses, taken by a student. In this case, we accept duplicates. Therefore, we must use an HISAM database with overflow.

**DBDGEN example A: Root is target with a dependent as source**

The example in Figure 3-9 shows a secondary index where the root is the target with a dependent as the source segment:

- The COURSE segment is the target.
- The INSTRUCT segment is the source.

The index is built on the INSTPHNO field. In this example, a /CK field is used to create unique keys.
Figure 3-10 shows the DBD for the DEDB, where:

- Segment COURSE is the target segment.
- Segment INSTRUCT is the source segment.

| DBD1  | DBD NAME=EDUCATDB, ACCESS=DEDB, RMNAME=RMODE3 |
| AREA  | DD1=EDAREA1, SIZE=1024, UOW=(100,10), ROOT=(236, 36) |
| SEGM  | NAME=COURSE, PARENT=0, BYTES=100 |
| FIELD | NAME=(COURSENO, SEQ, U), BYTES=5, START=1 |
| FIELD | NAME=COURNAME, BYTES=10, START=6 |
| LCHILD | NAME=(INSTXSEG, INSTSXDB), PTR=SYMB |
| XDFLD | NAME=XINST, SEGMENT=INSTRUCT,  
|       | SRCH=INSTPHNO, SUBSEQ=/CKAAAAA |
| SEGM  | NAME=CLASS, BYTES=50, PARENT=COURSE |
| FIELD | NAME=(CLASSNO, SEQ, U), BYTES=6, START=1 |
| SEGM  | NAME=INSTRUCT, BYTES=50, PARENT=CLASS |
| FIELD | NAME=(INSTNO, SEQ, U), BYTES=6, START=1 |
| FIELD | NAME=INSTPHNO, BYTES=10, START=7 |
| FIELD | NAME=/CKAAAAA, BYTES=15, START=1 |
| SEGM  | NAME=STUDENT, BYTES=50, PARENT=CLASS |
| FIELD | NAME=(STUDNO, SEQ, U), BYTES=10, START=1 |
| FIELD | NAME=STUDPHNO, BYTES=10, START=11 |
| FIELD | NAME=STUDNAME, BYTES=20, START=21 |

The LCHILD and XDFLD statements follow the SEGM statement for COURSE. The LCHILD segment specifies the secondary index segment, INSTXSEG, and database, INSTSXDB. PTR=SYMB is specified, as required.

The XDFLD statement specifies the name of the search field, XINST, for use with the secondary index. Because the source segment is not the target segment, the following parameters are used:

- **SEGMENT=INSTRUCT** is specified to indicate that INSTRUCT is the source segment.
- **SRCH=INSTPHNO** specifies that field INSTPHNO is used to build the search field.
- **SUBSEQ=/CKAAAAA** is used to create a unique key.
A FIELD statement with \texttt{name=/CKAAAAA} is included so that the XDFLD \texttt{subseq=} parameter can reference it (Figure 3-11).

\begin{verbatim}
DBDSX DBD NAME=INSTSXDB,ACCESS=(INDEX,VSAM)
DATASET DD1=INSTKSDS
SEGM NAME=INSTXSEG,PARENT=0,BYTES=30
FIELD NAME=(INSTSKEY,SEQ,U),BYTES=25,START=1
LCHILD NAME=(COURSE,EDUCATDB),INDEX=XINST,PTR=SYMB
DBDGEN
\end{verbatim}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{dbd.png}
\caption{DBD for index with target ROOT, source INSTRUCT}
\end{figure}

Figure 3-11 shows the DBD for the secondary index database, which is HISAM. Therefore, \texttt{access=(INDEX,VSAM)} is specified.

- \texttt{ddi=} on the DATASET statement specifies the DD name of the secondary index data set. Because the key is unique, the \texttt{ovflw=} parameter is not specified on the DATASET statement.

- The SEGM statement defines the segment in the secondary index. Its size is 30 bytes. It is composed of the secondary index key (10 bytes), the concatenated key of the INSTRUCT segment (\texttt{/CKAAAAA} field which is 15 bytes), and the symbolic pointer to the target (5 bytes).

- The FIELD statement defines the sequence field in the secondary index. Because the keys are unique, \texttt{u} is specified in the \texttt{name=} parameter. The size of the sequence field is 25 bytes. This is the size of the \texttt{instphno} field (10 bytes) in the INSTRUCT segment and the \texttt{/CKAAAAA} field (15 bytes).

- The LCHILD statement specifies the target segment, INSTRUCT, and database, EDUCATDB, in the \texttt{name=} parameter. The \texttt{index=} parameter specifies the \texttt{name=} value on the XDFLD statement of the target database. This is XINST. \texttt{ptr=SYMB} is required for Fast Path secondary index databases.
**DBDGEN example B: Target and source are two dependent segments**

Figure 3-12 shows a secondary index where the target segment is a dependent segment, and another dependent segment is the source:

- The CLASS dependent segment is the target.
- The STUDENT segment is the source.

![Diagram of DBDGEN example](image)

The index is built on the STUDPHNO field. We do not use a /CK field, so we can have synonyms for the key.

Figure 3-13 shows the DBD for the DEDB.

```
DBD1   DBD   NAME=EDUCATDB,ACCESS=DEDB,RMNAME=RMOD3
AREA   DD1=EDAREA1,SIZE=1024,UOW=(100,10),ROOT=(236,36)
SEGM   NAME=COURSE,PARENT=0,BYTES=100
FIELD  NAME=(COURSENO,SEQ,U),BYTES=5,START=1
FIELD  NAME=COURNAME,BYTES=10,START=6
SEGM   NAME=CLASS,BYTES=50,PARENT=COURSE
FIELD  NAME=(CLASSNO,SEQ,U),BYTES=6,START=1
LCHILD NAME=(STUDXSEG,STUDSXDB),MULTISEG=YES,PTR=SYMB
XDFLD  NAME=XSTUD,SEGMENT=STUDENT,SRCH=STUDPHNO
SEGM   NAME=INSTRUCT,BYTES=50,PARENT=CLASS
FIELD  NAME=(INSTNO,SEQ,U),BYTES=6,START=1
SEGM   NAME=STUDENT,BYTES=50,PARENT=CLASS
FIELD  NAME=(STUDNO,SEQ,U),BYTES=10,START=1
FIELD  NAME=STUDPHNO,BYTES=10,START=11
FIELD  NAME=STUDNAME,BYTES=20,START=21
DBDGEN
```

Figure 3-13  DBD for DEDB with target CLASS, source STUDENT

- Segment CLASS is the target segment.
- STUDENT is the source segment.

The LCHILD and XDFLD statements follow the SEGMENT statement for CLASS.
The LCHILD segment specifies the secondary index segment, STUDXSEG, and database, STUDSXDB. PTR=SYMB is specified, as required.

The XDFLD statement specifies the name of the search field, XSTUD, for use with the secondary index.

- Because the source segment is not the target segment, SEGMENT=STUDENT is specified to indicate that STUDENT is the source segment.
- SRCH=STUDPHNO specifies that field STUDPHNO is used to build the search field.
- A =/CK field is not used to make the STUDENT key unique. Therefore, we have to specify MULTISEG=YES in LCHILD.

Figure 3-14 shows the DBD for the secondary index database. It must be HISAM because of the duplicated secondary index keys, therefore, ACCESS=(INDEX, VSAM) is specified.

**Figure 3-14  DBD for index with target CLASS, source STUDENT**

- **DDI=** on the DATASET statement specifies the DD name of the secondary index data set. Because the key is not unique, the OVFLW= parameter must be specified on the DATASET statement.
- The SEGM statement defines the segment in the secondary index. Its size is 21 bytes. It is composed of the secondary index key (10 bytes) and the symbolic pointer to the target (11 bytes).
- The FIELD statement defines the sequence field in the secondary index. Because the keys are not unique, M is specified in the NAME= parameter.
- The LCHILD statement specifies the target segment CLASS and database EDUCATDB in the NAME= parameter. The INDEX= parameter specifies the NAME= value on the XDFLD statement of the target database. This is XSTUD. PTR=SYMB is required for Fast Path secondary index databases.

Secondary indexes for Fast Path databases are invisible to the application program. When a DEEDB database needs to be accessed using its Fast Path secondary index, the PROCSEQD=
parameter in the PCB is used to specify the name of the Fast Path secondary index database to use to access the primary DEDB database. The \texttt{PROCSEQD=} parameter has the same function as the full-function \texttt{PROCSEQ=} parameter. The \texttt{PROCSEQD=} parameter stands for \texttt{PROCSEQ} for DEDB databases.

When a primary DEDB database is accessed through its secondary index using the PCB with the \texttt{PROCSEQD=} parameter, the primary DEDB database is processed in an alternate sequence. The way in which the application program perceives the database record changes. If the target segment is a root segment in the primary DEDB database, the inverted structure in the primary DEDB database that is using the secondary index is the same as the physical structure of the primary DEDB database.

If the target segment is not a root segment in the primary DEDB database, the hierarchy in the database record is conceptually restructured as an inverted structure. The DEDB inverted structure access is limited to a subset of segments. For DEDB inverted structure access, the target segment, its direct parent segments from the target segment to the root segment, and all its child segments from the target segments are accessible.

Figure 3-15 shows the inverted sequence for target segment CLASS.

Nevertheless, the SENSEG statements in the PCB must be specified in the hierarchical sequence of the main DEDB (Example 3-16).

\textbf{Example 3-16}  PCB for processing by using the secondary index \texttt{STUDXSEG}

\begin{verbatim}
PCB TYPE=DB,DBDNAME=EDUCATDB,PROCOPT=G,KEYLEN=30,PROCSEQD=STUDXSEG
SENSEG NAME=COURSE,PARENT=0
SENSEG NAME=CLASS,PARENT=COURSE
SENSEG NAME=STUDENT,PARENT=CLASS
PSBGEN LANG=ASSEM,PSBNAME=CLASSTUD
END
\end{verbatim}

When you use a Fast Path secondary index, you must specify the \texttt{PROCSEQD=} parameter on the PCB statement in the PSB. Notice that you must specify \texttt{PROCSEQD}, and not \texttt{PROCSEQ}. The D is added to indicate that the index is for a DEDB.

- The order of the SENSEG statements is the physical order in the indexed database, not the secondary index processing order. In this example, the SENSEG statements appear in the order \texttt{COURSE}, \texttt{CLASS}, and \texttt{STUDENT}.
- The secondary index processing order is \texttt{CLASS}, \texttt{COURSE}, and \texttt{STUDENT}. 

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure3-15.png}
\caption{Processing sequence with target class}
\end{figure}
The KEYLEN value on the PCB statement must be the larger of:

- The largest concatenated key in the physical structure for any segment which might be accessed by the secondary index. In this case, it is the larger of the concatenated keys for segment STUDENT (5+6+10).
- The largest concatenated key in the secondary structure. The key of the target segment in the secondary structure is the secondary index key. In this case, the target is segment CLASS. The largest concatenated key is the secondary index key plus the largest of the keys for segments COURSE and STUDENT (6+10).

Secondary indexes for Fast Path databases have capabilities that are not available with secondary indexes for full function databases.

The first of these capabilities is user data partitioning (Figure 3-16) so that a secondary index can be spread across multiple physical databases. This supports large indexes.

Each index database contains a range of keys. Either HISAM or SHISAM can be used, but all “partitions” in an index must be the same type. Index keys are assigned to an index database by a user partition selection exit routine. The index that is passed is the secondary index key.

Each database in an index must have the same structure and attributes. The databases might have different sizes to accommodate the number of entries that can exist in the different key ranges.

![Figure 3-16  User partitioning for secondary indexes](image1.png)

To use this facility, you must use special keywords in the DBD definition of the DEDB (Example 3-17).

**Example 3-17  DBD additional keywords for user partitioning**

* Specify multiple index databases on DEDB LCHILD statement
  
  LCHILD NAME=(segname,(db1name,db2name,...)),PTR=SYMB

* Specify PSELRTN= on DEDB XDFLD statement

* Optionally specify PSELOPT= on DEDB XDFLD statement
  
  XDFLD NAME=searchname,SRCH=fldname,PSELRTN=rtnname,PSELOPT=MULT|SNGL,...

User data partitioning is defined by specifying multiple database names on the LCHILD statement in the indexed database. The order of the databases in the LCHILD statement determines the order in which the index partitions are processed by get next calls. This means that a get next call that reaches the end of an index database will continue to the next database defined in the LCHILD statement.
User data partitioning also requires the specification of PSELRTN= on the XDFLD statement in the indexed database. This parameter specifies the user partition selection exit routine.

The partition selection routine is called when an insert or qualified get call is issued. The routine determines which secondary index database is used for the call. That is, it is used to specify in which database a secondary index key is stored. The routine can be shared by multiple secondary indexes. These indexes might be on different databases.

A sample partition selection routine, DBFPSE00, is supplied by IMS.

You can specify the PSELOPT= parameter on the XDFLD statement.

The partition selection option controls whether only one secondary index user partition is used for a call. It is only used with secondary index processing. This means that PROCSEQD= must be specified in the PCB used for the call.

The value specified for PROCSEQD= must be the first partition.

- **PSELOPT=MULT** indicates that all partitions are used. A “GB” (end of database) status code is returned when the end of the last secondary index partition is reached.
- **PSELOPT=SNGL** indicates that only one partition is used. A “GB” status code is returned if the call reaches the end of the current partition. The PSELOPT= parameter can be specified in two places, the DBD and the PCB. The specification in the PCB overrides the specification in the DBD. If neither is specified, the PSELOPT=MULT parameter is used.

The second of these capabilities is support for multiple secondary index segments. This is one index created from different fields in the same source segments.

Because the index has one key size, the search fields must be the same size. With this capability, you can build one index from similar data that is stored in different fields. An example is an index based on telephone numbers. Multiple phone numbers, such as home phone, work phone, and cell phone can be stored in different fields, but only one index is used. The index might have an entry for each phone number.

![Figure 3-17 Multiple secondary index segments](image)

Multiple secondary index segments is the support for one index created from different fields in the same source segments. Each search field (or set of search fields) is used to create an entry in the secondary index. These fields or sets of fields must be the same size. In this example, telephone numbers in the OWNER segment have three fields. The index might contain an entry for each phone number.
Example 3-18  DBDGEN statements for multi secondary index segments

```
LCHILD NAME=(sisegname,sidbname),PTR=SYMB,MULTISEG=YES
XDFLD ..................NULLVAL=value1
```

Multiple secondary index segments are defined by including multiple LCHILD and XDFLD statement pairs in the indexed database (Figure 3-18).

The LCHILD segment must include the **MULTISEG=YES** parameter. **MULTISEG=NO** is the default for this parameter.

The **NULLVAL=** parameter is useful for avoiding the creation of secondary indices without a value. With this parameter, you suppress the creation of index pointer segments when the index source segment data used in the search field of an index pointer segment contains the specified value.

```
DBD    NAME=ACCTDB,ACCESS=DEDB,RMNAME=RMD4
AREA   DD1=ACCT1,SIZE=1024,UOW=(100,10),
       ROOT=(236,36)
SEGM   NAME=ACCT,PARENT=0,BYTES=100
FIELD  NAME=(ACCTNO,SEQ,U),BYTES=12,START=1
LCHILD NAME=(PHONEKEY,PHONINDX),PTR=SYMB,MULTISEG=YES
XDFLD  NAME=XPHON,SEGMENT=OWNER,SRCH=HOMEPHN
LCHILD NAME=(PHONEKEY,PHONINDX),PTR=SYMB,MULTISEG=YES
XDFLD  NAME=XPHON,SEGMENT=OWNER,SRCH=WORKPHN
LCHILD NAME=(PHONEKEY,PHONINDX),PTR=SYMB,MULTISEG=YES
XDFLD  NAME=XPHON,SEGMENT=OWNER,SRCH=CELLPHN
SEGM   NAME=OWNER,BYTES=300,PARENT=ACCT
FIELD  NAME=OWNNAME,BYTES=40,START=1
FIELD  NAME=HOMEPHN,BYTES=10,START=41
FIELD  NAME=WORKPHN,BYTES=10,START=51
FIELD  NAME=CELLPHN,BYTES=10,START=61
DBDGEN
```

**Figure 3-18** DBD for DEDB with multiple secondary indexes

Multiple LCHILD and XDFLD statements refer to the same secondary index. The LCHILD statements are the same. The XDFLD statements have the same **NAME=** and **SEGMENT=** parameters. The **SRCH=** parameters point to the various fields that are used for indexing.

Example 3-19  DBD for index database with multiple secondary indices

```
DBDSX  DBD    NAME=PHONINDEX,ACCESS=(INDEX,VSAM)
       DATASET DD1=PHONKSDS,OVFLW=PHONOVFL
       SEGM   NAME=PHONESG,PARENT=0,BYTES=22
       FIELD  NAME=(PHONEKEY,SEQ,U),BYTES=10,START=1
       LCHILD NAME=(OWNER,ACCTDB),INDEX=XPHON,PTR=SYMB
       DBDGEN
```

In this example, the XDFLD statements in the DEDB DBD might include the **NULLVAL=** parameters to specify that a secondary index entry is not built when the phone field contains blanks.
Creating DEDBs with secondary indexes

The following two scenarios are considered:

- Adding a secondary index to an existing DEDB
- Creating a new DEDB with a secondary index

**Adding a secondary index to an existing DEDB**

To add a secondary index on an existing DEDB, complete these steps:

1. Modify the DEDB DBD.
2. Add the LCHILD and XDFLD statements.
3. Create a secondary index DBD.
4. Allocate secondary index data sets.
5. Create a PSB with the `PROCSEQD=` parameter in PCB.
6. Run the `DBDGEN` and `PSBGEN` utilities, and then run the `ACBGEN` utility.
7. Take DEDB offline.
8. Run tool or user program to create the secondary index.
9. Add a secondary index to the system definition.
10. Run the `CREATE DB` command or system definition and online change.
11. Add any new programs and transactions that use the secondary index.
12. Run the `CREATE` commands or system definition and online change.
13. Make an online change for application control blocks library (ACBLIB).
14. Start DEDB.

You need to write a program to load the secondary index databases or use a tool. This program or tool must fill the segments of the secondary indices. Several fields must be filled depending on the XDFLD description of the contents of the SI.

Figure 3-19 shows an overview of the different segment layouts that are possible, depending on the index organization (HISAM, SHISAM), or whether we have duplicate (only with HISAM) SI values.

### HISAM secondary index pointer with a unique key

<table>
<thead>
<tr>
<th>Delete byte</th>
<th>Search</th>
<th>Subsequence (optional)</th>
<th>Duplicate data (optional)</th>
<th>Symbolic pointer concatenated key</th>
<th>User data (optional)</th>
</tr>
</thead>
</table>

### HISAM secondary index pointer with a non-unique key

<table>
<thead>
<tr>
<th>Duplicate key pointer</th>
<th>Delete byte</th>
<th>Search</th>
<th>Subsequence (optional)</th>
<th>Duplicate data (optional)</th>
<th>Symbolic pointer concatenated key</th>
<th>User data (optional)</th>
</tr>
</thead>
</table>

### SHISAM secondary index pointer

<table>
<thead>
<tr>
<th>Search</th>
<th>Subsequence (optional)</th>
<th>Duplicate data (optional)</th>
<th>Symbolic pointer concatenated key</th>
<th>User data (optional)</th>
</tr>
</thead>
</table>

*Figure 3-19  Secondary index segment layouts*
If the secondary index key is not unique then an additional pointer field of length 4 is added in front of the segment. This is the duplicate key pointer, pointing to extensions in the ESDS data set. In Figure 3-19 on page 94, we recognize the following fields:

- **Prefix**
  - Delete byte: one byte

- **Data fields:**
  - **Search**
    - This is the key of the secondary index. It contains variable-length bytes, made up of up to 5 fields from the source.
  - **Subsequence**
    - This is used to make the secondary index key unique. It contains variable-length bytes, made up of up to 5 fields from the source or IMS-generated values (optional). It is used to make the secondary index key unique. It can be used to order segments in a secondary index database. The search field and the subsequence field together make up the key of the secondary index.
  - **Duplicate data**
    - This is only used when processing the secondary index as a database. It contains variable-length bytes, made up of up to 5 fields from the source (optional). It is only used when processing the secondary index as a database.
  - **Symbolic pointer concatenated key**
    - This contains variable-length bytes. It is the concatenated key to the target.
  - **User data**
    - This contains variable-length bytes, made up of any user data fields (optional). It is only used when processing the secondary index as a database.
  - **Duplicate key pointer**
    - This is the pointer to chain duplicated segments. It points into the overflow data set of the index, which is a VSAM ESDS.

The IMS High Performance Fast Path Utilities offering is used for validating and building secondary indexes on Fast Path databases. Although this tool is described in Chapter 10, “Tools for IMS 12” on page 385, a brief overview is provided here.

FPA Secondary Index (FPSI) support provides build and analyze capabilities for the Fast Path secondary index databases that are supported in IMS 12 with the following features:

- HISAM/SHISAM secondary index databases
- Unique Key/Non-Unique Key
- Sparse Indexing
- Symbolic Pointer
- Multiple Secondary Index Segments
- Partition Selection

For more information, see *IBM Fast Path Solution Pack for z/OS V1R1, IMS High Performance Fast Path Utilities User's Guide*, SC19-2914.

The HPFPU Build Index function (INDEXBLD) is related to Fast Path secondary indexes. This function (Figure 3-20 on page 96) builds all FPSI databases of DEDB areas when the secondary indexes are defined against the existing DEDB. It can build multiple secondary index databases in one job step, and can rebuild the specific secondary index databases in case of system failures. APAR PM37894 has enabled the new INDEXBLD function for users of the IMS Database Recovery Facility under IMS 12.
This function reduces the amount of time that it takes to build multiple secondary index databases using both parallel sorting and parallel loading. It also provides the following new reports:

- Secondary Index Definition Report
- Secondary Index Processing Report

FPSI data set names are identified with the member in the DFSMDA library. You can also specify in the JCL DD statements. The NOTIFY.REORG command is issued for FPSI databases when the DBRC=YES parameter is specified (Example 3-20).

**Example 3-20 JCL and control information for INDEXBLD function**

```plaintext
//HFP EXEC PGM=HFPMAIN0
//STEP LIB DD DISP=SHR,DSN=HPFP.SHFPPLMD0
//   DD DISP=SHR,DSN=IMSVS.SDFSRESL
//   DD DISP=SHR,DSN=IMSVS.PGMLIB
//IMSACB DD DISP=SHR,DSN=IMSVS.ACBLIB
//IMSDALIB DD DISP=SHR,DSN=IMSVS.MDALIB
//IMS DD DISP=SHR,DSN=IMSVS.DBDLIB
//Hfpsysin DD *
GLOBAL DBRC=YES
INDEXBLD DBD=DEDBJN22,ITASKCTL=4,AREA=ALL,INDEXDBD=ALL
/*
```

**Creating a new DEDB with a secondary Index**

In this case, a tool or user-written program is not required to build the secondary index. Instead, the insertion of source segments in the DEDB causes the secondary index entries to be created.

To create a new DEDB with a secondary index and to create entries in the secondary index, complete these steps:

1. Create a DEDB DBD with the LCHILD and XDFLD statements.
2. Create a secondary index DBD.
3. Allocate the DEDB and secondary index data sets.
4. Create a PSB with the PROCSEQD= parameter in PCB.
5. Run the DBDGEN and PSBGEN utilities, and then run the ACBGEN utility.
6. Add the DEDB and secondary index to the system definition.
7. Run the **CREATE DB** commands or system definition and online change.
8. Add any new programs and transactions to the system definition.
9. Run the **CREATE** commands or system definition and online change.
10. Make an online change for ACBLIB.
11. Run the DEDB Initialization utility.
12. Load the DEDB.

An alternative method for adding a secondary index to an existing DEDB without requiring tool is to unload the DEDB database, and then use the same steps as documented here.

### 3.3.5 DEDB buffer pool enhancements

The DEDB buffer pool enhancements capability applies to DEDBs for both DBCTL and IMS Transaction Manager (TM) or database environment support. It was introduced in IMS 11 and has been enhanced in IMS 12.

**IMS 11**

In any IMS with the database manager, buffers are needed to fulfill requests from database calls are obtained from a global pool called the **Fast Path buffer pool**.

If you are using the Fast Path 64-bit buffer manager, IMS creates and manages the Fast Path buffer pools for you and places DEDB buffers in 64-bit storage. The buffers are placed in virtual storage above 2G, above the bar. The 64-bit buffer manager creates multiple subpools with different buffer sizes. The 64-bit buffer manager creates an initial allocation of subpools based on the number of areas of each CI size. It automatically created more buffers in a subpool when they were needed. This is only possible when you have the Z/architecture enabled. When the Fast Path 64-bit buffer manager is enabled, you do not need to design DEDB buffer pools or specify the **DBBF**, **DBFX**, and **BSIZ** parameters that define Fast Path buffer pools.

In IMS 11, you can enable the Fast Path 64-bit buffer manager by specifying **FPBP64=Y** in the Fast Path section of DFSDFxxx PROCLIB member (Example 3-21).

```plaintext
Example 3-21   IMS 11 Fast Path specifications in DFSDFXXX

<SECTION=FASTPATH>
FPBP64=Y,FPBP64M=xxxxxx (set maximum storage used)
</SECTION>
```

When the Fast Path 64-bit buffer manager is enabled, IMS ignores the **DBBF**, **DBFX**, and **BSIZ** parameters, if specified.

**IMS 12**

IMS 12 enhances the Fast Path 64-bit buffer manager, which was introduced in IMS 11. With IMS 12, you can specify the initial amount of 64-bit storage used for the buffer pool. Buffer pools are pre-expanded, that is, expanded in anticipation of future needs. They are compressed when the use of a subpool drops. IMS 12 moves some buffers that were still in extended common service area (ECSA) to 64-bit storage. Finally, IMS 12 enhances the **QUERY POOL TYPE(FPBP64)** command output.
The keywords FPBP64D, FPBP64C, FPBP64E, and FPBP64SR are introduced in IMS 12 and are described here:

- **FPBP64D=N** is the default. This default leaves the initial allocation as it was in IMS 11.
- **FPBP64D=Y** is specified in the FASTPATH section of the DFSDFxxx PROCLIB member, the **DBBF** execution parameter is used to determine the initial number of buffers. The initial number of buffers is 25% of the DBBF specification. These buffers are allocated among different buffer sizes based on the number of areas with different CI sizes.

For example, if a DBBF specification is 8000, then 2000 (25% of 8000) buffers will be used for the initial allocation:

- 100 areas with a 1 K CI size
- 200 areas with a 2 K CI size
- 700 areas with a 4 K CI size.

The sum of 100+200+700 is a total of 1000 areas.

10% of the areas have a 1 K CI size, 200 (10% of 2000) 1 K buffers are initially allocated.
20% areas have 2 K CI size, 400 (20% of 2000) 2 K buffers are initially allocated.
70% areas have 4 K CI size, 1400 (70% of 2000) 4 K buffers are initially allocated.

- **FPBP64D=Y** is specified but there is no DBBF specification, the initial allocation is the same as it was in IMS 11.
- **FPBP64E**, you can disable the pre-extension code in a way that is similar to how you can disable the compression code with FPBP64C.

IMS 11 expands a 64-bit subpool when a DL/I call requires a buffer but none is available. IMS 12 has an option to expand subpools in anticipation of the need for more buffers. When a subpool is almost out of available buffers, the extension process is initiated asynchronously. As the volume of buffer requests increase, the subpool extension process will increase the pace at which subpools are extended. By the time the additional buffers are required, the subpool should have been extended, avoiding wait-for-buffer conditions.

IMS 12 also adds the capability to compress subpools when there are substantial unused buffers. IMS resizes the subpool after 24 hours if it is grossly overallocated in size. Subpools are compressed by reducing the number of buffers in the subpool, which is the default.

Subpools can also be deleted. That is, all of the buffers of a certain size can be deleted. This will only be done when none of the buffers of this size have been used for more than 24 hours. If buffers of this size are needed again, the subpool will be rebuilt. The compression and deletion actions are the default. They can be disabled by specifying **FPBP64C=N** in the FASTPATH section of the DFSDFxxx PROCLIB member.

The use of the 64-bit Fast Path buffer manager in IMS 11 did not put all buffers in 64-bit storage. Buffers in ECSA were used for MSDBs, SDEP inserts and FLD calls by IMS 11. Additionally, ECSA storage was used for buffer headers and control blocks.

- **IMS 11 emergency restart uses ECSA buffers for all SDEP processing.**
- **IMS 12 uses 64-bit buffers for FLD calls.**
- **IMS 12 emergency restart uses 64-bit buffers for SDEPs unless FPBP64SR=N is specified.**

IMS 12 increases the size of the extents if too many extents exist, which keeps the number of extents to a minimum.

IMS 12 enhances the **QUERY POOL TYPE(FPBP64)** command from IMS 11. The new **SHOW(STATISTICS)** option provides a subset of the data returned with the **SHOW(ALL)** option,
which was the only SHOW option available in IMS 11. The SHOW(ALL) option provided so much information that, at times, it was difficult to find the most interesting data. The QUERY POOL TYPE(FPB64) SHOW(ALL) was also reformatted for easier reading.

The QUERY command in IMS 12 also provides data on extended private (EPVT) use. It provides new status information for subpools. The status shows if a pool is being compressed, expanded, or deleted.

If you are not using the Fast Path 64-bit buffer manager, you must specify the characteristics of the pool yourself during IMS system definition and during IMS startup.

### 3.4 CICS threadsafe support

Release of CICS Transaction Server for z/OS Version 4.2 includes scalability enhancements that allow more work to be done faster in a single CICS system. These enhancements allow increased vertical scaling and can decrease the need to scale horizontally, thus reducing the number of regions that are required to run the production business applications.

The scalability enhancements in CICS Transaction Server Version 4.2 fall into two broad areas: increased exploitation of Open Transaction Environment (OTE) and increased exploitation of 64-bit storage. For more information, see *IBM CICS Scalability: New Features in V4.2*, REDP-4787.

CICS OTE is an architecture that was introduced for the following purposes:

- To allow parallelism in using the mainframe resources, increasing the throughput of work through the system
- To improve the performance of existing applications accessing external resource managers, such as IMS and DB2
- To enrich the CICS application programming interface by providing application interfaces supplied by other software components and allowing CICS applications to use these interfaces

To benefit from OTE, your applications need to be threadsafe. OTE enhancements make more of the CICS API and system programming interface (SPI) threadsafe, including access to IMS databases through the CICS-DBCTL interface. This CICS threadsafe function was added with CICS V4.2 for IMS V12 with APARs PM31420 and PM47327.

With these APARs, IMS 12 provides the coordinator controller (CCTL) database resource adapter (DRA) Open Thread TCB (OTT) enhancement. IMS 12 now provides the option for CCTL exploiters to direct the DRA not to attach dedicated DRA thread control blocks (TCBs). This option avoids the overhead of TCB switching for IMS database calls and lead to improved parallel processing (Figure 3-21 on page 100).

CICS TS 4.2 extends the threadsafe support to the DBCTL/DRA interface taking advantage of the DRA OTT support. The enhancement applies to both EXEC DLI and CALL DLI.

Without this enhancement, the IMS CCTL DRA attaches dedicated thread TCBs in the CCTL address space. PSB schedule requests are assigned one of these thread TCBs. All subsequent thread-related DL/I requests result in the application's task being suspended, and processing switches onto the DRA thread TCB to complete the DL/I request. Upon returning, the application's task is resumed, switching processing off the DRA thread TCB which is then suspended awaiting the next DL/I request. This sequence of events is repeated for each DL/I
call, sync point requests, and thread termination, at which time the DRA thread TCB becomes available for a new PSB schedule request.

The expected resulting benefits are lower CPU usage and increased throughput.

![Figure 3-21  TCB switching with threadsafe support for CICS accessing IMS database](image)

### 3.5 Miscellaneous enhancements

IMS 12 contains other miscellaneous enhancements. Various changes improve the logging and the locking process, or avoid some abends, or help you to better understand problems by issuing new or enriched messages.

#### 3.5.1 Status messages for DB exit routines

If an HDAM or PHDAM database is opened as the result of an open command of the database, message DFS2842I is issued. See Example 3-22. Either LOADED or SHARED is displayed in the message. The DFS2842I message is issued for full function databases as a result of the following commands:

```
/START DB dbname OPEN
UPDATE DB NAME(dbname) START(ACCESS) OPTION(OPEN)
```

**Example 3-22  DFS2842I message**

```
DFS2842I RANDOMIZER name FOR database IS LOADED|SHARED
```

Where:

- LOADED appears when the routine is loaded as a result of the open of the database.
- SHARED appears when the routine is already in memory due to its use by another database.
When the database is closed as the result of a command, either GONE or SHARED is displayed in message DFS2838I (Example 3-23).

Example 3-23  DFS2838I message

DFS2838I RANDOMIZER name FOR database IS DELETED AND GONE|SHARED

The DFS2838I message is issued for full function databases as a result of the following commands:

- /DBR DB dbname
- /DBD DB dbname
- /STO DB dbname
- /STA DB dbname
- UPDATE DB NAME(dbname) STOP(ACCESS\updates\SCHD)
- UPDATE DB NAME(dbname) START(ACCESS)

In message DFS2838I:

- GONE appears when the routine is deleted from memory.
- SHARED appears when the routine remains in memory due to its use by another database.

If an HALDB database uses a partition selection exit routine, the DFS2406I message is issued when the database is opened or closed as the result of a command. When the database is opened, either LOADED or SHARED is displayed in the message. The following commands are among those that might cause the DFS2406I message to be issued:

- /START DB HALDBmaster OPEN
- UPDATE DB NAME(HALDBmaster) START(ACCESS) OPTION(OPEN)
- UPDATE DB NAME(HALDBmaster) STOP(ACCESS\UPDATES\SCHD)
- /DBR DB HALDBmaster
- /DBD DB HALDBmaster

These messages are especially useful when replacing a shared exit routine. They clearly indicate if the old routine has been deleted and if a new routine has been loaded (Example 3-24).

Example 3-24  DFS2406I message

DFS2406I THE HALDB PARTITION SELECTION EXIT ROUTINE rname FOR THE HALDB dbname IS LOADED|GONE|SHARED

When the database is opened, either LOADED or SHARED appears in the message.

- LOADED appears when the routine is loaded as a result of the opening of the database.
- SHARED appears when the routine is already in memory due to its use by another database.

When the database is closed, either GONE or SHARED appears in the message.

- GONE appears when the routine is deleted from memory.
- SHARED appears when the routine remains in memory due to its use by another database.
3.5.2 Lock timeout message and logging

IMS 12 adds optional DFS2291I diagnostic messages for lock timeouts (Example 3-25) and writes log record x’67D0’ subtype x’1B’ for lock timeouts. Both contain the same information.

If the wait for a lock exceeds the IMS LOCKTIME value when using the internal resource lock manager (IRLM), then the lock request is rejected and either the waiter is abended with a U3310 or a BD status code is returned to the program.

Example 3-25   Multiple DFS2291I messages

<table>
<thead>
<tr>
<th>DFS2291I LOCKNAME=0900001004801001D7</th>
<th>I12A</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFS2291I DBNAME=IVPDB1 LOCKFUNC=GET LCL AND GBL ROOT LOCKS</td>
<td>I12A</td>
</tr>
<tr>
<td>DFS2291I BLOCKER PST=0001 TRAN=IMSLBLK56 PSB=DFSIVP65 TYPE=BMP I12A</td>
<td></td>
</tr>
<tr>
<td>DFS2291I BLOCKER TRANELAPSEDTIME=00:00:35 IMSID=I12B</td>
<td>I12A</td>
</tr>
<tr>
<td>DFS2291I BLOCKER RECOVERY TOKEN=I12B 0000002D00000000 I12A</td>
<td></td>
</tr>
<tr>
<td>DFS2291I WAITER01 PST=0002 TRAN=IMSLDLK65 PSB=DFSIVP65 TYPE=BMP I12A</td>
<td></td>
</tr>
<tr>
<td>DFS2291I WAITER01 TRANELAPSEDTIME=00:00:35 IMSID=I12D</td>
<td>I12A</td>
</tr>
<tr>
<td>DFS2291I WAITER01 RECOVERY TOKEN=I12D 0000002700000000 I12A</td>
<td></td>
</tr>
<tr>
<td>DFS2291I WAITER02 PST=0001 TRAN=IMSLBLK64 PSB=DFSIVP64 TYPE=BMP I12A</td>
<td></td>
</tr>
<tr>
<td>DFS2291I WAITER02 TRANELAPSEDTIME=00:00:35 IMSID=I12B</td>
<td>I12A</td>
</tr>
<tr>
<td>DFS2291I WAITER02 RECOVERY TOKEN=I12B 0000002C00000000 I12A</td>
<td></td>
</tr>
<tr>
<td>DFS2291I WAITER03 PST=0006 TRAN=IMSLDLK65 PSB=DFSIVP65 TYPE=BMP I12A</td>
<td></td>
</tr>
<tr>
<td>DFS2291I WAITER03 TRANELAPSEDTIME=00:00:35 IMSID=I12B</td>
<td>I12A</td>
</tr>
<tr>
<td>DFS2291I WAITER03 RECOVERY TOKEN=I12B 0000002800000000 I12A</td>
<td></td>
</tr>
<tr>
<td>DFS2291I VICTIM PST=0005 TRAN=IMSLALK64 PSB=DFSIVP64 TYPE=BMP I12A</td>
<td></td>
</tr>
<tr>
<td>DFS2291I VICTIM TRANELAPSEDTIME=00:00:17 IMSID=I12A</td>
<td>I12A</td>
</tr>
<tr>
<td>DFS2291I VICTIM RECOVERY TOKEN=I12A 0000000F00000000 I12A</td>
<td></td>
</tr>
</tbody>
</table>

The first example is for the multiple line message. If the same lock has other waiters, they are also listed with the word WAITER where VICTIM appears in this example. The second example is for the single line or “short” message.

In this example, BMP IMSBLK65 using PSB DFSIVP65 on IMS I12B holds a local and global root lock in database IVPDB1. The elapsed time of this update is now 35 seconds. Another BMP IMSDLK65 using the same PSB DFSIVP65 on IMS I12D is waiting on this lock. Its elapsed time is now 35 seconds. The victim on IMS I12A is a BMP IMSALK64 using PSB DFSIVP64 on IMS system I12A.

Example 3-26 shows a similar problem except this time IMS is configured to only issue the short form of the DFS2291I message.

Example 3-26   Short form of the DFS2291 message

| DFS2291I BATCH REGION IMSCLK64 STARTED ID=00004 TIME=0942 | I12C |
| DFS2291I BATCH REGION IMSCLK65 STARTED ID=00005 TIME=0942 | I12C |
Previous IMS releases provide information only through IBM RMF™ reports. The RMF II ILOCK (IRLM Long Lock Detection) Report includes information about waiters and blockers when a lock request exceeds the IRLM TIMEOUT value. RMF records (type 79.15) can be formatted for more information about the task that is holding (or waiting for) the lock.

By using the IRLM Lock Timeout function, you can interrupt processes that are waiting for locks for longer than a specified number of seconds, with the integer value for the LOCKTIME parameter, specified in the DFSVSMxx member of the IMS PROCLIB data set (or DFSVSAMP DD statement for IMS batch procedures). See Example 3-27.

Example 3-27   LOCKTIME definition in DFSVSMxx

```
LOCKTIME=(xxxxx,ABEND/STATUS,[xxxxx,ABEND/STATUS])
```

The number of seconds can also be changed after IMS initialization by issuing the following command:

```
MODIFY irlmproc,SET,TIMEOUT=nn,imsid
```

The DFS2291I messages are only issued if they are requested by specifying MSG2291=ISSUE or MSG2291=SHORT in the DIAGNOSTIC_STATISTICS section of the DFSDFxxx member (Example 3-28).

Example 3-28   DFSDFxxx DIAGNOSTIC_STATISTICS section

```
<SECTION=DIAGNOSTICS_STATISTICS>
  MSG2291=ISSUE | SHORT | SUPPRESS
</SECTION>
```

Consider the following explanation:

- ISSUE causes multiple line messages to be issued.
- SHORT causes one line messages to be issued.
- SUPPRESS is the default.

If the user has chosen the ABEND option for timeouts, IMS TM transactions are retried with three exceptions (IFP regions, CPI-C driven applications, and protected conversations).

3.5.3 Batch data sharing abend elimination

In previous versions of IMS, a batch data sharing job might abend with U3303 when an OSAM or VSAM cache structure access failed. For example, an access attempt while a structure was being rebuilt failed. This problem did not occur with online systems. They survived access failures. They waited for the resolution to the structure access problem.

With IMS 12, batch jobs can survive when these structure accesses fail. Like online systems, they wait for the resolution to the problem. When the problem is resolved, the batch jobs continue processing. For example, when a rebuild of a structure completes, the batch jobs continue.
If the batch job detects the failure, it issues the new DFS2404A message:

DF52404A AN ERROR WAS ENCOUNTERED WHEN ACCESSING THE COUPLING FACILITY. STRUCTURE xxxxxxxxxx RSN yyy

The reason code in the message is used to identify the type of failure that occurred when the batch job attempted to access the structure.

With this enhancement, you can rebuild OSAM and VSAM cache structures while your data sharing batch jobs are executing. You might run this task to address coupling facility failures or to move structures between coupling facilities for reconfigurations. In previous versions of IMS, batch jobs did not survive these rebuilds.

### 3.5.4 RACF user ID in Data Capture batch log records

Data Capture exit routines are compatible with the following physical database structures:

- HDAM
- PHDAM
- HIDAM
- PHIDAM
- HISAM
- SHISAM
- DEDB

LOG is a parameter of the DBD macro in the DBDGEN utility (Example 3-29).

**Example 3-29  LOG specification in DBD of the DBDGEN utility**

```
DBD..., EXIT=([exitname],LOG/NOLOG)
```

Exitname is optional, and LOG can be specified without exit.

In previous versions of IMS, the IBM Resource Access Control Facility (IBM RACF®) user ID only appears in changed data capture log records (type x’9904’) when the log is produced by an online system. IMS 12 adds the RACF user ID to these log records when they are produced by batch (DLI or DBB) jobs. Changed data capture writes log records when the LOG (asynchronous) option is chosen in the DBDGEN.

The RACF user ID is specified by including the USER= parameter on the JOB statement of the batch job.

### 3.5.5 Increased VSAM pools

Previous versions of IMS allowed only 16 VSAM full function database buffer pools to be defined for an IMS online system, batch job, or utility. IMS 12 expands this number to 255. Each buffer pool can have separate subpools for different buffer sizes and for data and index components.

VSAM buffer pools are defined with POOLID statements in the DFSVSMxx member or DFSVSAMP data set. With IMS 12, you can specify up to 255 of these POOLID statements.
3.5.6 Elimination of OSAM U0080 Open, Close, or EOV abends

Previous versions of IMS had rare problems in OSAM open, close, or end-of-volume processing. When they occurred, the entire IMS system terminated with the U0080 abend.

IMS 12 has changed this processing. When such problems occur, the database is closed and marked recovery needed. The abend does not occur. Additionally, message DFS0730I is issued for open or close problems. X is included as the first character in the reason code. The yy and z values identify the actual problem (Example 3-30).

Example 3-30  DFS0730I message

| DFS0730I UNABLE TO OPEN|CLOSE DATASET WITH DDNAME ddname FOR REASON X, yy, z DATABASE dbname programid |

In previous versions, the DFS0730I message was not issued with the U0080 abend, which made it more difficult to determine the database and data set with the problem.

When the data set cannot be extended as part of EOV processing, message DFS0842I is issued:

| DFS0842I OSAM DATASET CANNOT BE EXTENDED, REASON=x, z dbname |

This message is enhanced in IMS 12 to include a subcode (z) to further explain the reason for extension failure.

3.5.7 Message DFS993I sent to system console

IMS 12 sends the DFS993I message to both the system console and the Master Terminal. Previous versions of IMS sent this message only to the Master Terminal. Because DBCTL systems do not have a Master Terminal, they did not receive the message:

| DFS993I (CSA PSB|DLS PSB|PSBW) POOL TOO SMALL, UNABLE TO SCHEDULE PSB PSBNAME |

The DFS993I message is issued when the PSB Work, CSA PSB, or DLI PSB pool is too small. With IMS 12, DBCTL users can easily determine why a PSB schedule failure occurs because of insufficient space in one of these pools.

3.5.8 Control Area reclaim support

When IMS database KSDS records are erased with z/OS 1.11 and previous releases, VSAM CI reclaim does not reclaim the empty CI that has the highest key of the Control Area (CA). This otherwise empty CA occupies the index structure as though it was not empty. If an application re-inserts records with the erased keys or keys of nearby values, those empty CAs are reused. However, if the application erases a range of keys and does not reuse those keys or only inserts records with ever-higher keys, VSAM does not reclaim or reuse those empty CAs with lower keys. The failure to reclaim the CAs results in wasted disk space and can cause performance problems in index search because much of the index structure can be populated with those empty index records (Figure 3-22 on page 106).
By using the CA Reclaim feature in z/OS 1.12, you can reuse the free CA space. With CA Reclaim, space fragmentation caused by erasing records from a KSDS will be minimized to reduce the need to reorganize the data set. When the freed CAs are placed in a free chain to be reused, the index structure can be shrunk to facilitate faster data accesses. When space is needed for a new CA, a CA from the free chain is reused so there will be fewer calls to EOV to extend the KSDS.

There is no requirement for all of the systems in a sysplex to be at the same z/OS release level. z/OS 1.10 and z/OS 1.11 have compatibility maintenance so that they can process data sets for which CA reclaim is being used with z/OS 1.12. However, CA reclaim is only processed on systems that have z/OS 1.12.

### 3.5.9 New command codes for sequential search

A small programming enhancement (SPE) for IMS 11 adds two new command codes for use with GN and GHN calls for root segments. These new command codes are also included in IMS 12.

- **Command code A** clears positioning and causes the GN or GHN call to start at the beginning of the database.
- **Command code G** is used with GN and GHN calls qualified on the root segment key. It prevents the use of the randomizer routine for DEDB, HDAM, and PHDAM databases. It also prevents the use of the partition selection exit routine for PHDAM or PHIDAM databases. Instead of using these routines, the call processes the next records in the database until one satisfying the SSAs is found or the end of the database is reached.

These codes change the actions that might otherwise occur for calls qualified on the key of the root segment. Consider a call qualified with key >= 1000 and key <= 2000. Without the A or G command codes, the call randomizes using key=1000. It returns the first segment found if it satisfies the SSAs. Otherwise, it returns a GE (not found) status code. If a segment is returned, successive calls using the same SSAs move forward in the database. If a segment not satisfying the call is encountered, a GE status code is returned. However, the use of the A command code causes the call to begin at the beginning of the database. The G command code causes the call to ignore roots which do not satisfy the SSAs and continue to following roots until one is found that satisfies the SSAs.

If you use these new command codes, use the A command code on the first GN or GHN call and do not use it on successive calls, because if A is used on every call, the calls always start at the beginning of the database and return the same segment.
These command codes should not be used with DL/I calls. They are used by JDBC with Open Database to provide results comparable to relational.

The JDBC driver uses the new command codes when converting a JDBC call to a DL/I call. This is done so that the JDBC call returns the same results with an IMS database that it returns with a relational database. Without these command codes, a search on a range of root segment keys is done with logic that assumes the roots are in key sequence. A call qualified on a range of root segment keys attempts to begin the search with the key at the bottom of the range.

For example, a search for roots with keys >= 1000 and <= 2000 begins by using the partition selection exit routine or the randomizer with key 1000. Subsequent GN or GHN calls with the same qualification move forward in the database. If a root segment with key>2000 is found, the search ends. The JDBC driver uses the new command codes to change the logic of the search. With the new command codes all root segments in the database are examined. This provides the expected results from the JDBC call.

Searching the entire database comes with a performance cost. The performance cost for the search can be eliminated by using an alternative. If many of these calls are issued by a program, you can create a secondary index on this key. Then you can use the secondary index for the calls by specifying PROCSEQ=, referencing the secondary index for full function databases or PROCSEQD=, referencing the secondary index for DEDBs. This way, the search can be performed without examining root segments that are not in the key range. It avoids the sequential scan of the database. This approach is analogous to placing an index on a column in a relational database. In fact, with a relational database, create the index when this type of JDBC call is made.

### 3.5.10 IRLM 2.3

Both IRLM 2.2 and IRLM 2.3 are delivered with IMS 12. Both of these IRLMs can be used with any supported version of IMS.

IRLM 2.3 is required by DB2 10 for z/OS. However, IRLM 2.2 can be used by the IMS database manager when DB2 is using IRLM 2.3. IRLM 2.3 supplies a 64-bit caller interface that is required by DB2 Version 10. IMS does not use this interface.

IRLM 2.3 must run under z/OS 1.10 or later. IRLM 2.3 provides some improved performance.

### 3.5.11 ACB library enhancements

The ACB library enhancements are also in IMS 11. With the ACBLIB usability enhancements, you can cache the ACB members into 64-bit storage, separately. You have to create DFSMDA members for the dynamic allocation of the ACBLIB data sets.

An ACB is built for each DBD and each PSB. Each ACB is stored as an ACB library member in the IMS.ACBLIB data set. ACBs are loaded into storage for online use either when the IMS control region is initialized or when an application program that requires the ACB is scheduled, depending on whether the ACB is resident or non-resident.

In online storage, the ACBs for PSBs and the ACBs for DBDs are stored separately in a PSB pool and DMB pool, respectively. IMS loads the non-resident ACB into 31-bit storage only when an application program that requires it is scheduled. Non-resident ACB members persist in storage only until the storage allocated for the non-resident ACBs is exhausted, at which time IMS frees storage by removing the ACBs that have remained unused for the longest period of time.
Non-resident ACB members can also be cached in a 64-bit storage pool to potentially improve the performance of program scheduling and reduce the usage of 31-bit storage. When 64-bit caching is enabled, when a non-resident ACB is first loaded into the 31-bit storage pool, it is also cached in the 64-bit storage pool. Later, if any non-resident ACBs that have been removed from online storage are required by a scheduled application program, the ACBs are retrieved from 64-bit storage instead of from the IMS.ACBLIB data set on DASD.

To enable your ACBs to use 64-bit storage, specify a \texttt{ACBIN64=ggg} parameter in the \texttt{<DATABASE>} section of the DFSDFxxx PROCLIB member. The value specified (ggg) is the amount of 64-bit storage (in GB) to be allocated for PSB and DMB ACB members. Because PSBs and DMBs are stored together in 64-bit storage pools, the size of a 64-bit storage pool must be large enough to contain all of the non-resident PSBs and DMBs combined. The size of an ACB member is reported by the ACB Maintenance utility when the ACB members are built.

To migrate ACB libraries to use dynamic allocation, you create DFSMDA members for the ACBLIBA and ACBLIBB data sets. Remove the IMSACBA and IMSACBB DD statements from the IMS and DL/I JCL procedures. IMS must be stopped and restarted with the new \texttt{ACBIN64} parameter in DFSDFxxx and with the DFSMDA members.

You can display information about the ACBs cached in 64-bit storage by using the type-2 \texttt{QUERY POOL TYPE(ACBIN64)} command.

For batch application programs, IMS does not support 64-bit caching of ACBs.

### 3.5.12 Reuse of local DMB numbers

The local database number (DMB number) is an internal value assigned by IMS when a database is defined to an IMS online system. The DMB is an IMS control block in main storage that describes and controls a physical database. A DMB is constructed from information that is obtained from the ACBLIB.

Before IMS 12, this local database number is never reused when its database is deleted by online change or DRD. A cold start is required when the local database number reaches the limit of 32K-1. In IMS 12, the local database numbers deleted by online change or DRD can be reused when databases are added by online change or DRD. This change affects only local DMB numbers.

\textbf{DBRC RECON data set:} Concurrent access to databases by systems in one or more z/OS operating systems is controlled with the shared Database Recovery Control (DBRC) RECON data set.

IMS systems automatically sign on to DBRC, ensuring that DBRC knows which IMS systems and utilities are currently participating in shared access. A given database must have a DMB number that uniquely identifies it to all the sharing IMS systems. Global DMB numbers are reused in previous versions of IMS. A global DMB numbers is assigned to a database when it is registered with DBRC. The reuse of global DMB numbers was introduced in IMS 9. The DMB number that DBRC records in its RECON data set is related to the order in which databases are registered to DBRC.
Chapter 4. Transaction Manager enhancements

This chapter describes the benefits of the Transaction Manager (TM) enhancements of IBM Information Management System (IMS) 12, including:

- APPC and OTMA shared queues enhancement
- OTMA ACEE reduction for multiple OTMA clients
- OTMA DFS2082 for Commit Mode 0
- Other Transaction Manager enhancements such as the following examples:
  - OTMA performance improvements
  - DFSMSCE0 exit modified for shared queue
  - LU 6.2 Input/Output Edit Exit (DFSLUEE0) Enhancement
  - Timer expiration enhancement and WebSphere MQ V7.0.1 Support for Message Expiry
### 4.1 APPC and OTMA shared queues enhancement

The target market of this enhancement is Advanced Program-to-Program Communication (APPC) and OTMA clients who use synchronous shared queues transactions. This is a major enhancement and has the potential for reducing CPU cost.

The capability to use shared queue for synchronous messages was introduced with IMS 8 and was based on Resource Recovery Services (RRS) to support the cascaded transactions between front-end and back-end IMS systems. However, RRS has a CPU cost and complicates the diagnostic in case of problem. Various clients had implemented pseudo wait for input (WFI) regions for a part of the workload to privilege a front-end IMS, and to reduce cascading in transaction processing and lower RRS consumption.

<table>
<thead>
<tr>
<th>Synchronous or asynchronous APPC transactions: APPC transactions can be synchronous or asynchronous.</th>
</tr>
</thead>
<tbody>
<tr>
<td>OTMA requests from remote clients can be either Send-Receive or Send-Only. The Send-Receive interaction supports the use of both Commit Mode 1 and Commit Mode 0. Send-Only interactions only support Commit Mode 0. Commit Mode 1 is considered to be a synchronous transaction request and Commit Mode 0 is asynchronous.</td>
</tr>
<tr>
<td>All synchronous transactions have a synchronization level (synclvl):</td>
</tr>
<tr>
<td>NONE or Synchronization Level 0 (SL0)</td>
</tr>
<tr>
<td>CONFIRM or Synchronization Level 1 (SL1)</td>
</tr>
<tr>
<td>SYNCPT or Synchronization Level 2 (SL2)</td>
</tr>
<tr>
<td>IMS 12 removes the RRS dependency for the synchronous transactions with a synchronization level NONE or CONFIRM. IMS 12 uses cross-system coupling facility (XCF) communication between shared queue's front-end and back-end systems instead of RRS. The benefit of this enhancement is a performance improvement and a simplification of the syncpoint process.</td>
</tr>
<tr>
<td>Important: Transactions with synchronization level SYNCPT still require RRS.</td>
</tr>
</tbody>
</table>

As mentioned, the synchronous APPC and OTMA shared queues enhancement was introduced with IMS 8. Before IMS 12, APPC and OTMA shared message queue enablement used RRS Multisystem Cascaded Transaction support to synchronize IMS systems in a sysplex.

RRS had to be turned on with the DFSPBxxx RRS parameter. Then, the control of this enablement was done through the DFSDCxxx AOS parameter. It had three possible values for this sharing:

| Y | Activate |
| N | Not activate |
| F | Force |

The values Y, N, and F are still available with IMS 12, and three new parameters offer the possibility to use XCF communications for transactions using SL0 and SL1.

Shared queues environments that are set with IMS versions before IMS 12 can still run with the new IMS 12 members with no new settings. You can plan the synchronous shared queues migration to XCF as a separate project.
The following sections describe the new values of the AOS parameter, various migration and setup considerations, and basics about operations and performance:

- Understanding the DFSDCxxx proclib parameters
- Setup and migration considerations
- Operations
- Performance considerations

### 4.1.1 Understanding the DFSDCxxx proclib parameters

You use the `AOS=` parameter of DFSDCxxx proclib member to enable IMSplex users to execute transactions that are originated by APPC or OTMA on a back-end system. AOS stands for APPC OTMA Synchronous.

The `AOS=` parameter includes the options listed in Figure 4-1 to manage the synchronous APPC/OTMA shared message queue (SMQ) support.

![Figure 4-1 SMQ parameters in DFSDCxxx proclib member](image)

With IMS 12, the AOS parameter has new values to indicate that XCF will be used rather than RRS to process APPC and OTMA synchronous transactions.

The front-end designates an IMS system that receives a message, sets the XCF indicator according to the DFSDCxxx settings, sets or unsets affinity, and then puts the message in the shared queue.

The back-end designates an IMS system that gets a message from the shared queue, schedules this message in a region, and then sends the response back. In some cases, the back-end is the same as the front-end.

The result of the AOS setting depends on the registration of IMS to RRS, which is controlled with the RRS parameter in the DFSPBxxx proclib member. It can be set to Yes or No.

Table 4-1 explains the meaning of the combination of setting the two parameters.

<table>
<thead>
<tr>
<th>AOS</th>
<th>RRS</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>Y/N</td>
<td>APPC/OTMA SMQ local status is INACTIVE (default). Transactions are queued with affinity and processed by the front-end.</td>
</tr>
</tbody>
</table>
| F   | N   | APPC/OTMA SMQ local status INACTIVE
|     |     | - If front-end, IMS enqueues messages with affinity. (Front-end, in this case, needs RRS to create the parent UR and suppress the affinity.)
|     |     | - If back-end, IMS schedules messages having no affinity from the queue:
|     |     |   - If the message has no XCF indicator (it from a front-end with RRS), the transaction abends with U0711.
|     |     |   - If the XCF indicator on, the message is scheduled normally. |
For APPC and OTMA transactions, the new DFSDCxxx AOSLOG parameter specifies whether the front-end system writes an X’6701’ log record for the following cases (using a value of Y or N as appropriate):

- A response message is returned from the back-end system by XCF for transactions with all synchronization levels.
- An error message is returned from the back-end system by XCF for transactions with all synchronization levels of NONE, CONFIRM, and SYNCPT.

AOSLOG=N is the default.

<table>
<thead>
<tr>
<th>AOS</th>
<th>RRS</th>
<th>Description</th>
</tr>
</thead>
</table>
| F   | Y   | APPC/OTMA SMQ local status FORCE-RRS and Multisystem Cascaded Transaction using RRS:  
  - If front-end, IMS enqueues messages with no affinity and creates a parent UR using RRS.  
  - If back-end, IMS schedules messages having no affinity, regardless of whether the XCF indicator is set on.  
  If such a system loses RRS, the other systems with AOS=F continue to work with this function active. |
| Y   | N   | APPC/OTMA SMQ local status INACTIVE:  
  - If front-end, IMS enqueues messages with affinity and schedules them.  
  - If back-end, IMS schedules any message having no affinity. |
| Y   | Y   | APPC/OTMA SMQ local status ACTIVE-RRS and Multi-System Cascaded Transaction using RRS:  
  - If front-end, IMS enqueues messages with no affinity and creates a parent UR using RRS.  
  - If back-end, IMS schedules messages with no affinity, whether the XCF indicator is set on or off.  
  If such a system loses RRS, all systems with AOS=Y from the shared queue group stop working with this function. |
| B, S, X | Y   | APPC/OTMA SMQ local status ACTIVE-XCF:  
  - If front-end, IMS enqueues messages with no affinity and sets the XCF indicator on.  
  - If back-end, IMS schedules messages having no affinity, regardless of whether the XCF indicator is set.  
  B, S, and X are equivalent for SL0 and SL1 processing:  
  - B is equivalent for SL2 processing to AOS=Y.  
  - S is equivalent for SL2 processing to AOS=F.  
  - X is equivalent for SL2 processing to AOS=N. |
| B, S, X | N   | APPC/OTMA SMQ local status ACTIVE-XCF:  
  - If front-end, IMS enqueues messages with no affinity and sets the XCF indicator on.  
  - If back-end, IMS schedules messages having no affinity, but if the XCF indicator is set off, the transaction abends with U0711.  
  SL2 processing not possible because RRS not there. |

Important: Do not mix RRS=Y and RRS=N in a shared queue group with members having AOS=Y or AOS=F.

SL2 messages always need RRS. If RRS is not present, SL2 messages cannot be processed.
For more information about DFSDCxxx and DFSPBxxx settings, see *IMS Version 12 System Definition*, GC19-3021.

**Color guide for Figure 4-2:** Figure 4-2 uses the following color scheme:

- IMS members in blue have RRS=N, and IMS members in purple have RRS=Y.
- The upper blue dash-dotted lines represent how a member enqueues the message to the shared queue, with affinity or not, with the XCF bit set or unset. In such cases, the member is front-end.
- The lower green dotted lines represent all the kinds of messages a member can schedule from the shared queue.
- The red dashed lines are two cases where you get abendU0711. In such cases, it is back-end.

![Figure 4-2 - AOS and RRS combinations at a glance](image)

**IMS log records indication**

The XCF indicator is set in Log Record 01. When LUY_FLAGS=x'40' at offset x'B4', LUY_XCF_INDICATOR is set on and a system with AOS=X, B, or S has enqueued this message.

### 4.1.2 Setup and migration considerations

If you have the APPC and OTMA synchronous shared queue function already activated using an IMS version other than IMS 12, keep in mind the considerations we discuss in the following sections when you plan a migration to use XCF.
**RECON minimum version consideration**

**Important:** IMS systems in a shared queues group must be at MINVERS 12.1 to enable this enhancement.

If MINVERS is not at version 12.1, message DFS2088I RSN40 might be sent, depending on the AOS/RRS combination.

Table 4-2 outlines the steps.

**Table 4-2  MINVERS influence on APPC/OTMA SMQ enablement**

<table>
<thead>
<tr>
<th>Member version</th>
<th>RECON MINVERS</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; v12</td>
<td>&lt; 12.1</td>
<td>AOS=N, Y, or F available. AOS=X, B, and S not supported.</td>
</tr>
<tr>
<td>&lt;12</td>
<td>=12.1</td>
<td>Recovery control (RECON) access not allowed.</td>
</tr>
<tr>
<td>=12</td>
<td>&lt;12.1</td>
<td>If AOS=B and RRS=Y/N, or AOS=S and RRS=N, the DFS2088I RSN40 APPC/OTMA SMQ support not activated message is sent. If AOS=S and RRS=Y, AOS=S is changed to AOS=F and the message DFS2089I APPC/OTMA SMQ Enablement active, RRS is used is sent.</td>
</tr>
<tr>
<td>=12</td>
<td>=12.1</td>
<td>AOS=X, B, and S full support.</td>
</tr>
</tbody>
</table>

You can check RECON MINVERS by using the `LIST RECON STATUS` command (Example 4-1).

**Example 4-1 LIST RECON STATUS and MINVERS V12**

<table>
<thead>
<tr>
<th>11.215 04:12:08.141311</th>
<th>LISTING OF RECON</th>
<th>PAGE 0003</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECON</td>
<td>RECOVERY CONTROL DATA SET, IMS V12R1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DMBE=23 INIT TOKEN=11192F0745595F</td>
<td></td>
</tr>
<tr>
<td></td>
<td>FORCER LOG DSN CHECK=CHECK44 STARTNEW=NO</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TAPE UNIT= DASD UNIT=SYSALLDA TRACENOFF SSID=I12A</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LIST DLOG=YES CA/IC/LOG DATA SETS CATALOGED=YES</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MINIMUM VERSION = 12.1 CROSS DBRC SERVICE LEVEL ID= 00001</td>
<td></td>
</tr>
<tr>
<td></td>
<td>REORG NUMBER VERIFICATION=NO</td>
<td></td>
</tr>
<tr>
<td></td>
<td>LOG RETENTION PERIOD=00.001 00:00:00.0</td>
<td></td>
</tr>
</tbody>
</table>

**Activating the proclib new setting**

**Important:** New proclib AOS parameter values need /NRE or /ERE to be activated.

**Mixed environment RRS/XCF considerations**

If the workload includes only SL0/SL1 messages, migrate any AOS=Y or F environment to AOS=X. In that way, all messages are placed in queue with no affinity. Those messages from an XCF front-end will have the XCF indicator on. As soon as RRS is active everywhere, all the RRS or XCF messages are scheduled on every back-end.

Keep the setting RRS=Y until all members in the shared queue group are AOS=X. At that point, and only if there are no applications or functions working with RRS or SL2 messages, there is no reason to keep it active. However, if the workload includes SL2 messages, use AOS=B for
an equivalent to $A0S=Y$, and use $A0S=S$ for an equivalent to $A0S=F$. In such cases, keep RRS active because SL2 always needs RRS.

### 4.1.3 Operations

This section illustrates the old process flow using RRS and the new branded process flow using XCF and how it is seen through the OLDS using the interactive IMS Problem Investigator IBM tool. For more information about this tool, see “IMS Problem Investigator” on page 402.

Assume that tests have been run on a two-way shared queue. IMSplex and IVP transactions have been sent from an IMS Connect client using a REXX program, sending a message including an IRM with CM1 and SL0.

#### Operation using RRS

The IMSPLEX supports synchronous APPC and OTMA messages when the following conditions are true:

- All IMS systems in the shared queues group are Version 9 or higher.
- RRS is active.
- The IMS control region parameter $RRS=Y$ is defined for all IMS systems in the IMSplex.
- $A0S=Y$ or $A0S=F$ is specified in the DFSDCxxx PROCLIB member data set.

For more considerations about processing in this environment, see “Managing APPC and OTMA messages in a sysplex environment” in *IMS Version 12 System Administration*, SC19-3020.

#### High level flow using RRS

Figure 4-3 shows the high level flow of a synchronous message. (Therefore, it does not show all possible interactions.) The flow works the same in IMS 12 as it worked in the previous releases since IMS 8.

![Figure 4-3 General flow of a synchronous message using RRS](image-url)
A synchronous message passes through the following flow as illustrated in Figure 4-3 on page 115:

1. When an input message is received, IMS determines if the message is synchronous, checks to see if the shared queues capability is enabled, and invokes RRS callable services to establish the RRS working environment.

2. The input message is placed on the global shared queues to be processed by an available IMS, that is, an IMS where the transaction is defined and has an available message region. In this example, the back-end system is immediately available to process the transaction.

3. The back-end IMS invokes RRS callable services to establish the RRS environment and perform message synchronization.

4. The message is processed in the back-end and the IOPCB reply sent to the front-end. All IMS resources are held pending syncpoint processing until either a commit or backout.

5. Depending on the size of the IOPCB reply and whether this is a conversational transaction, the message is either sent by using XCF services or routed through shared queues. An indicator is received.

6. The front-end IMS sends the message and interfaces with the partner client following the rules of synclvl processing (None, Confirm, or Syncpoint).

7. Based on the success or failure of the partner interaction, the front-end IMS invokes RRS to either commit or back out.

8. The RRS commit or backout is communicated to the back-end IMS for the corresponding commit or backout.

9. RRS on both sides provides the support for the synchronization of commit or backout.

10. At the completion of the commit or backout, the front-end IMS interacts with the partner program to either terminate the connection or get the next message.

**Setting up and monitoring by using RRS**

Set DFSDCxxx AOS to Y or F, and set DFSPBxxx RRS to Y. Example 4-2 shows the messages and command output during a normal startup with AOS=F, RRS=Y on two IMS members named I12A and I12C.

**Example 4-2   Startup messages and OTMA CM1 SL0 transaction monitoring messages with RRS**

**First IMS Startup messages:**
05.36.04 STC02823 DFS0653I PROTECTED CONVERSATION PROCESSING WITH RRS/MVS ENABLED I12A
05.36.04 STC02823 DFS2089I APPC/OTMA SMQ Enablement active. RRS is used I12A

/DIS A DC shows:
05:40:24.77 STC02823 00000090 DFS000I APPC/OTMA SHARED QUEUE STATUS - LOCAL=FORCE-RRS
GLOBAL=ACTIVE-RRS I12A
05:40:24.78 STC02823 00000090 DFS000I APPC/OTMA SHARED QUEUES LOGGING=N I12A

**Second IMS Startup messages:**
05.42.20 STC02834 DFS0653I PROTECTED CONVERSATION PROCESSING WITH RRS/MVS ENABLED I12C
05.42.20 STC02834 DFS2089I APPC/OTMA SMQ Enablement active. RRS is used I12C

/DIS A DC shows:
999 + APPC/OTMA SHARED QUEUE STATUS - LOCAL=FORCE-RRS GLOBAL=ACTIVE-RRS E-RRS
999 APPC/OTMA SHARED QUEUES LOGGING=N

/DIS UOR shows:
RESPONSE=SC63
DFS4444I DISPLAY FROM ID=I12C
ST P-TOKEN  PSBNAME  RRS-URID  IMS-RECTOKN
+A DFSIVP1 C82811C07E76AFA00000004B01010000 I12C 0000002700000002
   WID=206401F8NATIVE CASCADED TRAN
*11213/083557*
      650 00000090  /192454
      650 00000090  *11213/083250*

Example 4-3 shows the log records that you can find at the front-end.

Example 4-3  Message flow on the front-end of I12A for an RRS scenario

---
01 Input Message  11.45.04.859413
UTC=11.45.04.859406 TranCode=IVTNO Userid=IMSR6 LTerm=7100
Terminal=7100 OrgUOWID=I12A/C8324BB4B3CEA546 Port=7100
LogToken=C8324BA78ABD02004 SSN=07 Socket=TRAN CM=1 SL=0 Source=Connect
Message associated with CQSPUT MSGUFLG1=x'80'
---
35 Input Message Enqueue  11.45.04.859425
UTC=11.45.04.859406 TranCode=IVTNO Userid=IMSR6 LTerm=7100
Terminal=7100 OrgUOWID=I12A/C8324BB4B3CEA546 Port=7100
LogToken=C8324BA78ABD02004 SSN=07 Socket=TRAN CM=1 SL=0
Message enqueued to the shared queue
---
33 Free Message  11.45.04.859694
OrgUOWID=I12A/C8324BB4B3CEA546
---

Example 4-4 shows the log records you can get at the back-end.

Example 4-4  Message flow on back-end I12C for an RRS scenario

---
08 Application Start  11.45.04.860883
UTC=11.45.04.860878 TranCode=IVTNO Region=0001
RecToken=I12C/0000000600000000 RegTyp=MPP TClass=01 TPrtty=01
Application Program scheduled
---
5607 Start of UOR  11.45.04.860884
UTC=11.43.23.069649 Program=DFSIVP1 Region=0001 IMSID=I12C
RecToken=I12C/0000000600000000 UOR processing on back-end I12C
---
01 Input Message  11.45.04.861068
UTC=11.45.04.859406 TranCode=IVTNO Userid=IMSR6 LTerm=7100
Terminal=7100 OrgUOWID=I12A/C8324BB4B3CEA546 Port=7100
LogToken=C8324BA78ABD02004 SSN=07 Socket=TRAN CM=1 SL=0 Source=Connect
---
31 DLI GU  11.45.04.861082
UTC=11.45.04.861080 TranCode=IVTNO Region=0001
OrgUOWID=I12A/C8324BB4B3CEA546 RecToken=I12C/0000000600000000
Message picked up from the shared queue / front-end = I12A
---
5616 Start of protected UOW  11.45.04.862476
Region=0001 IMSID=I12C RecToken=I12C/0000000600000000
Interest has been registered with RRS for this UOW
TPCPDATA shows NATIVE CASCADED
---
When an output IOPCB reply is sent, it is always sent before sync point. If the client partner is unavailable or the output reply cannot be sent, the following actions occur:

- If the transaction that replies to the IOPCB executes on the front-end, the default action is to abend the transaction with a U0119 and discard the output reply. User exit DFSCMUX0 (Message Control/Error Exit Routine) can be implemented to change the default action. The exit can change the default abort action (except for sync-level of sync point, which must continue with the abort). This is how errors with synchronous message replies have been processed for several releases of IMS.

- If the transaction that replies to the IOPCB executes in a back-end IMS and the front-end IMS cannot deliver the message, an RRS Take Backout command is issued which forces backout processing to occur on the back-end. The back-end aborts the transaction with U0711-1E.
For any RRS error occurring before sending the output, abend U0711 is sent and the transaction is stopped.

For more information about coding DFSCMUX0, see *IMS Version 12 Communications and Connections*, SC19-3012, and *IMS Version 12 Exit Routines*, SC19-3016.

**Operation using XCF**

XCF communications for synchronous SL0 and SL1 transactions can be used if the following statements are true:

- A0S=B or S is specified in the DFSDCxxx PROCLIB member data set.
- All IMS systems in a shared queue group must be at MINVERS (see 4.1.2, “Setup and migration considerations” on page 113).
- /ERE or /NRE after the DFSDCxxx PROCLIB modification.

**High level flow for synchronous transactions using XCF**

Figure 4-4 shows the general process flow of a synchronous SL0 transaction.

![Figure 4-4 Synchronous SL0 processing with XCF communication](image)

A synchronous SL0 transaction follows this flow as illustrated in Figure 4-4:

1. A client application enters OTMA CM1 or APPC synchronous conversation transaction with SL0 from the front-end IMS system.
2. The input transaction message with the XCF indicator is put on the SQ.
3. A dependent region on the back-end IMS system retrieves and processes the input message.
4. The back-end IMS system sends the transaction response to the FE and waits for confirmation.
5. The front-end forwards the transaction response to the client application.
6. The front-end sends the receive-confirmation to the back-end and waits for the commit completion.

7. The back-end receives the receive-confirmation, completes the commit process, and sends back the commit completion message to the front-end.

8. The front-end forwards the commit completion message to the client application.

Figure 4-5 illustrates the general process flow for a synchronous SL1 transaction.

![Figure 4-5 Synchronous SL1 processing with XCF communication]

A synchronous SL1 transaction follows this flow as illustrated in Figure 4-5:

1. A client application enters an OTMA CM1 or APPC synchronous conversation transaction with SL1 from the FE IMS system.

2. The input transaction message with the XCF indicator is put on the SQ.

3. A dependent region on the back-end IMS system retrieves and processes the input message.

4. The back-end IMS system sends the transaction response to the front end and waits for an acknowledgement (ACK).

5. The front-end forwards the transaction response to the client application.

6. The client application returns an ACK.

7. The front-end forwards the ACK to the back end and waits for a commit completion.

8. The back-end receives the ACK, completes the commit process, and sends back the commit completion message to the front-end.

9. The front-end forwards the commit completion message to the client application.
Set up and monitoring using XCF

Set DFSDCxxx AOS to B or S. Then decide whether you want the front-end IMS to write 6701 log record by setting AOSLOG=Y. See Example 4-5.

Example 4-5  Startup messages and OTMA CM1 SL0 transaction monitoring messages with XCF

First IMS Startup messages I12A AOS=S RRS=Y
09.54.28 STC03112 DFS0653I PROTECTED CONVERSATION PROCESSING WITH RRS/MVS ENABLED I12A
09.54.28 STC03112 DFS2089I APPC/OTMA SMQ Enablement active. XCF and RRS are used I12A

/DIS A DC shows:
09.57.46 STC03112 DFS4444I DISPLAY FROM ID=I12A 924
924 VTAM STATUS AND ACTIVE DC COUNTS
924 VTAM ACB OPEN -LOGONS ENABLED
924 IMSLU=N/A,N/A APPC STATUS=DISABLED TIMEOUT= 0
924 OTMA GROUP=I12XOTMA STATUS=ACTIVE
924 + APPC/OTMA SHARED QUEUE STATUS - LOCAL=FORCE-RRS/XCF GLOBAL=ACTIV
924 E-RRS/XCF
924 APPC/OTMA SHARED QUEUES LOGGING=Y
924 + APPC/OTMA RRS MAX TCBS - 40 ATTACHED TCBS - 2 QUEUED RRSWKS-
924 0
924 APPLID=APPLI12A GRSNAME=IMS12XCF STATUS=ACTIVE

Second IMS Startup messages I12C AOS=S RRS=N
09.58.55 STC03125 DFS0698W PROTECTED CONVERSATION PROCESSING NOT ENABLED - RRS=Y NOT SPECIFIED I12C
09.58.56 STC03125 DFS2089I APPC/OTMA SMQ Enablement active. XCF is used I12C

/DIS A DC shows:
10:52:58.00 STC03125 00000090 DFS4444I DISPLAY FROM ID=I12C 598
598 00000090 VTAM STATUS AND ACTIVE DC COUNTS
598 00000090 VTAM ACB OPEN -LOGONS ENABLED
598 00000090 IMSLU=N/A,N/A APPC STATUS=DISABLED TIMEOUT= 0
598 00000090 OTMA GROUP=I12XOTMA STATUS=ACTIVE
598 00000090 + APPC/OTMA SHARED QUEUE STATUS - LOCAL=ACTIVE-XCF GLOBAL=ACTIV
598 00000090 E-RRS/XCF
598 00000090 APPC/OTMA SHARED QUEUES LOGGING=Y
598 00000090 + APPC/OTMA RRS MAX TCBS - 40 ATTACHED TCBS - 2 QUEUED RRSWKS-
598 00000090 0
598 00000090 APPLID=APPLI12C GRSNAME=IMS12XCF STATUS=ACTIVE

I12A message when I12C joins:
09.58.54 STC03112 DFS0778I I12A, MSC SUBSYSTEM I12C HAS JOINED THE SHARED QUEUES GROUP, I12A

Example 4-6 and Example 4-7 show the log records for a message processing where XCF is used between front-end and back-end. Example 4-6 shows the message flow on the front end.

Example 4-6  Message flow on front-end I12A for an XCF scenario
01 Input Message 09.07.25.206790
UTC=09.07.25.206782 TranCode=IVTNO Userid=IMSR6 LTerm=7100
Terminal=7100 OrguOWID=I12A/C8322877464D5E0C Port=7100
LogToken=C83228699B0A680C SSN=06 Socket=TRAN CM=1 SL=0 Source=Connect
Message associated with CQSPUT MSGUFLG1=x’80’
LUY_FLAGS=x’40’ LUY_XCF_INDICATOR ON
----------------------------------------------------------------------------------
35 Input Message Enqueue 09.07.25.206801
Example 4-7 shows the message flow on the back-end.

Example 4-7  Message flow on back-end I12C for an XCF scenario

08  Application Start  09.07.02.876952
UTC=09.07.02.876946 TranCode=IVTNO Region=0003
RecToken=I12C/0000000500000000 RegTyp=MPP TClass=01 TPrtty=01

Application Program is scheduled

5607  Start of UOR  09.07.02.890783
UTC=19.35.27.208423 Program=DFSIVP1 Region=0003  IMSID=I12C
RecToken=I12C/0000000500000001

UOR processing on back-end I12C

01  Input Message  09.07.25.207425
UTC=09.07.25.206782 TranCode=IVTNO Userid=IMSR6 LTerm=7100
Terminal=7100 OrguOWID=I12A/C8322877464D5E0C Port=7100
LogToken=C83228699B0A680C SSN=06 Socket=TRAN CM=1 SL=0 Source=Connect

Message picked up from the shared queue / front-end = I12A

50  Database Update  09.07.25.227814
UTC=09.07.25.227813 Program=DFSIVP1 Userid=IMSR6 Database=IVPDB1
RBA=00000002 Region=003 RecToken=I12C/0000000500000001

03  Output Message Response  09.07.25.244638
UTC=09.07.25.206782 TranCode=IVTNO Userid=IMSR6 LTerm=7100
Terminal=7100 OrguOWID=I12A/C8322877464D5E0C Port=7100
LogToken=C83228699B0A680C SSN=06 Socket=TRAN CM=1 SL=0 Source=Connect

Data response put in a message queue buffer
MSGORGID=I12A (origin)  and  MSGPROID=I12C(process)
MSGU1PUT=x’80’  Message associated with CQSPUT

31  Message GU for APPC  09.07.25.244648
UTC=09.07.25.244642 TranCode=IVTNO Userid=IMSR6 LTerm=7100
Terminal=7100 OrguOWID=I12A/C8322877464D5E0C
RecToken=I12C/0000000500000001 Port=7100 LogToken=C83228699B0A680C
SSN=06 Socket=TRAN CM=1 SL=0
QLGUORID=I12A (origin)  QLGUPRID=I12C (process)
QLGUUFLG1=x’80’  Message associated with CQSPUT

33  Free Message  09.07.25.246293
OrguOWID=I12A/C8322877464D5E0C
Diagnostic improvements

Logging and the /DISPLAY command output have been improved, and a new abend has been added to fit with the XCF case.

Front-end logging

When the AOSLOG=Y parameter is set on the front-end, or the /DIAGNOSE SET AOSLOG(ON) command is run, log record x'6701' is cut when the response messages or error messages are returned from the back-end by using XCF (Example 4-8 on page 123). This is true for all synchronization levels (none, confirm or syncpt).

With AOSLOG=Y, IMS also generates a new record of TIB3. APAR PM45923 and PTF UK71520 must be applied to have the log record x'6701' cut properly without setting a trace.

Example 4-8  'X'6701' diagnostic log record in a front-end system process flow

<table>
<thead>
<tr>
<th>01</th>
<th>Input Message</th>
<th>17.26.34.001126</th>
</tr>
</thead>
<tbody>
<tr>
<td>UTC=17.26.34.001122 TranCode=IVTNO Userid=IMSR6 LTerm=7100 Terminal=7100 OrgUOWID=I12A/C8329808AB82338C Port=7100 LogToken=C83297F6AD385306 SSN=04 Socket=TRAN CM=1 SL=0 Source=Connect</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| 35 | Input Message Enqueue | 17.26.34.001137 |
|-----------------|---------------------|
| UTC=17.26.34.001122 TranCode=IVTNO Userid=IMSR6 LTerm=7100 Terminal=7100 OrgUOWID=I12A/C8329808AB82338C Port=7100 LogToken=C83297F6AD385306 SSN=04 Socket=TRAN CM=1 SL=0 |

| 33 | Free Message | 17.26.34.001508 |
|----------------|-----------------|
| OrgUOWID=I12A/C8329808AB82338C |

| 6701 | YSND XCF message sent to OTMA client | 17.26.34.007908 |
|-------------------|-----------------|
| UTC=17.26.34.007905 Terminal=7100 Port=7100 LogToken=C83297F6AD385306 SSN=05 ID=YSND |

For more information about the /DIAGNOSE command, see IMS Version 12 Commands, Volume 1: IMS Commands A-M, SC19-3009.

This log record retrieves diagnostic information for system resources, such as IMS control blocks, user-defined nodes, or user-defined transactions, at any time without taking a console dump.
If **AOSLOG(ON)** is specified in a non-shared queues environment, or when **AOS=N** is specified in the DFSDCxxx PROCLIB member, the **/DIAGNOSE SET AOSLOG(ON)** command is rejected with a DFS2859I message.

**New abend U0109 in the back-end IMS**
The U0109 abend means that IMS did not deliver a response for the APPC synchronous or OTMA (CM1) transaction, or that the LU 6.2 device or OTMA client returned a negative acknowledgment (NAK) to the output message of the synlevel(confirm) synchronous CM1 transaction using XCF communications:

- RC=01 means the -end received a NAK from the front-end for the SEND of the output message of a send-then-commt (CM1) transaction with sync level of NONE or CONFIRM (SL0 or SL1) where XCF communication is used.
- RC=02 means the front-end terminated or restarted.

The transaction instance is ended, but the program and transaction are not stopped.

For more information about this abend, see *IMS Version 12 Messages and Codes, Volume 3: IMS Abend Codes*, GC19-9714.

**Enhancement to the output of /DISPLAY A DC command**
The **/DISPAY A DC** command shows new APPC/OTMA shared queues support statuses:

- **ACTIVE-RRS** Changed from ACTIVE, where **AOS=Y** and **RRS=Y** are specified.
- **ACTIVE-XCF** A new status, where **AOS=X** is specified; or **AOS=B** and **RRS=N** are specified.
- **ACTIVE-RRS/XCF** A new status, where **AOS=B** and **RRS=Y** are specified.
- **FORCE-RRS** Changed from FORCE, where **AOS=F** is specified.
- **FORCE-RRS/XCF** A new status, where **AOS=S** is specified.

New statuses are available for **APPC/OTMA SHARED QUEUE STATUS – LOCAL=status1 GLOBAL=status2**:

- Status1 can be one of the following options:
  - ACTIVE-RRS
  - ACTIVE-XCF
  - ACTIVE-RRS/XCF
  - FORCE-RRS
  - FORCE-RRS/XCF
  - INACTIVE
  - UNSUPPORTED

- Status2 can be one of the following options:
  - ACTIVE-RRS
  - ACTIVE-XCF
  - ACTIVE-RRS/XCF
  - CHECK
  - INACTIVE

**Enhancement to the output of /DISPLAY A REG command**
Two new values, **WAIT-XCF** and **TERM-WAIT-XCF**, have been added to describe the MPP status in the output of the **/DISPLAY** command.

WAIT-XCF is a new status that is shown when an application program is the middle of processing a sync point for an APPC or OTMA transaction with a synchronization level of
NONE or CONFIRM. Sync point can continue after the client issues either an ACK or a NAK.
XCF is used to communicate between the back-end and the front-end IMS systems.

TERM-WAIT XCF is another new status that is displayed when a dependent region termination is in progress and the application in the region is still processing a sync point for an APPC or OTMA transaction with sync level of NONE or CONFIRM. Sync point can continue after the client issues either an ACK or NAK. When a dependent region is found in this state, a continuation line is inserted into the display, which shows either the transaction member (TMEMBER) and the transaction pipe (TPIPE) for OTMA client or the network ID (NETWORKID) and the logic unit name (LUNAME) for APPC client, in addition to the originating IMS system ID (ORIGIN) that is associated with the transaction processing in the dependent region.

Example 4-9 illustrates the new statuses for the /DIS A REG command when using XCF.

**Example 4-9   New statuses for the /DIS A REG command when using XCF**

<table>
<thead>
<tr>
<th>DFC000I</th>
<th>REGID</th>
<th>JOBNAME</th>
<th>TYPE</th>
<th>TRAN/STEP</th>
<th>PROGRAM</th>
<th>STATUS</th>
<th>CLASS</th>
<th>IMS1</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFS000I</td>
<td>REGID</td>
<td>JMPRGN</td>
<td>JMP</td>
<td>NONE</td>
<td>IMS1</td>
<td></td>
<td>IMS1</td>
<td></td>
</tr>
<tr>
<td>DFS000I</td>
<td>REGID</td>
<td>1 IMSMPPA</td>
<td>TPI</td>
<td>APOL11</td>
<td>IMS1</td>
<td>WAIT-RRS/PC</td>
<td>IMS1</td>
<td></td>
</tr>
<tr>
<td>DFS000I</td>
<td>URID:</td>
<td>C2D68917DEB2000000000001010000 ORIGIN: IMS2</td>
<td>IMSSPPB</td>
<td>TPI</td>
<td>APOL12</td>
<td>IMS1</td>
<td>TERMIN-WAIT RRS</td>
<td>IMS1</td>
</tr>
<tr>
<td>DFS000I</td>
<td>URID:</td>
<td>C2D68917DE8300000000001010000 ORIGIN: IMS2</td>
<td>IMSSPPC</td>
<td>TPI</td>
<td>APOL13</td>
<td>IMS1</td>
<td>WAIT-XCF</td>
<td>IMS1</td>
</tr>
<tr>
<td>DFS000I</td>
<td>TMEM:</td>
<td>HWS1</td>
<td>TPIPE</td>
<td>CLIENT1</td>
<td>IMS1</td>
<td>TERMIN-WAIT XCF</td>
<td>IMS1</td>
<td></td>
</tr>
<tr>
<td>DFS000I</td>
<td>LUNAME:</td>
<td>IMSNETWK.LU62IMS1 ORIGIN: IMS2</td>
<td>JMPPGN</td>
<td>JBP</td>
<td>NONE</td>
<td>IMS1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DFS000I</td>
<td>BATCHREG</td>
<td>BMP</td>
<td>NONE</td>
<td>IMS1</td>
<td></td>
<td>IMS1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DFS000I</td>
<td>FPRGN</td>
<td>FP</td>
<td>NONE</td>
<td>IMS1</td>
<td></td>
<td>IMS1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DFS000I</td>
<td>DBTRGN</td>
<td>DBT</td>
<td>NONE</td>
<td>IMS1</td>
<td></td>
<td>IMS1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DFS000I</td>
<td>DBRICTAB</td>
<td>DBRC</td>
<td>IMS1</td>
<td>IMS1</td>
<td></td>
<td>IMS1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

4.1.4 Performance consideration

The IMS 12 OTMA synchronous shared queues enhancement with XCF for Commit Mode 1 (send-then-commit) provides significant CPU efficiency improvements leading to increased throughput compared to RRS for Commit Mode 1 transactions with a sync-level of NONE or CONFIRM.

**Important:** Based on our performance evaluation, we have seen a transaction rate (external throughput rate (ETR)) improvement of 6–18% and path-length (internal throughput rate (ITR)) improvement of 27–81%.

All measurements were conducted within a stable and isolated environment using a full-function workload.

The ETR is the actual arrival rate of transaction per second. The ITR is a calculated by determining the throughput if the processor is running at 100%.
4.2 OTMA ACEE reduction for multiple OTMA clients

By using the OTMA Access Control Environment Element (ACEE) reduction for multiple OTMA clients capability introduced in IMS 12, you can create, share, and cache a single ACEE associated with a Resource Access Control Facility (RACF) user ID, regardless of the instances of the OTMA member clients (TMEMBER). It also reduces the maximum ACEE aging value, and implements a rule to always take the lowest value when there are multiple clients. This function helps in reducing storage use and improves security.

Important: Do not confuse this capability with the ACEE caching done with the IMS Connect Extensions IBM Tool, which is related to IMS Connect security validation.

4.2.1 Overview of OTMA enhancement

The OTMA enhancement applies to ACEE caching and the modification of the maximum aging value of ACEEs.

ACEE caching

To fully benefit from the availability, scalability, and performance advantages of the z/OS environment, multiple instances of OTMA member clients (TMEMBERS), such as IMS Connect, DB2 Stored Procedures, and WebSphere MQ, can run in parallel on multiple servers or LPARs while still being able to connect to the same IMS.

In previous releases, IMS OTMA isolated the security environment of one member client from another. This meant that an Access Control Environment Element (ACEE) for a user ID had to be created for each OTMA member client instance if the same user ID accessed IMS through more than one path (IMS Connect and WebSphere MQ). Creating multiple copies of the same ACEE resulted in increased storage usage in subpool 249 in Extended Private storage.

Important: In IMS 12, OTMA caches the ACEE so that only one copy exists for the same user even when messages from that user are sent in through different member clients. The cached ACEEs also reside in subpool 249.

New maximum aging value for ACEEs

IMS 12 introduces a new maximum aging value for ACEEs. The ACEE aging value triggers when an ACEE is refreshed. This value is specified by the OTMA member client during the client-bid process. The maximum has been reduced from 68 years to 11.5 days. Each cached OTMA ACEE will have an aging value based on the OTMA member client using it that has specified the lowest number.

For example, if WebSphere MQ sets its ACEE aging value to 5 days while IMS Connect sets its ACEE aging value to 1 day, any ACEEs that only WebSphere MQ uses have an aging value of 5 days and any ACEEs that are only invoked through IMS Connect have an aging value of 1 day. After the ACEE, which was previous only used through WebSphere MQ, is invoked on behalf of the same user sending a message through IMS Connect, the aging value is set to 1 day, the lower of the 5 days versus the 1 day value.

This enhancement is important because some OTMA clients, such as the DB2 Stored Procedure DSNAIMS, set the "7FFFFFFF"X seconds (68 years) as the ACEE aging value for users sending IMS transactions and commands. The enhancement also detects obsolete ACEEs, improving security.
To enable ACEE refreshes, the OTMA member client needs to request a minimum ACEE aging value of 300 seconds during the client-bid process. If the aging value specified by an OTMA member client is less than 300 then OTMA resets the value to 0, meaning that ACEEs will not be refreshed. When an aging value OAAV value is sent to IMS 12 that is greater than 999999 seconds (11.5 days), OTMA resets that value to 999999 seconds. IMS 10 and IMS 11 allowed the value to be up to 2147483647 or 68 years.

### 4.2.2 OTMA enhancement implementation

This section explains how environments can benefit from this enhancement and illustrates how it works through an example.

**Challenge addressed**

The scenario illustrated in Figure 4-6 that references IMS 11 describes the issue when a user accesses IMS using several paths. With sysplex distributor and multiple servers, a connection can be routed through more than one IMS Connect for the same IMS system.

Creating multiple ACEEs occupies storage in subpool 249 and requires calls to RACF to create each instance of an ACEE. For example, we have two IMS Connect member clients connected to IMS and each one submitted IMS transactions on behalf of the same 10,000 RACF user IDs. In this case, IMS OTMA creates 20,000 RACF ACEEs. Because the same user sends in messages through WebSphere MQ, a third ACEE is created for that user. The system storage for multiple copies of ACEEs on behalf of the same user can keep growing.

Additionally, a potential security exposure is introduced if a security change is made to the user profile but not all copies of the ACEE are refreshed.

For example, an IMS input transaction for a user ID by using an instance of an OTMA member client application can be rejected by OTMA. However, the same user ID sending in a message through a different path can be accepted by OTMA because the ACEE was created before the profile change and before a refresh of the ACEE is requested.

![Figure 4-6 OTMA ACEE reduction](image)

However, as shown with the connections to IMS 12, the issues described previously no longer apply to this environment. The ACEE is created when a transaction message on behalf of a
specific user ID first comes in from an OTMA member client and then is cached for reuse for subsequent transaction requests from the same user regardless of the communication path used.

How the OTMA enhancement works
Consider three IMS Connect users accessing the same data store I12A with different OTMA Aging Value settings in the IMS Connect Configuration Member (Figure 4-7).

Figure 4-7 illustrates the following process flow:
1. User Isabelle sends a transaction to data store I12A through HWSB. An ACEE is created for user ID Isabelle in I12A SP229 with an aging value set to 600.
2. User Angie sends a transaction to data store I12A through HWSA. An ACEE is created for user ANGIE with an aging value of 300.
3. User Isabelle sends a transaction to I12A through HWSA. An ACEE is already created for that user with a separate aging value than the one specified in HWSA. Therefore, IMS chooses the lowest value and will modify it in the ACEE already there. No new ACEE is created for that user, but the aging value is set to 300.
4. User Isabelle sends a transaction through HWSC where an aging value of 0 is specified. An aging value of 0 means that no cached ACEE will be used. A new ACEE is created and is used only one time.
The /DISPLAY OTMA command has been modified to show the ACEE aging value for each OTMA client; see Example 4-10.

**Example 4-10  /DISPLAY OTMA output**

```
/112ADIS OTMA

<table>
<thead>
<tr>
<th>GROUP/MEMBER</th>
<th>XCF-STATUS</th>
<th>USER-STATUS</th>
<th>SECURITY</th>
<th>TIB INPT</th>
<th>SMEM</th>
<th>DRUEXIT</th>
<th>T/O</th>
<th>ACEEAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>-I12AOTMA</td>
<td>ACTIVE</td>
<td>SERVER</td>
<td>FULL</td>
<td>0</td>
<td>8000</td>
<td>N/A</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>-HWBI12A1</td>
<td>ACTIVE</td>
<td>ACCEPT TRAFFIC</td>
<td>FULL</td>
<td>0</td>
<td>5000</td>
<td>I12S</td>
<td>120</td>
<td>300</td>
</tr>
<tr>
<td>-HWBI12A3</td>
<td>ACTIVE</td>
<td>ACCEPT TRAFFIC</td>
<td>FULL</td>
<td>0</td>
<td>5000</td>
<td>I12S</td>
<td>120</td>
<td>600</td>
</tr>
<tr>
<td>-HWBI12A2</td>
<td>ACTIVE</td>
<td>ACCEPT TRAFFIC</td>
<td>FULL</td>
<td>0</td>
<td>5000</td>
<td>I12S</td>
<td>120</td>
<td>0</td>
</tr>
<tr>
<td>-HWBI12C3</td>
<td>NOT DEFINED</td>
<td>SMQ BACKEND</td>
<td>FULL</td>
<td>0</td>
<td>5000</td>
<td>I12S</td>
<td>120</td>
<td></td>
</tr>
<tr>
<td>-I12S</td>
<td>SUPER MEMBER</td>
<td></td>
<td>FULL</td>
<td>0</td>
<td>5000</td>
<td></td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
```

### 4.2.3 Considerations and benefits

The OTMA enhancement has some impact on performance and security.

**Performance**

Because cached ACEEs are applicable to all OTMA TMEMBER clients, the requirement to issue the RACROUTE REQUEST=VERIFY command requests to create ACEEs for user IDs for a new OTMA client instance is reduced, also reducing possible RACF I/Os.

When an input transaction message carries a user ID whose ACEE has expired, OTMA uses an asynchronous request to create a new ACEE for this user ID, which will be kept in the cache for subsequent transaction requests. For the current transaction request, an ACEE is created for the authorization process and deleted when the authorization is complete.

From an operational perspective, the /SECURE OTMA REFRESH TMEMBER membername and /SECURE OTMA REFRESH commands have an identical impact because both commands work on the single OTMA ACEE table for all users.

**Security**

OTMA detects obsolete RACF ACEEs by using an internal IMS timer. Every two minutes, OTMA performs an ACEE cleanup operation for 10 expired user IDs.

IMS 12 changes the input ACEE aging value to 999999 seconds (11.5 days). This can increase the number of RACF I/Os and impact performance if more refreshes are done based on the aging value.

**Mixed environments**

In a mixed version environment, the difference in maximum aging value can cause an ACEE to be refreshed in an IMS 12 system but not in the IMS 10 or IMS 11 environments. (It can be 999999 seconds in IMS 12 and 2147483647 seconds in IMS 10 or 11.)

**Benefits**

Cached ACEEs reduce system storage requirements while providing better security and performance, as explained here:

- Having only one copy of the ACEE instead of multiples per OTMA client helps to reduce storage usage, security exposures, and improve performance.
- The same security result, regardless of which OTMA client is used, provides consistency.
By lowering the maximum ACEE aging value, you can trigger faster cache refresh. Security exposures are reduced because a user ID is revoked or permissions are changed.

4.3 OTMA DFS2082 for Commit Mode 0

The DFS2082 IMS message is sent instead of the application output reply to prevent the terminal from being hung in response mode. An OTMA send-then-commit (CM1) or APPC synchronous input message receives a DFS2082 message when the IMS application did not reply to the IOPCB or does a message switch to another transaction.

This section explains why the DFS2082 message has been implemented for Commit Mode 0 OTMA transaction and how it is requested.

4.3.1 Problem addressed

The DFS2082 (Response Mode transactions terminated without reply) message is a mechanism to release an outstanding wait for CM1 (Send-then-Commit) transactions when the IMS application flow across program-to-program switches terminates without inserting a reply to the IOPCB.

When converting remote programs that use CM1 to CM0 (Commit-the-Send), a problem can arise. Before IMS 12, a remote program that sends a CM0 transaction message and then waits for a response to the OTMA client can possibly wait a long time until a timeout occurs, if the IMS application does not send any reply to the IOPCB.

IMS OTMA in Version 12 resolves this potential problem by introducing a new commit-then-send (CM0) optional flag to request the DFS2082 message. When this new flag is specified for an input commit-then-send transaction and the IMS application either does not reply to the IOPCB or performs a message switch to another transaction, OTMA will send a DFS2082 message to the client regardless of the IMS transaction response mode.

This DFS2082 message for a commit-then-send transaction only occurs for the original input transaction and does not support the program-to-program switch. This restriction means that there is no DFS2082 message for a switched-to transaction, even if the switched-to transaction fails to reply and the original transaction does not reply either.

This function eases the CM1-to-CM0 application conversion by reducing the necessary timeout in remote applications.
4.3.2 Implementation

Figure 4-8 shows how this message is requested from various OTMA clients.

The new optional flag TMAMHRSP is for OTMA input commit-then-send transactions (CM0) only. When TMAMHRSP is specified for an input send-then-commit (CM1) transaction, IMS OTMA ignores it. The DFS2082 message has been supported for OTMA send-then-commit messages without the need of setting any OTMA message prefix flag.

IMS Connect uses this function by introducing a new IRM flag, IRM_F3_DFS2082. With this flag, the client applications can request the DFS2082 message for a CM0 input transaction. After the flag is set and the IMS application either does not reply to the IOPCB or message-switch to another transaction, OTMA sends a DFS2082 message to the client regardless of the IMS transaction response mode so that the client’s application does not need to wait for the timeout. This DFS2082 message for a CM0 input only occurs for the original input transaction and does not support IMS program-to-program switches.

IMS TM RA also uses this function by introducing a new InteractionSpec property, CM0Response. With this property, clients can request the DFS2082 message for a CM0 input transaction. After the flag is set and the IMS application either does not reply to the IOPCB or message switch to another transaction, OTMA sends a DFS2082 message to the client regardless of the IMS transaction response mode so that the client’s application does not need to wait for the timeout. This DFS2082 message for a CM0 input will only occur for the original input transaction and does not support IMS program-to-program switches.
4.4 Other Transaction Manager enhancements

The following additional enhancements have been set:

- In OTMA, to boost OTMA performance
- To let the DFSMSCE0 exit support the new shared queues enhancement with XCF
- To allow the DFSLUEE0 exit to dequeue an undeliverable asynchronous message
- To expand the transaction expiration function to WebSphere MQ

4.4.1 OTMA performance

The path length for OTMA transaction processing has been reduced by simplifying the logic when validating a tpipe name. Before this performance enhancement, OTMA experiences unnecessary CPU overhead because it always validates a tpipe name against keyword names when each message is received.

With the enhancement, OTMA performs tpipe validation only when a new tpipe name is received. For the subsequent transactions using the same tpipe name, OTMA no longer does tpipe name validation checking. Even for OTMA tmember clients like IMS Connect that only use 1 port tpipe for CM1 messages, validation checking will only be invoked one time instead of for every message from that port tpipe.

This enhancement is also available for IMS 10 and IMS 11 with the APARs PM20292 (V10) and PM20293 (V11).

4.4.2 TM and MSC Message Routing and Control exit routine (DFSMSC0) exit modified for shared queues

Users can also use the DFSMSCE0 TM and MSC Message Routing and Control user exit to specify the APPC synchronous conversation or OTMA CM1 transaction with sync level of NONE or CONFIRM to be queued with the RRS or the XCF indicator option.

- On input to the exit, the following indicators are set
  
  MSCE3XCF EQU X'80' A global XCF enabled IMS system.
  MSCE3RRS EQU X'40' A global RRS enabled IMS system.

- On output, the exit can set the following indicators:
  
  MSTR2XCF EQU X'02'
  Message is to be queued globally to the SQ with the XCF indicator.
  Only for synclvel of NONE or CONFIRM.

  MSTR2RRS EQU X'01'
  Message is to be queued globally to the SQ with the RRS indicator.
  Only for synclvel of NONE or CONFIRM.

If both options are set, IMS uses the XCF indicator option and ignores the RRS indicator option.

If the XCF indicator option is selected, IMS determines whether the message can be queued globally with the XCF indicator. If not, IMS determines whether the message can be queued globally with the RRS indicator. If not, IMS queues the message locally.
If the exit defaults to or sets the XCF indicator, IMS determines input transaction message queuing based on the following order:

1. Queue the input globally to the SQ with the XCF indicator if eligible.
2. Queue the input globally to the SQ with the RRS indicator if eligible.
3. Queue the input locally with the affinity to the FE system where the input is received.

If the exit selects the RRS indicator option, the input transaction message queuing is determined as follows:

1. Queue the input globally to the SQ with the RRS indicator if eligible.
2. Queue the input locally with the affinity to the FE system where the input is received.

### 4.4.3 LU 6.2 Edit exit routine (DFSLUEE0)

With the LU 6.2 Edit exit routine (DFSLUEE0), you can edit input and output LU 6.2 messages for IMS-managed LU 6.2 conversations. It is also called if a message is inserted from an alternate PCB destined for an LU 6.2 destination.

You can use this exit routine to perform the following tasks:

- Change the APPC local LU name of an asynchronous LU 6.2 outbound conversation.
- Change the synchronization level of an asynchronous LU 6.2 conversation.
- View the contents of a message segment and continue processing.
- Change the contents of a message segment and continue processing.
- Discard a message segment.
- Perform a DEALLOCATE_ABEND of the LU 6.2 conversation.

For input messages, IMS calls the LU 6.2 Edit exit routine for each message segment before the message segment is inserted to the IMS message queue. The exit routine can edit message segments as necessary before the application program processes the input message.

For output messages, IMS calls the LU 6.2 Edit exit routine for each message segment before the message segment is sent to the LU 6.2 program. The exit routine can intercept the data sent by the application program and edit it for the particular destination.

For more information, see *IMS Version 12 Exit Routines*, SC19-3016, and *IMS Version 12 Communications and Connections*, SC19-3012.

In IMS 12, an enhancement to the LU6.2 Input/Output Edit Exit (DFSLUEE0) supports a new return code (RC=2) that requests the dequeuing of an undeliverable asynchronous output message. Previously, IMS always requeued the message.

**Important:** The new return code RC=2 is only valid for asynchronous conversations.

### 4.4.4 Transaction expiration and WebSphere MQ support

IMS 12 provides an extension of the WebSphere MQ Message Expiry facility to include the IMS transaction expiration function.

**Transaction expiration**

IMS Connect can adjust the expiration time for IMS transactions to match the timeout value of the socket connection on which the transaction is submitted. If an expiration time is specified
in IMS, transactions can expire and be discarded if IMS does not process them before the expiration time is exceeded.

A client application can submit a transaction request to IMS. IMS receives the transaction, processes it, and sends a reply. If IMS does not have the resources to process the transaction in the allotted time frame, the client application might time out the transaction call.

However, this transaction request might have been received by an IMS, but by the time the transaction is processed and a reply is sent, the client application no longer requires the response message. Processing unwanted transactions in IMS increases processing costs and CPU cycles.

The transaction expiration function has been introduced for the following versions:
- In IMS 10 for OTMA messages with APAR PK74017/UK50901
- In IMS 11 for all messages

Before IMS 12, OTMA checks if a transaction is expired at three points:
- When OTMA first receives a transaction from XCF
  If the expiration time has already passed, OTMA discards the transaction and returns a NAK message to the client.
- When OTMA enqueues a transaction in IMS
  If a transaction expires before OTMA enqueues the transaction, OTMA discards the transaction and returns a NAK message to the client.
- For transactions that are not MSC, Fast Path, or conversational, when an IMS application program issues a GU call to retrieve a transaction from the input queue
  If a transaction expires before an IMS application program retrieves the transaction, OTMA discards the transaction, issues pseudo abend 0243, and issues either message DFS555I or, if IMS is running in a shared-queues environment, the back-end system issues message DFS2224I.

**Important:** The purpose of the IMS Transaction Expiration SPE is to issue message DFS3688I including tpipe and tmember information instead of DFS555I or DFS2224I when the transaction expiration time is reached at GU time. See Example 4-11.

**Example 4-11** Message DFS3688I sent for expired OTMA message at GU time

DFS3688I Transaction aaaaaaaa expired: EXPRTIME=nnnnnn, ELAPSE=ssssss Tmember xxxxx Tpipe xxxx

**APARs:** This enhancement is retrofitted in IMS 11 and IMS 10 with APAR PM05984 and APAR PM05985 respectively.

Expired non-OTMA messages already receive the message DFS3688I since APAR PK86426/UK47070 for IMS 11.

For details about how to use the transaction expiration support, see *IMS Version 12 Communications and Connections, SC19-3012.*
WebSphere MQ support
For information about how the message expiry facility is implemented in WebSphere MQ, see the paper WebSphere MQ for z/OS and IMS Transaction Expiration in the Techdocs Technical Sales Library at:

http://www.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/TD105559

Important: In WebSphere MQ, the Message Expiry facility provides a new service parameter that indicates whether WebSphere MQ will take advantage of the OTMA message expiry interface. This new service parameter SERVICE = 0000000001 specified in the ZPARM CSQ6SYSP or set by using the SET SYSTEM SERVICE command.

With this service parameter, WebSphere MQ can tolerate the receipt of an OTMA NACK FOR_TRANS_EXPIRED response when WebSphere MQ passes a message by using OTMA to IMS.

A unique expiration time can also be set at the MQ message level. In this case, the desired expiration interval is passed by the application to MQ using the MQMD.Expiry field and the service parameter should be set. The time is expressed in tenths of a second and can be thought of as a time to live (TTL) for the message.

In Figure 4-9, from the remote application perspective, the MQPUT application is unaware of an expiry unless it specifies a Report option. This option can entail the following actions:

- Include the generation of an expiry report, which is sent to the specified reply-to queue.
- Pass the remaining expiry interval from a request message to a response message.
- Discard the expired message.

![Figure 4-9  Example of expiration time at MQ message level](image)

An attempt to perform an MQGET of a message after the expiration time has expired results in the MQ message being removed from the queue and the execution of expiry processing. As an MQ message flows between queue managers in an MQ network, the “time to live” expiration field is decremented even while the message waits on a transmission queue for movement between queue managers. MQ expiry processing is driven by the Report options specified by the application in the MQMD header of the message. The putting application can request a report on expiration, and will get a message sent to the reply to queues with the expiration information.

If the service parameter for OTMA transaction expiration is set, then WebSphere MQ also looks to see if an MQ expiry time has been defined for the message. If an MQ expiry time value exists, then WebSphere MQ calculates the residual value (MQ expiry time minus the time the message has already spent in MQ) and uses that value when building the OTMA interface. If no MQ expiry value has been set, then WebSphere MQ does not send a user-specified value to IMS.
Database Recovery Control enhancements

This chapter explores the main Database Recovery Control (DBRC) enhancements that are included in IMS 12. It provides background information to help familiarize or remind you about key aspects of DBRC and its components.

The chapter includes a review the new DBRC functions that you can use to add valuable user information to the recovery control (RECON) data set records, to take advantage of greater flexibility with user written skeletal job control language (JCL) with new keys, to perform a deeper cleanup when removing obsolete information, and to better control change accumulation (CA) records.

In addition, it explains the DBRC commands and functions that have changed to improve serviceability. They now eliminate the need to type unnecessary information, and provide improved information from the some of the LIST commands.

Finally, this chapter highlights DBRC migration and coexistence.

It includes the following sections:
- DBRC background information
- New features for managing RECON information
- Changes on RECON
- DBRC with IMS 12
5.1 DBRC background information

The IMS DBRC facility manages information that is required for system and database integrity, database recovery, and databases access by various IMS subsystems. As an integral part of the system, DBRC manages information about many items and stores this information in a set of Virtual Storage Access Method (VSAM) data sets that are collectively called the RECON data set.

More specifically, DBRC offers the following support:

- It helps you ensure IMS system and database integrity by recording and managing information associated with the logging process.
- It assists IMS in the restart process by notifying IMS of which logs to use for restart.
- It assists IMS to allow or prevent access to databases in data-sharing environments by recording and managing database authorization information.
- It facilitates database and log recovery.
- It supports extended recovery facility (XRF) by identifying whether the subsystem is XRF-capable.
- It supports Remote Site Recovery (RSR) by containing the RSR complex definition in the RECON data set and providing other services associated with controlling RSR.
- It supports IMSplexes.

DBRC is not required for IMS batch jobs and for some offline utilities. However, if batch jobs and utilities that access registered databases are allowed to run without DBRC, the recoverability and integrity of the databases can be lost. Even if your configuration does not require the use of DBRC, you can simplify your recovery process by using DBRC to supervise recovery and protect your databases.

DBRC includes the RECON data sets, the DBRC utility (DSPURX00), and skeletal job control language (JCL):

- DBRC stores recovery-related information in the RECON data sets. DBRC uses two RECON data sets to increase availability and recoverability. They contain identical information. If you want to continue operations in dual mode after an error occurs on one of the two active RECON data sets, you can define a third RECON data set. DBRC does not use this spare data set unless an error occurs. The RECON data sets are critical resources for both DBRC and IMS.
- The DBRC utility (DSPURX00) is used to issue commands that build and maintain the RECON data set, add information to the RECON data set, and generate jobs for utilities.
- Skeletal JCL are input models or templates stored as members in a partitioned data set (PDS) for generating input for some of the recovery utilities. DBRC uses the skeletal JCL, information from the RECON data set, and instructions from a GENJCL command to generate the JCL and control statements that are needed to run some of the recovery utilities.

The RECON data set contains many types of records. Certain records, such as header records, exist primarily to control processing of the RECON data set. Other records exist to define the various data sets used in the recovery of database data sets (DBDSs). Still other records exist to record events related to the use of DBDSs.
Figure 5-1 shows the major relationship among RECON record types.

### RECON considerations

Allocate the RECON data sets with different amounts of space so that if one becomes full, the system can continue using the other RECON data set while you provide a replacement. Allocate secondary extents for the RECON data set when you define space for it.

Back up the RECON data sets frequently. They are a critical resource. Always make backup copies of the RECON data set after performing any RECON record maintenance, such as registering databases and adding or deleting CA groups.

You need to reorganize the RECON data sets periodically. Many of the record keys in the RECON data set include the date and time. DBRC recording of IMS log and database activity can cause control interval (CI) and CA splits, which can degrade performance. In addition, deleting unnecessary records might not prevent the RECON data set from filling up because VSAM does not always reuse the space freed.

### 5.2 New features for managing RECON information

IMS has always featured DBRC as an integral piece of the whole software and relies on the information stored in the RECON data sets to perform several important activities. The following features are new in terms of managing RECON information:

- **New user information**

  With IMS 12, you can add valuable user information to the RECON records related to some of the most important processes, such as the database image copy, CA, reorganization, and recovery processes, in the database administration role.
GENJCL new user keys
IMS 12 adds more possibilities to reflect the system configuration of your installation by increasing the number of user keys in the DBRC skeletal JCL from 32 to 64. This number ultimately provides greater flexibility with user-written skeletal JCL.

New CLEANUP command parameters
IMS 12 improves the CLEANUP.RECON command by adding the possibility to delete CA execution records that are old or expired. Thus, with IMS 12, the DELETE.LOG INACTIVE or TOTIME command can be processed when the LOGALL record does not exist.

New change accumulation retention period
IMS 12 introduces the retention period keyword into the CA group type of record. The RECOVPD optional keyword is added to INIT.CAGRP or CHANGE.CAGRP to control DBRC keeping of CA execution records. When GRPMAX is exceeded, CA execution record is kept if the RECOVPD value is not exceeded.

5.2.1 New user information

With IMS 12, to allow clients to add their own control information into some of the important DBRC RECON records related to the database administration role you can add user data into IC, CA, RECOV, and REORG RECON records by using the UDATA optional keyword. The UDATA("string") is an optional keyword you use to specify up to 80 bytes of information about the identified record. You can use the variable field of this keyword to describe how the data set was created. The string value must be enclosed in single quotation marks (').

The following DBRC commands can use the UDATA keyword:
- The CHANGE.CA and NOTIFY.CA commands modify information in a CA record. Figure 5-2 shows the new syntax of the CHANGE.CA command.

![Figure 5-2 The CHANGE.CA command syntax](image-url)
The **CHANGE.IC** and **NOTIFY.IC** commands modify information contained in an image copy record. Figure 5-3 shows the new syntax of the **NOTIFY.IC** command. For the syntax of all DBRC commands, see *IMS Version 12 Commands, Volume 3: IMS Component and z/OS Commands*, SC19-3011.

![Figure 5-3   The NOTIFY.IC command syntax](image)

- The **NOTIFY.RECOV** command adds information about the recovery of the database.
- The **NOTIFY.REORG** command adds a record about the reorganization of the database.

### Using the UDATA keyword

In Example 5-1, the CA record that is identified by the GRPNAME and RECTIME keywords is being changed and valuable user information is included in the **UDATA** parameter.

**Example 5-1   The CHANGE.CA command with the UDATA parameter**

```plaintext
IMS VERSION 12 RELEASE 1 DATA BASE RECOVERY CONTROL       PAGE 0001
CHANGE.CA GRPNAME(GRPIVP) -
RECTIME(11102082132593200) -
CADSN('IMSVS.IVPDB1.CA0011') -
UDATA('LAST CA EXECUTED BEFORE CHANGING THE CPU BOX') -
DSP0203I  COMMAND COMPLETED WITH CONDITION CODE 00
DSP0220I  COMMAND COMPLETION TIME 11.208 18:31:21.487046
```
In Example 5-2, DBRC is notified of the successful completion of an image copy for the database description (DBD) specified. The UDATA optional keyword is used to clearly identify that this IC is the first using a new IMS version.

Example 5-2  The NOTIFY.IC command with the UDATA keyword

IMS VERSION 12 RELEASE 1 DATA BASE RECOVERY CONTROL          PAGE 0001
NOTIFY.IC   DBD(IVPDB1) DDN(DFSIVD1) -
RUNTIME(10049211228460952) -
ICDSN(IMS12B.IVPDB1.DFSIVD1.IC102) -
UDATA('FIRST CYCLE USING IMS NEW VERSION')
DSP0203I  COMMAND COMPLETED WITH CONDITION CODE 00
DSP0220I  COMMAND COMPLETION TIME 11.208 15:11:24.404791

In Example 5-3, the LIST.DBS command shows how the UDATA is displayed in a report.

Example 5-3   LIST.DBS showing UDATA

IMS VERSION 12 RELEASE 1 DATA BASE RECOVERY CONTROL          PAGE 0001
LIST.DBDS  DBD(IVPDB1)                      PAGE 0002
11.208 15:06:22.423389                      LISTING OF RECON
-------------------------------------------------------------------------------
DBDS
  DSN=IMS12B.DFSIVD1                             TYPE=IMS
  DBD=IVPDB1 DDN=DFSIVD1 DSID=001 DBORG=HIDAM DSORG=OSAM
  CAGRP=**NULL** GENMAX=3 IC AVAIL=2 IC USED=2 DSSN=00000001
  REUSE RECOVPD=0
  DEFLTJCL=**NULL** ICICL=ICICL OICICL=OICICL RECOVJCL=RECOVJCL
  RECVJCL=ICRCVJCL
  FLAGS:
    IC NEEDED =OFF
    RECOV NEEDED =OFF
    RECEIVE NEEDED =OFF
    EEQE COUNT =0
-------------------------------------------------------------------------------
ALLOC
  ALLOC  =10.053 19:07:28.472125  ALLOC LRID =0000000000000000
  DSSN=0000000001 USID=0000000002 START = 10.053 14:51:54.800000
IMAGE
  RUN    = 10.049 21:12:28.460952  RECORD COUNT =0
  STOP   = 00.000 00:00:00.000000
  USERDATA= FIRST CYCLE USING IMS NEW VERSION

Considerations for the UDATA keyword

User data is always listed when the RECON record is listed. User data is available by using the DBRC API.

The DSPAPQCA, DSPAPQIC, DSPAPQRV, and DSPAPQRR DSECTS are changed for the UDATA enhancement. Therefore, reassemble any of user-written programs that use it.
5.2.2 GENJCL new user keys

DBRC uses, along with other elements, PDS members as templates or skeletal JCL to correctly run recovery utilities. Usually, you modify the skeletal JCL to reflect the system configuration of your installation.

When you issue a GENJCL command, the command uses a skeletal JCL execution member, which contains symbolic keywords. You can define your own symbolic keywords and use the symbolic keywords that already exist. DBRC substitutes current information for the symbolic keywords.

IMS 12 increases the number of user keys in skeletal JCL from 32 to 64, keeping the same conventions and restrictions applied to earlier versions. In addition, the existing %DBTYPE key can be used when selecting DBDS allocation.

The following commands are affected:

- **GENJCL.ARCHIVE**: Log Archive utility
- **GENJCL.CA**: Database CA utility
- **GENJCL.CLOSE**: Log Recovery utility
- **GENJCL.IC**: Database Image Copy
- **GENJCL.OIC**: Online Database Image Copy utility
- **GENJCL.RECEIVE**: Database Recovery utility
- **GENJCL.RECOV**: Database Recovery utility
- **GENJCL.USER**: User-defined output
Figure 5-4 shows the new syntax of the `GENJCL.IC` command. For the syntax of all DBRC commands, see *IMS Version 12 Commands, Volume 3: IMS Component and z/OS Commands*, SC19-3011.

The keyword `USERKEYS (%key1, 'value' | %key2)` is an optional keyword that you use to set the value of keywords you have defined, where:

- `%key1` This is a user-defined keyword that is assigned a value. The maximum length of the keyword is 8 characters, including the percent sign (%). The first character after the % must be alphabetic (A–Z). The remaining characters must be alphanumeric (A–Z, 0–9).

- `'value'` This is the value assigned to the user-defined keyword when it is encountered. The value can be any character string enclosed in single quotation marks. The maximum length of the value is 132 characters (excluding the quotation marks). If the value contains a quotation
mark, use two single quotation marks. The value can be a null string ("."). If the value is a time stamp, it can be 0.

%key2  This is any simple keyword that was previously assigned a value, including DBRC-defined and user-defined keywords.

Up to 64 keywords can be specified.

**Using the USERKEYS keyword**

In Example 5-4, the USERKEYS keyword is used to assign a JOB name to the Image Copy process that adheres to the installation standards.

**Example 5-4  The GENJCL.IC command with the USERKEYS keyword**

```plaintext
IMS VERSION 12 RELEASE 1 DATA BASE RECOVERY CONTROL
GENJCL.IC DBD(IVPDB1) DDN(DFSIVD1) -
    JOB(JOBJCL2)  JCLOUT(JCLOUT)  JCLPDS(JCLPDS) -
    USERKEYS((%IVPJOB,'IV2F104J') -
        (%IVPCIC,'VANAERS'))
DSP0203I  COMMAND COMPLETED WITH CONDITION CODE 00
DSP0220I  COMMAND COMPLETION TIME 11.208 21:02:15.934457

//%IVPJOB   JOB ACTINFO1,
//  'PGMRNAME',
//  CLASS=A,
//  MSGCLASS=H,MSGLEVEL=(1,1),
//  NOTIFY=%IVPCIC,
//  REGION=128M
//*
//IV2F104J  JOB ACTINFO1,
//  'PGMRNAME',
//  CLASS=A,
//  MSGCLASS=H,MSGLEVEL=(1,1),
//  NOTIFY=VANAERS,
//  REGION=128M
```

**Considerations for the user keys**

The %DBTYPE key is similar to %SELECT DBDS in previous IMS versions.

### 5.2.3  New CLEANUP command parameters

IMS 11 introduced the **CLEANUP.RECON** command to delete old or expired recovery-related information from the RECON data set, including image copy, allocation, reorganization, and recovery records and log information.

IMS 12 adds the ability to delete CA execution records information by the new CAGRANGE, CAONLY, and LASTCA keywords. In addition, IMS 12 issues the message DSP0115I instead of setting a fatal return code when a **DELETE.LOG INACTIVE** or **TOTIME** command was issued and a LOGALL record did not exist for the inactive log.

The following commands are affected:

- **CLEANUP.RECON**  Deletes obsolete recovery-related CA execution and log records.
- **DELETE.LOG**  Deletes LOG records.
Figure 5-5 shows the new syntax of the **CLEANUP.RECON** command. For the syntax of other DBRC commands in *IMS Version 12 Commands, Volume 3: IMS Component and z/OS Commands*, SC19-3011.

![Figure 5-5 The CLEANUP.RECON command syntax](image)

The **CLEANUP.RECON** command has the following options:

**CAGRANGE(firstcag,lastcag | firstcag ,lastcag)**

This is an optional keyword that specifies that CA data sets for a range of CA groups are to have cleanup done. If this keyword is not specified, all CA groups are processed. Use this keyword to clean up CA data sets for multiple CA groups or to resume cleanup after a **CLEANUP.RECON CAGRANGE** command is stopped.

- **firstcag,lastcag** These are the upper and lower limits of the range of CA groups desired. These parameters do not need to match the name of a CA group in the RECON data set. They are used as the beginning and ending arguments in an alphabetic search.

- **firstcag** This is the name of the CA group from which RECON cleanup begins processing the CA execution data sets. This CA group and all subsequent CA groups are processed until the last group in the RECON data set is reached or until the CA group specified as the lastcag value is reached.

- **lastcag** This is the name of the CA group up to which RECON cleanup continues to process CA groups. RECON cleanup starts with the first CA group defined in the RECON data set and stops after this CA group is processed. If lastcag is specified, it must be preceded by a comma (,).

**CAONLY**

This is an optional keyword you use to specify that you want to delete only CA execution records. (No database recovery-related information or log information is processed.) However, if the DBONLY keyword is also included in the command, database recovery-related information is also processed, and only the log information remain unprocessed.

**LASTCA**

This is an optional keyword you use to specify that the last CA execution record for a CA group can be deleted as part of the RECON cleanup process if it meets the criteria for deletion. If this keyword is not specified, the last CA execution record for a CA group is retained.
Using the CLEANUP and DELETE commands

Example 5-5 shows the output from a CLEANUP.RECON command submitted with LASTIC and LASTCA optional keywords. These keywords mean that IC and CA records can be deleted as part of the RECON cleanup process if they meet the criteria for deletion, including the retention period specified.

Example 5-5   The CLEANUP.RECON command

For more examples of the CLEANUP.RECON command, see IMS Version 12 Commands, Volume 3: IMS Component and z/OS Commands, SC19-3011.

Example 5-6 shows the DELETE.LOG command when specifying that the records of all inactive recovery log data sets (RLDSs) and system log data sets (SLDSs) that have a start time older than the time specified with the TOTIME keyword are to be deleted.

Example 5-6   The DELETE.LOG command

For more examples of the CLEANUP.RECON command, see IMS Version 12 Commands, Volume 3: IMS Component and z/OS Commands, SC19-3011.

Example 5-6 shows the DELETE.LOG command when specifying that the records of all inactive recovery log data sets (RLDSs) and system log data sets (SLDSs) that have a start time older than the time specified with the TOTIME keyword are to be deleted.

Example 5-6   The DELETE.LOG command
Considerations for the CLEANUP command

After you delete amounts of data, you might want to reorganize your RECONs to ensure that the freed space can be reused. You must issue the CLEANUP.RECON command through a batch DBRC job or the DBRC API. This command is not available from IMS through an /RMx command.

Evaluate the usage of the CLEANUP.RECON command first on a copy of the RECONs, to avoid unintended deletions and to estimate the amount of time to process the command.

The last CA execution record for a CA group is only deleted whenever specifically requested.

5.2.4 New change accumulation retention period

Earlier versions of IMS use the GRPMAX parameter to control how many versions of CA execution should be kept in the RECON data sets. When the number of times you run the CA utility for the specified group exceeds the GRPMAX value, the record with the earliest CA stop time for the group is deleted for CA groups.

To improve the control of relevant information stored in the CA group record, IMS 12 provides the RECOVPD optional keyword that can be used to set the recovery period for a specified CA group records. The recovery period is the amount of time before the current date for which DBRC maintains CA information in the RECON data set.

The CHANGE.CAGRP and INIT.CAGRP commands are affected, which you use to modify information contained in a specified CA group record. Figure 5-6 shows the new syntax of CHANGE.CAGRP command. For the syntax of other DBRC commands, see IMS Version 12 Commands, Volume 3: IMS Component and z/OS Commands, SC19-3011.

![Figure 5-6 Syntax of the CHANGE.CAGRP command](image)

The CHANGE.CAGRP command has the RECOVPD(0 | value) option. You use this optional keyword to modify the recovery period for a specified CA group. The recovery period is the amount of time before the current date for which DBRC maintains CA information in the RECON data set. For example, if the recovery period of a CA group is 14 days, DBRC maintains sufficient CA execution records for at least 14 days.

To determine whether the CA execution record falls within the recovery period, subtract the RECOVPD value from the current time. Any CA execution records with stop times that are newer than the calculated time is to be kept in the RECON data set.
For value, specify a decimal number in the range 0–999 that represents the number of days the CA execution records are to be kept in the RECON data set. If you specify 0 (the default), there is no recovery period.

**Using the RECOVPD keyword**

In Example 5-7, the retention period for the CA data sets for CA group GRPIVP is being changed to 7 days. This change accommodates new business requirements because the user started doing more than one CA process per day.

**Example 5-7   The CHANGE.CAGRP command with the RECOVPD keyword**

<table>
<thead>
<tr>
<th>IMS VERSION 12 RELEASE 1 DATA BASE RECOVERY CONTROL</th>
<th>PAGE 0001</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHANGE.CAGRP GRPNAME(GRPIVP) RECOVPD(7)</td>
<td></td>
</tr>
<tr>
<td>DSP0203I COMMAND COMPLETED WITH CONDITION CODE 00</td>
<td></td>
</tr>
<tr>
<td>DSP02201 COMMAND COMPLETION TIME 11.208 17:05:49.648625</td>
<td></td>
</tr>
<tr>
<td>IMS VERSION 12 RELEASE 1 DATA BASE RECOVERY CONTROL</td>
<td>PAGE 0002</td>
</tr>
<tr>
<td>LIST.CAGRP GRPNAME(GRPIVP)</td>
<td></td>
</tr>
<tr>
<td>11.208 17:05:49.408264</td>
<td>LISTING OF RECON</td>
</tr>
</tbody>
</table>

```
CAGRP
GRPNAME=GRPIVP GRPMAX=7 CA AVAIL=0 CA USED=0
NOREUSE CAJCL=CAJCL DEFLTJCL=**NULL** RECOVPD=7
#MEMBERS=1 -DBD- -DDN- IVPDB1 DFSIVD1
```

DSP0180I NUMBER OF RECORDS LISTED IS 1
DSP0203I COMMAND COMPLETED WITH CONDITION CODE 00

**Considerations for the RECOVPD keyword**

The recovery period value is a decimal number in the range 0–999 that represents the quantity of days the CA execution records are kept in the RECON data set. If you specify 0 (the default), there is no recovery period. If you specify 7, DBRC maintains sufficient CA execution records for at least 7 days based on the stop time of the CA execution record.

If you issue the `CHANGE.CAGRP` command and specify the GRPMAX and RECOVPD values that are less than the existing values, any used CA data sets with stop times that are beyond the recovery period are deleted. These data sets remain deleted until the number of remaining CA data sets equals the specified GRPMAX value.

If you issue the `DELETE.CA` command, any specified CA data set record is deleted regardless of the RECOVPD or GRPMAX values.

If the GRPMAX limit is reached, but the RECOVPD for the oldest CA record has not expired, DBRC issues an informational message (DSP1232I) and does not discard the record. If the DSP1232I message appears frequently, you might need to tune the GRPMAX or RECOVPD values by using the `CHANGE.CAGRP` command.

**Attention:** If the GRPMAX value is lowered by using the `CHANGE.CAGRP` command, the GRPMAX value is recorded regardless of whether the oldest CA data sets can be deleted because they are within the recovery period.
5.3 Changes on RECON

IMS uses RECON data set information to perform some of its internal activities and to provide enough information so that you can make informed decisions when issues arise. The following changes have occurred regarding managing RECON information:

- **CATDS specification**
  IMS 12 makes the **VOLLIST** parameter optional when creating or changing the Image Copy and CA records because the CATDS is in effect for the control record. However, IMS 12 enforces the VOLLIST specification for the **INIT.IC** command when the NOCATDS is in effect.

- **LIST command**
  IMS 12 modifies some of the output listing from **LIST** commands to tailor better from the user requirement perspective. The online **RMLIST** command allows an output larger than 32K. An optional suppression of recovery information was introduced on the **LIST.DB** and **LIST.DBDS** commands by the NORCVINF keyword. The use of full precision timestamps is added to the **LIST.HISTORY** command, and the **LIST.RECON** command now reports the number of registered databases at DBRC.

The following sections provide more details.

5.3.1 **CATDS specification**

Earlier versions of IMS required the VOLLIST keyword to specify the serial numbers of the volumes on which the image copy or the CA data set being defined are to reside. Even the RECON data set was initialized with the CATDS parameter. (CATDS specifies that these data sets are cataloged or SMS-managed.) Most of the time, depending on your installation standards, the FILESEQ and UNIT keywords must also be specified.

To reduce the unnecessary information required to create and manipulate Image copy and CA records, IMS 12 makes the VOLLIST keyword parameter optional if the RECON status record indicates that the data sets are to be treated as cataloged. (CATDS is in effect.)

The following commands are affected:

**INIT.CA** / **NOTIFY.CA** This command modifies information in a CA record.

**INIT.IC** / **NOTIFY.IC** This command modifies information in an image copy record.
Figure 5-7 shows the new syntax of the \texttt{INIT.RECON} command. For the syntax of other DBRC commands, see \textit{IMS Version 12 Commands, Volume 3: IMS Component and z/OS Commands}, SC19-3011.

The \texttt{INIT.RECON} command has the following mutually exclusive, optional keywords that you use to indicate whether image copy, CA, and log data sets are cataloged:

\textbf{NOCATDS}  
This keyword specifies that these data sets, regardless of whether they are cataloged, are not to be treated as cataloged. DBRC verifies that the volume serial and file sequence numbers appearing in the job file control block are the same as the information recorded in the RECON data set.

\textbf{CATDS}  
This keyword specifies that these data sets are cataloged or SMS-managed. If the data set is allocated by the catalog and the CATDS option is used, DBRC bypasses volume serial and file sequence verification for the data set. For the CATDS option to be effective, the data set must be cataloged, and volume serial information for the data set must be omitted from the JCL.

\begin{itemize}
  \item If the data set is cataloged, CATDS is specified, and volume serial information is included in the JCL. DBRC ignores CATDS and allocates the data set by the JCL. Normal volume serial and file sequence checking occurs.
  \item If the data set is not cataloged, CATDS is not effective, and DBRC allocates the data set by the JCL, with volume serial and file sequence checking. If log data sets are SMS-managed, select the CATDS option and remove the \texttt{\%LOGVOLS} keyword from skeletal JCL member CAJCL.
\end{itemize}
Using the CATDS specification in effect
In Example 5-8, a record is created in the RECON data set that defines an image copy data set for the specified database. VOLLIST is not needed because CATDS is in effect.

Example 5-8  The NOTIFY.IC command with CATDS in effect

<table>
<thead>
<tr>
<th>NOTIFY.IC</th>
<th>DBD(IVPDB1) DDN(DFSIVD1) -</th>
</tr>
</thead>
<tbody>
<tr>
<td>RUNTIME(1105221128460952) -</td>
<td></td>
</tr>
<tr>
<td>ICDSN(IMS12B.IVPDB1.DFSIVD1.IC105) -</td>
<td></td>
</tr>
<tr>
<td>UDATA('EXTRA IC DUE TO MASSIVE UPDATE')</td>
<td></td>
</tr>
</tbody>
</table>

DSP0203I  COMMAND COMPLETED WITH CONDITION CODE 00
DSP0220I  COMMAND COMPLETION TIME 11.208 17:15:21.588069

Considerations for the CATDS specification
The CHANGE.RECON and INIT.RECON commands update the RECON data set header record accordingly if you specify CATDS or NOCATDS.

IMS 12 does not allow usage of the INIT.IC command without VOLLIST when NOCATDS is in effect, which is different from earlier versions. In this sense, this command is now more restrictive.

5.3.2 LIST command
The most common usage of the LIST command is to produce a formatted printout of all or selected information contained in a RECON data set. Normally, you use the information printed to make an analysis and then take specific action regarding the DBRC controlled resources.

IMS 12 allows an output larger than 32 K for the /RMLIST online command, but only when the command is entered through the Operations Manager (OM) API. The output size is restricted by the DBRC private storage available for buffering the output message or OM limitations.

IMS 12 provides the NORCVINF keyword for the LIST.DB and LIST.DBDS commands, which suppresses recovery-related information. That is, ALLOC, IC, RECOV and REORG records are not listed, which reduces command output.

IMS 12 adds the full precision timestamps and more information about the HALDB type of databases such as active DBDSs, DDNames of inactive DBDSs, and current reorganization number for partition in the LIST.HISTORY command output.

In the LIST.RECON command output, IMS 12 includes the amount of registered databases, facilitating your control over the DBRC limit of 32,767 registered databases.
Figure 5-8 shows the new syntax of **LIST.DB** command. For the syntax of other DBRC commands, see *IMS Version 12 Commands, Volume 3: IMS Component and z/OS Commands*, SC19-3011.

![The LIST.DB command syntax](image)

The **LIST.DB** command has the NORCVINF option. This option suppresses recovery-related records (ALLOC, IC, RECOV, and REORG) for those DBDSs or areas in the RECON data set that are associated with the specified database. If the **LIST.DB** command output is truncated and message DSP0057I is returned, you can specify the NORCVINF keyword to minimize the size of the output.

**Using the LIST command**

In Example 5-9, the **LIST.DB** command is used with the NORCVINF information.

```
IMS VERSION 12 RELEASE 1 DATA BASE RECOVERY CONTROL PAGE 0001
LIST.DB DBD(IVPDB1) DBDS NORCVINF
11.208 19:04:12.800733              LISTING OF RECON                  PAGE 0002
-------------------------------------------------------------------------------
DB
DBD=IVPDB1                   IRLMID=**NULL**        DMB#=1       TYPE=IMS
SHARE LEVEL=3               GSGNAME=**NULL**     USID=0000000002
AUTHORIZED USID=0000000002  RECEIVE USID=0000000002 HARD USID=0000000002
RECEIVE NEEDED USID=0000000000
DBRCVGRP=**NULL**
FLAGS:                             COUNTERS:
   BACKOUT NEEDED        =OFF         RECOVERY NEEDED COUNT   =0
   READ ONLY             =OFF         IMAGE COPY NEEDED COUNT =1
   PROHIBIT AUTHORIZATION=OFF         AUTHORIZED SUBSYSTEMS   =0
   RECOVERABLE           =YES         HELD AUTHORIZATION STATE=0
   TRACKING SUSPENDED    =NO          EEQE COUNT              =0
   OFR REQUIRED          =NO          RECEIVE REQUIRED COUNT =0
   REORG INTENT          =NO
   QUIESCE IN PROGRESS   =NO
   QUIESCE HELD          =NO
-------------------------------------------------------------------------------
11.208 19:04:12.800733              LISTING OF RECON                  PAGE 0003
-------------------------------------------------------------------------------
```

**Example 5-9** The **LIST.DB** command with the NORCVINF keyword
Example 5-10 shows the output of the LIST.HISTORY command, which includes full precision timestamps.

Example 5-10  Output of the LIST.HISTORY command

```
IMS VERSION 12 RELEASE 1 DATA BASE RECOVERY CONTROL        PAGE 0001
LIST.HISTORY DBD(IVPDB1)                                      PAGE 0002
11.208 19:16:28.594191              LISTING OF RECON                  PAGE 0003
```

```
DB
BB=IVPDB1 IRLMID=**NULL** TYPE=IMS
SHARE LEVEL=3 USID=00000002
AUTHORIZED USID=0000000002 RECEIVE USID=0000000002 HARD USID=0000000002
RECEIVE NEEDED USID=0000000000
DBRCVGRP=**NULL**
FLAGS:                                 COUNTERS:
 BACKOUT NEEDED =OFF RECOVERY NEEDED COUNT =0
 READ ONLY =OFF IMAGE COPY NEEDED COUNT =0
 PROHIBIT AUTHORIZATION=OFF AUTHORIZED SUBSYSTEMS =0
 RECOVERABLE =YES HELD AUTHORIZATION STATE=0
 EEQE COUNT =0
 TRACKING SUSPENDED =NO RECEIVE REQUIRED COUNT =0
 OFR REQUIRED =NO
 REORG INTENT =NO
 QUIESCE IN PROGRESS =NO
 QUIESCE HELD =NO
```

```
11.208 19:16:28.594191              LISTING OF RECON                  PAGE 0004
```

```
DBDS
DSN=IMS12B.DFSIVD1 TYPE=IMS
DBD=IVPDB1 DSN=DFSIVD1 DSID=0010 DBORG=HIDAM DSORG=OSAM
CAGRP=**NULL** GENMAX=3 IC AVAIL=2 IC USED=0 DSSN=00000001
REUSE RECOVPD=0
DEFLTJCL=**NULL** ICJCL=ICJCL OICJCL=OICJCL RECOVJCL=RECOVJCL
RECVJCL=ICRCVJCL
FLAGS:                                 COUNTERS:
 IC NEEDED =OFF EEQE COUNT =0
 RECOV NEEDED =OFF
 RECEIVE NEEDED =OFF
```

```
11.208 19:16:28.594191              LISTING OF RECON                  PAGE 0005
```

```
DBDS
DSN=IMS12B.DFSIVD1 TYPE=IMS
DBD=IVPDB1 DSN=DFSIVD1 DSID=0010 DBORG=HIDAM DSORG=OSAM
CAGRP=**NULL** GENMAX=3 IC AVAIL=2 IC USED=0 DSSN=00000001
REUSE RECOVPD=0
DEFLTJCL=**NULL** ICJCL=ICJCL OICJCL=OICJCL RECOVJCL=RECOVJCL
RECVJCL=ICRCVJCL
FLAGS:                                 COUNTERS:
 IC NEEDED =OFF EEQE COUNT =0
 RECOV NEEDED =OFF
 RECEIVE NEEDED =OFF
```

154  IBM IMS Version 12 Technical Overview
Example 5-11 shows the LIST.RECON STATUS command to verify how many databases are already registered into the RECON data sets.

**Example 5-11  The LIST.RECON STATUS command**

```
DSP1123I  IMSR4C5C DBRC REGISTERED WITH IMSPLEX IM12X USING EXIT IMS VERSION 12 RELEASE 1 DATA BASE RECOVERY CONTROL PAGE 0002
LIST.RECON STATUS
11.208 12:45:39.109351 LISTING OF RECON PAGE 0003

RECON
RECOVERY CONTROL DATA SET, IMS V12R1
DMB#=23 INIT TOKEN=11192F074559F5
FORCER LOG DSN CHECK=CHECK44 STARTNEW=NO
TAPE UNIT= DASD UNIT=SYSALLDA TRACEOFF SSID=I12A
LIST DLOG=YES CA/IC/LOG DATA SETS CATALOGED=YES
MINIMUM VERSION = 10.1 CROSS DBRC SERVICE LEVEL ID= 00001
REORG NUMBER VERIFICATION=NO
LOG RETENTION PERIOD=00.001 00:00:00.0
COMMAND AUTH=NONE HLQ=**NULL**
RCNQUAL=**NULL**
ACCESS=SERIAL LIST=STATIC
SIZALERT DSNUM=15 VOLNUM=16 PERCENT= 95
LOGALERT DSNUM=3 VOLNUM=16

NUMBER OF REGISTERED DATABASES = 8
```

DSP0180I  NUMBER OF RECORDS LISTED IS 1
DSP0203I  COMMAND COMPLETED WITH CONDITION CODE 00
DSP0220I  COMMAND COMPLETION TIME 11.208 12:45:39.359739

**Considerations for the /RMLIST command**

When using the /RMLIST command, a performance impact might result from reading and reserving the RECONs for an extended amount of information.
5.4 DBRC with IMS 12

IMS 12 DBRC enhancements provide modifications to its internal records and functionality. Therefore, you must keep in mind certain considerations when upgrading from earlier versions. This section explores coexistence and migration from the DBRC point of view. For more and complete information about IMS 12 coexistence and migration, see Chapter 8, “Installation and migration considerations” on page 279.

5.4.1 Coexistence

IMS 12 can coexist with IMS 10 and IMS 11, so that existing applications and data can be used without change. However, coexistence considerations apply to each of the IMS versions.

Coexistence with IMS 10
An IMS 10 system can coexist with IMS 12 in the following situations:

- The RECON has been upgraded to IMS 12.
- APAR PM05243 (PTF UK62970) is applied.
- MINVERS in the RECON data set has not been set higher than 10.1.

Therefore, database quiesce is not available.

Coexistence with IMS 11
An IMS 11 system can coexist with IMS 12 in the following situations:

- The RECON has been upgraded to IMS 12.
- APAR PM05244 (PTF UK62971) is applied.
- MINVERS in the RECON data set has been set to 11.1.

Therefore, cross-system coupling facility (XCF) use by APPC synchronous conversations and OTMA CM1 (send-then-commit) are not available.

Considerations
The MINVERS level must be set to the lowest level of IMS that uses or shares the RECON data sets.

DBRC applications compiled with Version 1.0 DSPAPI macros work without modification or reassembly with Version 2.0 of the DBRC API. However, these applications cannot use any of the newer functions or options that are supported in Version 2.0 macros, which are available only with IMS 10 and later.

5.4.2 Migration

IMS 12 requires you to upgrade the RECON data set before you start a control region. A new CHECKUP keyword was introduced in the CHANGE.RECON UPGRADE command to inform you in advance if the RECON data set from earlier version is in a state that allows upgrade. The benefit is that you can react and take the appropriate actions before effectively doing the upgrade, and thereby avoid facing unexpected conditions.

If you have to coexist with earlier versions of IMS, you must apply the DBRC migration or coexistence APARs. You are not required to change the MINVERS value to ‘12.1’ when you migrate to IMS 12. Change this value only after you verify that you do not need to coexist with an earlier version of IMS. When you do not need to fall back to an earlier version and you need to use new functions, the MINVERS value must be set to ‘12.1’.
The upgrade process begins with a quiesce close and a check for shunted I/O. If there is no shunted I/O, the records in COPY1 and COPY2 are upgraded in parallel RECON access (PRA) mode. After upgrade completes, the quiesce for the RECON data sets is released and other DBRC instances can access the upgraded RECON data sets.

Upgrade processing from IMS 11 to IMS 12 offers the following benefits:

- Reads SSYS records to check for DBRC small programming enhancements (SPEs)
- Reads all database records
- Turns on the high-order bit if it is not on and the database is not authorized
- Fails if the high-order bit is not on and the database is authorized
- Builds a DMB table record
- Updates the RECON header record
- Sets the version indicator and MINVERS value
- Updates the RECON header extension record
- Sets the version indicator
- After COPY1 is upgraded, copies it to COPY2

Upgrade processing from IMS 10 to IMS 12 offers the following benefits:

- Reads SSYS records to check for DBRC SPE
- Reads all database records
- Turns on the high-order bit if it is not on and the database is not authorized; fails if the high-order bit is not on and the database is authorized
- Builds a DMB table record
- Updates the RECON header record
- Expands a record by 44 bytes for the RECON qualifier field used by DBRC Security Override support
- Sets the version indicator and MINVERS value
- Updates the RECON header extension record
- Sets the version indicator
- Updates CA execution records
- Expands records by 16 bytes
- Updates database or area authorization records
- Expands records by 20 bytes to support future enhancements
- After COPY1 is upgraded, copies it to COPY2
Figure 5-9 shows the new syntax of **LIST.DB** command. For the syntax of other DBRC commands, see *IMS Version 12 Commands, Volume 3: IMS Component and z/OS Commands*, SC19-3011.

Figure 5-9   The CHANGE.RECON syntax

**Using the CHANGE.UPGRADE command**

Example 5-12 shows how to use the CHECKUP keyword to verify in advance if the RECON is ready to upgrade to IMS 12.

*Example 5-12   The CHANGE.RECON UPGRADE command*

<table>
<thead>
<tr>
<th>MESSAGE</th>
<th>TEXT</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSP1238I</td>
<td>RECON UPGRADE CHECKUP IS BEGINNING</td>
</tr>
<tr>
<td>DSP1239I</td>
<td>RECON UPGRADE CHECKUP COMPLETED WITH NO ERRORS FOUND</td>
</tr>
<tr>
<td>DSP0203I</td>
<td>COMMAND COMPLETED WITH CONDITION CODE 00</td>
</tr>
<tr>
<td>DSP0220I</td>
<td>COMMAND COMPLETION TIME 11.208 12:14:25.329216</td>
</tr>
<tr>
<td>DSP0211I</td>
<td>COMMAND PROCESSING COMPLETE</td>
</tr>
<tr>
<td>DSP0211I</td>
<td>HIGHEST CONDITION CODE = 00</td>
</tr>
</tbody>
</table>
Example 5-13 shows sample JCL and control cards to perform the RECON upgrade from IMS 11 to IMS 12 in a copy of the RECON. This approach helps to determine how long it will take when using the actual RECON data sets.

Example 5-13  JCL sample to perform an upgrade in a RECON Copy

```plaintext
//CLEANUP EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=* 
//SYSIN DD *
  DELETE IMS11B.RECON1.COPY PURGE
  DELETE IMS11B.RECON2.COPY PURGE
  DELETE IMS11B.RECON3.COPY PURGE
  SET MAXCC=0
/*
//COPYALOC EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=* 
//SYSIN DD *
  DEFINE CLUSTER (NAME(IMS11B.RECON1.COPY) VOL(SBOX01) CYL(7 0) -
                FOR(9999) SHR(3 3) BUFSPC(32768) NONSPANNED NOWRITECHECK) -
                DATA (NAME(IMS11B.RECON1.D.COPY) CISZ(32768) RECSZ(30 32600) -
                FSPC(30 35) KEYS(32 0)) -
                INDEX (NAME(IMS11B.RECON1.I.COPY) CISZ(2048))

  DEFINE CLUSTER (NAME(IMS11B.RECON2.COPY) VOL(SBOX01) CYL(9 0) -
                FOR(9999) SHR(3 3) BUFSPC(32768) NONSPANNED NOWRITECHECK) -
                DATA (NAME(IMS11B.RECON2.D.COPY) CISZ(32768) RECSZ(30 32600) -
                FSPC(30 35) KEYS(32 0)) -
                INDEX (NAME(IMS11B.RECON2.I.COPY) CISZ(2048))

  DEFINE CLUSTER (NAME(IMS11B.RECON3.COPY) VOL(SBOX01) CYL(9 0) -
                FOR(9999) SHR(3 3) BUFSPC(32768) NONSPANNED NOWRITECHECK) -
                DATA (NAME(IMS11B.RECON3.D.COPY) CISZ(32768) RECSZ(30 32600) -
                FSPC(30 35) KEYS(32 0)) -
                INDEX (NAME(IMS11B.RECON3.I.COPY) CISZ(2048))
/*
//COPY EXEC PGM=DSPURX00,REGION=2048K
//STEPLIB DD DISP=SHR,DSN=IMS11B.SDFSRESL
//RECON1 DD DSN=IMS11B.RECON1,DISP=SHR
//RECON2 DD DSN=IMS11B.RECON2,DISP=SHR
//BACKUP1 DD DSN=IMS11B.RECON1.COPY,DISP=SHR
//BACKUP2 DD DSN=IMS11B.RECON2.COPY,DISP=SHR
//SYSPRINT DD SYSOUT=* 
//SYSIN DD *
  BACKUP.RECON BOTH
/*
//UPGRADE EXEC PGM=DSPURX00,REGION=2048K
//STEPLIB DD DISP=SHR,DSN=IMS12Q.SDFSRESL
//SYSPRINT DD SYSOUT=* 
//RECON1 DD DSN=IMS11B.RECON1.COPY,DISP=SHR
//RECON2 DD DSN=IMS11B.RECON2.COPY,DISP=SHR
//RECON3 DD DSN=IMS11B.RECON3.COPY,DISP=SHR
//SYSIN DD *
  CHANGE.RECON UPGRADE
*/
```

Chapter 5. Database Recovery Control enhancements
Considerations for the Upgrade option
Optionally, issue the CHANGE.RECON UPGRADE CHECKUP command to ensure that your RECON data set is ready to be migrated.

Before you issue the CHANGE.RECON UPGRADE command against the production RECON data sets, upgrade a copy of the production RECON data sets to verify how long it will take to perform the upgrade. Issue the CHANGE.RECON UPGRADE command by using either the IMS 12 DBRC Recovery Control utility (DSPURX00) or the IMS 12 DBRC Command API request. IMS online Command is not allowed.

After this command successfully completes, DBRC sets the value for MINVERS (the minimum version of IMS that can sign on to DBRC) to ‘10.1’ if the value was less than ‘10.1’.

Ensure that you have two active RECON data sets (COPY1 and COPY2) and a spare data set when you upgrade the RECON data sets while other jobs are accessing them.

Upgrading from IMS 10 can increase the size of the RECONs. Evaluate whether the current size space can accommodate the IMS 12 records.

The upgrade process reads all database records to ensure that the high-order bit is on in all DMB numbers. If the high-order bit is not on (this should not occur), the bit is turned on if the database is not authorized and the DSP1235W message is displayed:

DSP1235W THE INTERNAL REPRESENTATION OF THE DMB NUMBER FOR DATABASE xxxxxxxx IS INCORRECT

The high-order bit is not turned on if the database is authorized; instead, the upgrade fails and the DSP1236E message is displayed:

DSP1236E THE INTERNAL REPRESENTATION OF THE DMB NUMBER FOR DATABASE xxxxxxxx COULD NOT BE CORRECTED BECAUSE THE DATABASE IS AUTHORIZED

Attention: After RECONs are upgraded to IMS 12, IMS 11 or IMS 10 Log Archive jobs use additional memory because versions of RECON records are kept in memory. Use REGION=0M for IMS 10 and IMS 11 archive jobs.
Connectivity enhancements

This chapter describes the changes and enhancements to IBM Information Management System (IMS) and IMS Connect to support asynchronous IMS to IMS communication, Multiple Systems Coupling (MSC), using TCP/IP and the new IMS Connect type-2 single point of control (SPOC) commands. It also documents the new recorder trace enhancements, the XML converter refresh, and the Resource Access Control Facility (RACF) return code changes.

This chapter includes the following sections:
▶ OTMA support for asynchronous IMS to IMS communications
▶ Multiple Systems Coupling using TCP/IP
▶ Other IMS Connect changes
▶ IMS Connect type-2 Single Point of Control commands
6.1 OTMA support for asynchronous IMS to IMS communications

Before IMS 12, if you wanted to route an asynchronous transaction from one IMS to another IMS over a TCP/IP network, you had to create a gateway application that received messages from one IMS using resume TPipe processing. Then you had to insert it into the other IMS using send-only processing.

The need for a customer written gateway is removed with IMS 12. You can now define a remote IMS system in IMS Connect and an Open Transaction Manager Access (OTMA) descriptor in IMS (Figure 6-1). This way, an application program can push a message to the remote system using a normal sequence of CHNG, ISRT, and PURG calls. (PURG is only valid for an EXPRESS=YES PCB).

![Figure 6-1 Asynchronous IMS to IMS](image)

6.1.1 OTMA descriptor definitions

To define remote asynchronous IMS to IMS communications, you need to define an OTMA descriptor. To define an OTMA descriptor, you can add a descriptor to the DFSYDTX member of IMS.PROCLIB or define it dynamically.

**IMS PROCLIB definitions**

You can define OTMA descriptors by updating the DFSYDXTx member of IMS.PROCLIB (Example 6-1). When you activate an OTMA descriptor this way, you must restart IMS.

*Example 6-1 IMS.PROCLIB member DFSYDTC*

```
D OTMA2CD TYPE=IMSCON TMEMBER=HWBI12C2 RMTIMSCON=HWSI12D1 RMTIMS=I12D
D OTMA2CD RMTTRAN=OTMAPRNT
```

You also need a matching entry in the other system (Example 6-2) so that the remote IMS can send replies. The reply message is not sent back to the originating transaction in the local IMS.

*Example 6-2 IMS.PROCLIB member DFSYDTD*

```
D OTMA2DC TYPE=IMSCON TMEMBER=HWBI12D4 RMTIMSCON=HWSI12C1 RMTIMS=I12C
D OTMA2DC RMTTRAN=OTMAPRNT
```
This entry gives you a destination that you can use with an alternate I/O PCB CHNG call when we want to send data from IMS C to IMS D or back from IMS D to IMS C.

**Dynamic OTMA descriptors**
It is much easier to define the OTMA descriptors dynamically using the new type-2 CREATE OTMADESC command. This approach does not require you to restart IMS. The statically defined definitions created by updating the DFSYDTCx members can be updated or deleted by using the UPDATE OTMADESC or DELETE OTMADESC commands. You can use the QUERY OTMADESC command to see what you have defined.

### 6.1.2 Modified IMS commands

IMS 12 has updated the CREATE OTMADESC, UPDATE OTMADESC, DELETE OTMADESC, and QUERY OTMADESC commands to support the new asynchronous IMS to IMS communications.

#### The CREATE OTMADESC command
IMS 12 has added options to the CREATE OTMADESC command to set the RMTIMS, RMTIMSCON USERID, and RMTTRAN parameters.

Example 6-3 defines the same OTMA descriptors created with the static definitions in Example 6-1.

**Example 6-3 Creating the first OTMA descriptor**

```sql
CREATE OTMADESC NAME(OTMA2CD) -
SET(TYPE(IMSCON) TMEMBER(HWBI12C2) -
RMTIMSCON(HWSI12D1) RMTIMS(I12D) RMTTRAN(OTMAPRNT))
```

Example 6-4 defines the same OTMA descriptors created with the static definitions in Example 6-2.

**Example 6-4 Creating the second OTMA descriptor**

```sql
CREATE OTMADESC NAME(OTMA2DC) -
SET(TYPE(IMSCON) TMEMBER(HWBI12D4) -
RMTIMSCON(HWSI12C1) RMTIMS(I12C) RMTTRAN(OTMAPRNT))
```

#### The UPDATE OTMADESC command
IMS 12 has added the same options to the UPDATE OTMADESC command as the CREATE OTMADESC command to support changing the new parameters (Example 6-5).

**Example 6-5 The Update OTMA descriptor**

```sql
UPDATE OTMADESC NAME(OTMA2DC) SET(RMTTRAN(NEWTRAN))
```

#### The QUERY OTMADESC command
IMS 12 has updated the QUERY OTMADESC command to display the new options. To see the definitions, you can use the command shown in Example 6-6.

**Example 6-6 The Query OTMA descriptor command**

```sql
QRY OTMADESC NAME(*) OPTION(WILDCARD) -
SHOW(TMEMBER, RMTIMS, RMTIMSCON, RMTTRAN) TYPE(IMSCON)
```
Example 6-7 shows the output from the `QUERY OTMASDESC` command.

**Example 6-7  Output from the QRY OTMADESC command**

<table>
<thead>
<tr>
<th>DestName</th>
<th>MbrName</th>
<th>CC Type</th>
<th>Type</th>
<th>TMember</th>
<th>RmtIMSCon</th>
<th>RmtIMS</th>
<th>RmtTran</th>
</tr>
</thead>
<tbody>
<tr>
<td>OTMA2CD</td>
<td>I12C</td>
<td>0</td>
<td>IMSCON</td>
<td>HWBI12C3</td>
<td>HWSI12D1</td>
<td>I12D</td>
<td>OTMAPRNT</td>
</tr>
<tr>
<td>OTMA2DC</td>
<td>I12D</td>
<td>0</td>
<td>IMSCON</td>
<td>HWBI12D4</td>
<td>HWSI12C1</td>
<td>I12C</td>
<td>OTMAPRNT</td>
</tr>
</tbody>
</table>

Notice that, in these examples, we used **OPTION(WILDCARD)** as part of that command. With any OTMA descriptor, we can use an asterisk (*) at the end of the name of the descriptor. Therefore, a descriptor called OTMAD* matches OTMAD1 or OTMADX on a CHNG call. To display that one descriptor, we use **NAME(OTMAD*)** with the **OPTION(NOWILDCARD)**. If we want the command to look for any descriptors that start with OTMAD, we must specify **NAME(OTMAD*)** with **OPTION(WILDCARD)**. **OPTION(NOWILDCARD)** is the default and is supported for the DELETE OTMADESC command.

### 6.1.3 IMS Connect definitions

For IMS to make the connection from one IMS system to the other, you must update the IMS Connect configuration with a new RMTIMSCON definition. The remote IMS connection in the local IMS Connect must be defined to match the local OTMA descriptor. The **TMember** and **RMTIMSCON** definitions on the OTMADESC must match the local **DATASTORE MEMBER** and **RMTIMSCON ID** parameters in IMS connect. The RMTIMS definition must match the **DATASTORE ID** in the remote IMS connect. If defined, RMTTRAN must point to a transaction definition in the remote system, it cannot point to a remote LTERM.

Example 6-8 shows the local IMS definitions.

**Example 6-8  Local IMS Connect TCP/IP, data store, and remote IMS definitions**

```
TCPIP=(EXIT=(HWSSMPL1),HOSTNAME=TCPIP,PORT(ID=7301))

DATASTORE=(GROUP=I12XOTMA,ID=I12C,
            MEMBER=HWBI12C3,TMEMBER=I12COTMA)

RMTIMSCON=(ID=HWSI12D1,
           HOSTNAME=WTSC64.ITSO.IBM.COM,PORT=7401,
           AUTOCONN=Y,PERSISTENT=Y,
           IDLETO=60000,RESVSOC=10)
```

Example 6-9 shows the remote IMS definitions.

**Example 6-9  Remote IMS Connect TCP/IP, data store, and remote IMS definitions**

```
TCPIP=(EXIT=(HWSSMPL1),HOSTNAME=TCPIP,PORT(ID=7401))

DATASTORE=(GROUP=I12XOTMA,ID=I12D,
            MEMBER=HWBI112D4,TMEMBER=I12DOTMA)

RMTIMSCON=(ID=HWSI12C1,
           HOSTNAME=WTSC63.ITSO.IBM.COM,PORT=7301,
           AUTOCONN=Y,PERSISTENT=Y,
           IDLETO=60000,RESVSOC=10)
```
You can see how the RMTIMSCON defines the connection to the PORT ID used by the remote system in both cases.

### 6.1.4 New and modified IMS Connect commands

IMS 12 has introduced a new set of commands for IMS to IMS communications. Also a new interface is available from the IMS SPOC to operate IMS Connect.

#### New and updated IMS Connect WTOR reply commands

The write to operator with reply (WTOR) reply commands in the following sections are updated or added to support asynchronous IMS to IMS communications.

**The VIEWHWS command**

The **VIEWHWS** command is updated to display additional information about the RMTIMSCON definitions. Example 6-10 shows the new output.

**Example 6-10  Output from the VIEWHWS command**

```
HWSC0001I NO ACTIVE MSC

HWSC0001I RMTIMSCON=HWSI12D1  STATUS=NOT ACTIVE
HWSC0001I IP-ADDRESS=009.012.006.009  PORT=7401
HWSC0001I HOSTNAME=WTSC64.ITSO.IBM.COM

HWSC0001I AUTOCONN=Y  PERSISTENT=Y
HWSC0001I IDLETO=60000
HWSC0001I RESVSOC=10   NUMSOC=0
HWSC0001I NO ACTIVE CLIENTS
```

**The VIEWRMT ALL command**

The new **VIEWRMT ALL** command shows the status of RMTIMSCON definitions. Example 6-11 shows the output.

**Example 6-11  Output from the VIEWRMT ALL command**

```
R 685,VIEWRMT ALL
IEE600I REPLY TO 685 IS;VIEWRMT ALL

HWSC0001I RMTIMSCON=HWSI12B1  STATUS=NOT ACTIVE
HWSC0001I IP-ADDRESS=009.012.006.009  PORT=7202
HWSC0001I HOSTNAME=WTSC64.ITSO.IBM.COM

HWSC0001I AUTOCONN=N  PERSISTENT=Y
HWSC0001I IDLETO=0
HWSC0001I RESVSOC=10   NUMSOC=0
HWSC0001I NO ACTIVE CLIENTS
```

**The VIEWRMT rmtimscon command**

The new **VIEWRMT rmtimscon** command shows a specific RMTIMSCON definition. Example 6-12 shows the output, but only the named RMTIMSCON is displayed unlike the **VIEWRMT ALL** command, which shows all RMTIMSCON connections.

**Example 6-12  Output from the VIEWRMT xxxx command**

```
R 660,VIEWRMT HWSI12A1
IEE600I REPLY TO 660 IS;VIEWRMT HWSI12A1
```
The STARTRMT rmtimscon command
IMS has added the STARTRMT rmtimscon command to activate a RMTIMSCON connection. Example 6-13 shows the output.

Example 6-13   Output from the STARTRMT HWSI12C1 command
R 709,STARTRMT HWSI12C1
IEE600I REPLY TO 709 IS;STARTRMT HWSI12C1
HWST3500I COMMUNICATIONS WITH REMOTE IMS CONNECT HWSI12C1 STARTED;
M=TSCH

The STOPRMT rmtimscon command
The new STOPRMT rmtimscon command stops a RMTIMSCON connection. Example 6-14 shows the output.

Example 6-14   Output from the STOPRMT HWSI12C1 command
R 710,STOPRMT HWSI12C1
IEE600I REPLY TO 710 IS;STOPRMT HWSI12C1
HWST3505I COMMUNICATIONS WITH REMOTE IMS CONNECT HWSI12C1 STOPPED;
M=TSCH

New IMS Connect z/OS modify commands
The commands in the following sections are added as IMS Connect z/OS modify commands.

QUERY RMTIMSCON NAME(*)
The new QUERY RMTIMSCON NAME(*) command is equivalent to the VIEWRMT ALL command. Example 6-11 on page 165 shows the output.

QUERY RMTIMSCON NAME(rmtinm)
The new QUERY RMTIMSCON command is equivalent to VIEWRMT rmtimscon command. It only displays the listed RMTIMSCON. Example 6-12 on page 165 shows the output.

UPDATE RMTIMSCON NAME(rmtinm) START(COMM)
The new UPDATE RMTIMSCON NAME(rmtinm) START(COMM) command is equivalent to the STARTRMT rmtimscon command. Example 6-13 on page 166 shows the output.

UPDATE RMTIMSCON NAME(rmtinm) STOP(COMM)
The new UPDATE RMTIMSCON NAME(rmtinm) STOP(COMM) command is equivalent to the STOPRMT rmtimscon command. Example 6-14 on page 166 shows the output.
New IMS Connect commands

IMS Connect has a new type-2 command interface using the IMS SPOC. All of the existing WTOR replies and z/OS modify commands have a SPOC equivalent.

**QUERY IMSCON TYPE(RMTIMSCON) NAME(*) SHOW(ALL | showparm)**

The new `QUERY IMSCON TYPE(RMTIMSCON) NAME(*) SHOW(ALL | showparm)` command shows the status of all RMTIMSCON connections. Example 6-15 shows the output.

---

### Example 6-15  Output from QRY IMSCON TYPE(RMTIMSCON) NAME(*) SHOW(ALL)

<table>
<thead>
<tr>
<th>RmtImsCon MbrName</th>
<th>CC</th>
<th>IpAddress</th>
<th>HostName</th>
<th>Port</th>
<th>AutoConn</th>
<th>Persist</th>
<th>IdleTO</th>
<th>ResvSoc</th>
<th>NumSoc</th>
<th>Appl</th>
<th>UserID</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>HWSI12B1</td>
<td></td>
<td>0 9.12.6.9</td>
<td>WTSC64.ITSO.IBM.COM</td>
<td>7202</td>
<td>N</td>
<td>Y</td>
<td>0</td>
<td>10</td>
<td>0</td>
<td></td>
<td></td>
<td>NOTACTIVE</td>
</tr>
<tr>
<td>HWSI12B1</td>
<td></td>
<td>0 9.12.6.70</td>
<td>WTSC63.ITSO.IBM.COM</td>
<td>7102</td>
<td>N</td>
<td>Y</td>
<td>0</td>
<td>10</td>
<td>0</td>
<td></td>
<td></td>
<td>NOTACTIVE</td>
</tr>
<tr>
<td>HWSI12D1</td>
<td></td>
<td>0 9.12.6.9</td>
<td>WTSC64.ITSO.IBM.COM</td>
<td>7401</td>
<td>Y</td>
<td>Y</td>
<td>60000</td>
<td>10</td>
<td>0</td>
<td></td>
<td></td>
<td>NOTACTIVE</td>
</tr>
<tr>
<td>HWSI12C1</td>
<td></td>
<td>0 9.12.6.70</td>
<td>WTSC63.ITSO.IBM.COM</td>
<td>7301</td>
<td>Y</td>
<td>Y</td>
<td>60000</td>
<td>10</td>
<td>0</td>
<td></td>
<td></td>
<td>NOTACTIVE</td>
</tr>
</tbody>
</table>

---

**QUERY IMSCON TYPE(RMTIMSCON) NAME(rmtimn) SHOW(ALL | showparm)**

The new `QUERY IMSCON TYPE(RMTIMSCON) NAME(rmtimn) SHOW(ALL | showparm)` command shows the status of a specific RMTIMSCON connection. The output is similar to the output from the `QRY RMTIMSCON NAME(*) SHOW(...)` command, but it is restricted to the listed connections. Example 6-16 shows the output.

---

### Example 6-16  Output from the QRY IMSCON TYPE(RMTIMSCON) NAME(HWSI12C1) SHOW(ALL) command

<table>
<thead>
<tr>
<th>RmtImsCon MbrName</th>
<th>CC</th>
<th>IpAddress</th>
<th>HostName</th>
<th>Port</th>
<th>AutoConn</th>
<th>Persist</th>
<th>IdleTO</th>
<th>ResvSoc</th>
<th>NumSoc</th>
<th>Appl</th>
<th>UserID</th>
<th>Status</th>
<th>TotSClnts</th>
<th>TotRecv</th>
<th>TotConn</th>
<th>TotXmit</th>
<th>TotOther</th>
<th>SendClnt</th>
<th>SendUID</th>
<th>Second</th>
<th>SendPort</th>
<th>SendStatus</th>
</tr>
</thead>
<tbody>
<tr>
<td>HWSI12C1</td>
<td></td>
<td>0 9.12.6.9</td>
<td>WTSC63.ITSO.IBM.COM</td>
<td>7301</td>
<td>Y</td>
<td>Y</td>
<td>60000</td>
<td>10</td>
<td>0</td>
<td></td>
<td></td>
<td>ACTIVE</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

**UPDATE IMSCON TYPE(RMTIMSCON) NAME(rmtimn) START(COMM)**

The new `UPDATE IMSCON TYPE(RMTIMSCON) NAME(rmtimn) START(COMM)` command starts a RMTIMSCON connection. Example 6-17 shows the output.

---

### Example 6-17  Output from UPD IMSCON TYPE(RMTIMSCON) NAME(HWSI12C1) START(COMM)

<table>
<thead>
<tr>
<th>RmtImsCon MbrName</th>
<th>CC</th>
<th>CCText</th>
</tr>
</thead>
<tbody>
<tr>
<td>HWSI12C1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**UPDATE IMSCON TYPE(RMTIMSCON) NAME(rmtinm) STOP(COMM)**

The new `UPDATE IMSCON TYPE(RMTIMSCON) NAME(rmtinm) STOP(COMM)` command terminates communications for a RMTIMSCON connection. Example 6-18 shows the output.

*Example 6-18  Output from UPD IMSCON TYPE(RMTIMSCON) NAME(rmtinm) STOP(COMM) command*

<table>
<thead>
<tr>
<th>RmtImsCon MbrName</th>
<th>CC</th>
<th>CCText</th>
</tr>
</thead>
<tbody>
<tr>
<td>HWSI12C1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HWSI12D1</td>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>

**QUERY IMSCON TYPE(SENDCLNT) NAME(*) SHOW(ALL)**

The new `QUERY IMSCON TYPE(SENDCLNT) NAME(*) SHOW(ALL)` command shows the send sockets on the connection to the remote IMS Connect instance. Example 6-19 shows the output.

*Example 6-19  Output from QRY IMSCON TYPE(SENDCLNT) NAME(*) SHOW(ALL) command*

<table>
<thead>
<tr>
<th>SendClnt MbrName</th>
<th>CC</th>
<th>UserID</th>
<th>MscName</th>
<th>Second SendPort</th>
<th>RmtImsCon MbrName</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>OTMA9644 HWSI12C1</td>
<td>0</td>
<td>IMSR3</td>
<td></td>
<td>94</td>
<td>8148 HWSI12D1</td>
<td>CONN</td>
</tr>
<tr>
<td>OTM73C4B HWSI12D1</td>
<td>0</td>
<td>IMSR3</td>
<td></td>
<td>207</td>
<td>35616 HWSI12C1</td>
<td>CONN</td>
</tr>
</tbody>
</table>

**6.1.5  Super member support with asynchronous IMS to IMS communications**

IMS provides a super member function that enables multiple, redundant instances of IMS Connect, which is the integrated TCP/IP server of IMS, to run in parallel as a single application on multiple z/OS images. IMS has expanded the super member function to support the IMS synchronous callout messages. There is support for IMS Connect super members with the new IMS to IMS asynchronous communications.

When you create the OTMA descriptor, you need to add the new `SMEM(Y)` parameter and change the value for the `TMEMBER` parameter to be the super member name. Example 6-20 shows the modified `CREATE OTMADESC` command. We can also activate super member support on an existing OTMA descriptor with the `UPDATE OTMADESC` command.

*Example 6-20  CREATE OTMADESC with super member support*

```
CREATE OTMADESC NAME(OTMA2CD) -
SET(TYPE(IMSCON) TMEMBER(I12S) SMEM(Y) RMTIMS(I12D) -
  RMTIMSCON(HWSI12D1) RMTTRAN(OTMAPRNT))
```

If you define the OTMA descriptor in the `IMS.PROCLIB(DFSYDTx)` member, you must add `SMEM=YES` (the default is `SMEM=NO`) in that OTMA descriptor definition (Example 6-21).

*Example 6-21  IMS.PROCLIB member DFSYDTC with super member support*

```
D OTMA2CD TYPE=IMSCON TMEMBER=I12S RMTIMS=HWSI12D1
D OTMA2CD RMTTRAN=OTMAPRNT RMTIMS=I12D SMEM=YES
```
6.1.6 OTMA security

You define the security needed to establish the connection and run the transaction at the other remote IMS system in the following places:

- From the user ID that is running the transaction at the local system that uses the CHNG and ISRT DL/I calls to send a transaction to the destination defined by the OTMA descriptor
- From the user ID defined on the descriptor either in the DFSYDTx member (USERID=xxx) or on the OTMADESC USERID(yyy) parameter (overrides option 1)
- From the user ID defined in the USERID=xxx and APPL=yyy parameters on the RMTIMSCON statement in IMS Connect configuration member

The security here is used with RACF to generate a PassTicket for the remote system. It is just for connection security to ensure that the local IMS Connect is authorized to connect to the remote IMS Connect and vice versa. This approach does not affect the user ID passed from one IMS system to the other in the OTMA header.

For an explanation of how to define USERID=xxx, APPL=yyy, and RACF PTKTDATA definitions to create a secure connection, see 6.2.6, “IMS Connect security using RACF PassTickets” on page 186.

6.1.7 Configuration summary for asynchronous IMS to IMS communications

Figure 6-2 shows two IMS systems. IMS1 needs to send a transaction to IMS2. Therefore, the OTMA descriptor DESC1 is defined in IMS1. The descriptor defines ICON1 as the local IMS Connect system and ICON2 IMS2 as the remote IMS system.

ICON1 has a RMTIMSCON definition that defines ICON2 and gives the TCP/IP host name (or IP address) and the PORT of the remote IMS Connect. ICON2 is configured as normal with a TCP/IP statement that defines the PORT it should listen on and a DATASTORE to define the OTMA connection to its local IMS system IMS2.
A special security definition is provided so the transaction run at the remote system has the user ID (USERXYZ) as defined on the OTMA descriptor in IMS1. However no connection security is available using USER=xxx, APPL=yyy, and a PassTicket between ICON1 and ICON2.

In this example, if we need IMS2 to send transactions back to IMS1, we must add an RMTIMSCON in ICON2 and an OTMA descriptor in IMS2.

6.2 Multiple Systems Coupling using TCP/IP

IMS has had MSC using TYPE=CTC, TYPE=MTM, and TYPE=VTAM for many releases. The support in IMS 12 adds a new TYPE to the MSPLINK macro. We can now define an MSPLINK with TYPE=TCPIP.

In addition, a new function in IMS Connect establishes network connectivity between two IMS systems using TYPE=TCPIP MSPLINK. Figure 6-3 shows a general overview of how the TCP/IP MSC link support in IMS 12 is configured.

![Figure 6-3 MSC using TCP/IP for IMS to IMS connectivity](image)

6.2.1 Stage 1 definitions needed in IMS

The new TCP/IP MSC links must be defined in IMS in the stage1 or stage2 system generation in the same way that you define a TYPE=VTAM, TYPE=CTC, or TYPE=MTM link. Only the MSPLINK macro has changed. The other stage1 macros to define the MSLINK and MSNAME remain the same.

There are several changes to the MSPLINK macro to support a TCP/IP MSC link. The NAME parameter now defines the definition on a RMTIMSCON. Two additional parameters, LCLICON and LCLPLKID, define the IMS Connect and the MSC statement in the IMS Connect configuration that defines the link.

Example 6-22 shows the IMS stage1 macros that are needed to define an MSC link using TCP/IP for the local IMS system. The MSLINK and MSNAME macros have no special requirements.

Example 6-22 IMS stage 1 macros for a TCP/IP MSC link (local system)

| LINKBA | MSPLINK TYPE=TCPIP,NAME=I12B,SESSION=2,BUFSIZE=4096, | X |
|        | LCLICON=HWSI12A1, | X |
|        | LCLPLKID=MSC2A2B |
|        | MSLINK PARTNER=BA,MSPLINK=LINKBA |
| MSCBA  | MSNAME SYSID=(102,101) |
Example 6-23 shows the definitions for the remote IMS system.

**Example 6-23**  IMS stage 1 macros for a TCP/IP MSC link (remote system)

```plaintext
Example 6-23  IMS stage 1 macros for a TCP/IP MSC link (remote system)

LINKBA  MSPLINK TYPE=TCPIP,NAME=I12A,SESSION=2,BUFSIZE=4096,
        X
        LCLICON=HWSI12B1,  X
        LCLPLKID=MSC2B2A

MSLINK  PARTNER=BA,MSPLINK=LINKBA

MSCBA    MSNAME SYSID=(101,102)
```

**Restriction:** When the first TYPE=TCPIP MSPLINK is added to your IMS stage1, you must run an ALL or NUCLEUS system generation and restart IMS.

The relationship between the MSPLINK macro and the MSC configuration statements in IMS Connect is created by the IMSPLEx, TMEMber, NAME, LCLPLKID, LCLICON, and LCLIMS specifications.

The IMS MSVERIFY utility and /MSVERIFY command is enhanced in IMS 12 to support the new TCP/IP MSC links. To use the MSVERIFY command, you must add an MSVID operand to the INSMCTRL macro before running a stage1 or stage2 generation. This method causes additional modules to be written to the MODBLKS data set. You can then run the MSVERIFY utility (Example 6-24).

**Example 6-24**  MSVERIFY utility job control language (JCL)

```plaintext
// JCLLIB ORDER=IMS12Q.PROCLIB
//STEP1   EXEC DFSMSV,ALL=''
//SYSIN     DD *
001,002,003,004
/*
```

The output from running the MSVERIFY utility shows the configuration and assignments that are active when the linked IMS systems are restarted (Example 6-25).

**Example 6-25**  Output from IMS MS Verification utility

```
DFS23001 IMS - MS VERIFICATION UTILITY -  RUN TUESDAY  08/02/11 17:58:22

DFS23011 INPUT 001,002

DFS23021 SYSTEM MODULES
DFS23031 DFSMS001
DFS23031 DFSMS002
DFS23031 DFSMS003
DFS23031 DFSMS004

... |
<table>
<thead>
<tr>
<th>MS001</th>
<th>MS002</th>
<th>MS003</th>
<th>MS004</th>
</tr>
</thead>
<tbody>
<tr>
<td>0001</td>
<td>LOCAL</td>
<td>001 AB</td>
<td>001 AC</td>
</tr>
<tr>
<td>0002</td>
<td>002 AB</td>
<td>LOCAL</td>
<td>002 BC</td>
</tr>
<tr>
<td>0003</td>
<td>003 AC</td>
<td>003 BC</td>
<td>LOCAL</td>
</tr>
<tr>
<td>0004</td>
<td>004 AD</td>
<td>004 BD</td>
<td>004 CD</td>
</tr>
<tr>
<td>0101</td>
<td>LOCAL</td>
<td>001 BA</td>
<td>001 CA</td>
</tr>
<tr>
<td>0102</td>
<td>002 BA</td>
<td>LOCAL</td>
<td>002 CB</td>
</tr>
</tbody>
</table>
```

6.2.2 New and modified IMS commands

IMS 12 has added new function to some IMS type-1 commands and added or updated various type-2 commands to support operating the new MSC links using TCP/IP.

Updated IMS type-1 commands
Only one IMS type-1 command has changed for TCP/IP MSC links.

/DISPLAY ASSIGNMENT MSPLINK ALL

The /DISPLAY ASSIGNMENT MSPLINK command has changed to show the status of TCP/IP physical links. Example 6-26 shows the output.

Example 6-26   Output from the /DIS ASSIGNMENT MSPLINK ALL command

<table>
<thead>
<tr>
<th>DFS4444I DISPLAY FROM ID=I12A</th>
</tr>
</thead>
<tbody>
<tr>
<td>LINK PLINK TYPE ADDR MAXSESS NODE</td>
</tr>
<tr>
<td>1 LINKAB VTAM 36000007 2 APPLI12B</td>
</tr>
<tr>
<td>2 LINKBA TCP1P **** 2 I12B</td>
</tr>
<tr>
<td>3 LINKAC VTAM 00000000 2 APPLI12C</td>
</tr>
<tr>
<td>4 LINKCA TCP1P **** 2 I12C</td>
</tr>
<tr>
<td>5 LINKAD VTAM 58000006 2 APPLI12D</td>
</tr>
<tr>
<td>6 LINKDA TCP1P **** 2 I12D</td>
</tr>
</tbody>
</table>

*11214/165528*

New and updated IMS type-2 commands
IMS has added or modified type-2 query commands to support TCP/IP MSC links.

QUERY MSPLINK NAME(msplinkname | *) SHOW(ALL)

The QUERY MSPLINK NAME(msplinkname | *) SHOW(ALL) command is updated to show the TCP/IP MSC links. Example 6-27 on page 173 shows the output.
Example 6-27  Output from QUERY MSPLINK NAME(*) SHOW(ALL)

Response for: QRY MSPLINK NAME(*) SHOW(ALL)

<table>
<thead>
<tr>
<th>MSPLink</th>
<th>MbrName</th>
<th>CC Type</th>
<th>NodeName</th>
<th>RmtIms</th>
<th>LclImsCon</th>
<th>LclPlkID</th>
<th>LclStat</th>
</tr>
</thead>
<tbody>
<tr>
<td>LINKAB</td>
<td>I12A</td>
<td>0</td>
<td>VTAM</td>
<td>APPLI12B</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LINKAB</td>
<td>I12B</td>
<td>0</td>
<td>VTAM</td>
<td>APPLI12A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LINKAC</td>
<td>I12A</td>
<td>0</td>
<td>VTAM</td>
<td>APPLI12C</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LINKAD</td>
<td>I12A</td>
<td>0</td>
<td>VTAM</td>
<td>APPLI12D</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LINKBA</td>
<td>I12A</td>
<td>0</td>
<td>TCPIP</td>
<td>I12B</td>
<td>HWSI12A1</td>
<td>MSC2A2B</td>
<td></td>
</tr>
<tr>
<td>LINKBA</td>
<td>I12B</td>
<td>0</td>
<td>TCPIP</td>
<td>I12A</td>
<td>HWSI12B1</td>
<td>MSC2B2A</td>
<td></td>
</tr>
<tr>
<td>LINKBC</td>
<td>I12B</td>
<td>0</td>
<td>VTAM</td>
<td>APPLI12C</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LINKBD</td>
<td>I12B</td>
<td>0</td>
<td>VTAM</td>
<td>APPLI12D</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LINKCA</td>
<td>I12A</td>
<td>0</td>
<td>TCPIP</td>
<td>I12C</td>
<td>HWSI12A1</td>
<td>MSC2A2C</td>
<td></td>
</tr>
<tr>
<td>LINKCB</td>
<td>I12B</td>
<td>0</td>
<td>TCPIP</td>
<td>I12C</td>
<td>HWSI12B1</td>
<td>MSC2B2C</td>
<td></td>
</tr>
<tr>
<td>LINKDA</td>
<td>I12A</td>
<td>0</td>
<td>TCPIP</td>
<td>I12D</td>
<td>HWSI12A1</td>
<td>MSC2A2D</td>
<td></td>
</tr>
<tr>
<td>LINKDB</td>
<td>I12B</td>
<td>0</td>
<td>TCPIP</td>
<td>I12D</td>
<td>HWSI12B1</td>
<td>MSC2B2D</td>
<td></td>
</tr>
</tbody>
</table>

Example 6-28 shows the MSPLINK being stopped, updated to have a new local ICONPLKID, and then restarted.

Example 6-28  Output from UPD MSPLINK NAME(...) SET(ICONPLKID(...)) command

Response for: UPD MSPLINK NAME(LINKBA) STOP(LOGON)

<table>
<thead>
<tr>
<th>MSPLink</th>
<th>MbrName</th>
<th>CC</th>
</tr>
</thead>
<tbody>
<tr>
<td>LINKBA</td>
<td>I12A</td>
<td>0</td>
</tr>
</tbody>
</table>

Response for: UPD MSPLINK NAME(LINKBA) SET(ICONPLKID(MSC2A2B))

<table>
<thead>
<tr>
<th>MSPLink</th>
<th>MbrName</th>
<th>CC</th>
</tr>
</thead>
<tbody>
<tr>
<td>LINKBA</td>
<td>I12A</td>
<td>0</td>
</tr>
</tbody>
</table>

Response for: UPD MSPLINK NAME(LINKBA) START(LOGON)

<table>
<thead>
<tr>
<th>MSPLink</th>
<th>MbrName</th>
<th>CC</th>
</tr>
</thead>
<tbody>
<tr>
<td>LINKBA</td>
<td>I12A</td>
<td>0</td>
</tr>
</tbody>
</table>

Example 6-29 shows the output from a link being stopped, updated with a new IMSCON name, and then restarted.

Example 6-29  Output from UPD MSPLINK NAME(...) SET(IMSCON(...)) command

Response for: UPD MSPLINK NAME(LINKBA) STOP(LOGON)

<table>
<thead>
<tr>
<th>MSPlink</th>
<th>MbrName</th>
<th>CC</th>
</tr>
</thead>
<tbody>
<tr>
<td>LINKBA</td>
<td>I12A</td>
<td>0</td>
</tr>
</tbody>
</table>

Response for: UPD MSPLINK NAME(LINKBA) SET(IMSCON(HWSI12A1))

Example 6-29 shows the output from a link being stopped, updated with a new IMSCON name, and then restarted.
**UPDATE MSPLINK NAME(msplinkname) SET(RMTIMS(rmtims))**

The `UPDATE MSPLINK NAME(msplinkname) SET(RMTIMS(rmtims))` command has a new operand that you can use to change the name of the remote IMS system. Example 6-30 shows the output from a link being stopped, updated with a new remote name, and then restarted.

**Example 6-30   Output form UPD MSPLINK NAME(...) SET(RMTIMS(...)) command**

Response for: UPD MSPLINK NAME(LINKBA) STOP(LOGON)
MSPLink  MbrName    CC
--------------------------------------------------------
LINKBA   I12A        0
Response for: UPD MSPLINK NAME(LINKBA) SET(RMTIMS(I12B))
MSPLink  MbrName    CC
--------------------------------------------------------
LINKBA   I12A        0
Response for: UPD MSPLINK NAME(LINKBA) START(LOGON)
MSPLink  MbrName    CC
--------------------------------------------------------
LINKBA   I12A        0

### 6.2.3 IMS Connect definitions

The new TCP/IP MSC links require two additional configurations in IMS Connect. An MSC statement to define the local `MSPLINK` and a `RMTIMSCON` statement to define the remote IMS Connect system. Example 6-31 shows the local definition required in IMS Connect.

**Example 6-31   IMS Connect configuration for a TCP/IP MSC link (local system)**

```
HWS=(ID=HWSI12A1,PSWDMC=N,RACF=N,RRS=Y,UIDCACHE=Y,XIBAREA=50)
TCPIP=(EXIT=(HWSSMPL1,HWSSOAP1),HOSTNAME=TCPIP,PORT(ID=7102))
MSC=(LCLPLKID=MSC2A2B,RMTPLKID=MSC2B2A,
    LCLIMS=I12A,RMTIMS=I12B,
    IMSPLEX=(MEMBER=HWSI12A1,TMEMBER=IM12X),
    RMTIMSCON=HWSI12B1)
RMTIMSCON=(ID=HWSI12B1,
    HOSTNAME=WTSC64.ITSO.IBM.COM,PORT=7202,
    PERSISTENT=Y,RESVSOC=1)
```
Example 6-32 shows the remote definition required in IMS Connect.

Example 6-32  IMS Connect configuration for a TCP/IP MSC link (remote system)

```
HWS=(ID=HWSI12B1,PSWDMC=N,RACF=N,RRS=Y,UIDCACHE=Y,XIBAREA=50)

TCPIP=(EXIT=(HWSSMPL1,HWSSOAP1),HOSTNAME=TCPPIP,PORT(ID=7202))

MSC=(LCLPLKID=MSC2B2A,RMTPLKID=MSC2A2B,
    LCLIMS=I12B,RMTIMS=I12A,
    IMSPLEX=(MEMBER=HWSI12B1,TMEMBER=IM12X),
    RMTIMSCON=HWSI12A1)

RMTIMSCON=(ID=HWSI12A1,
    HOSTNAME=WTSC63.ITSO.IBM.COM,PORT=7102,
    PERSISTENT=Y,RESVSOC=1)
```

The relationship between the MSC and RMTIMSCON configuration statements in IMS Connect is created by the RMTPLKID, RMTIMS, RMTIMSCON, and ID specifications.

The number of reserved sockets (RESVSOC) is determined by the number of logical links (MSLINK macros) that are defined for the MSC link in the IMS stage1 and by the number of existing MSLINK macros that are assigned, by using the /MSASSIGN command, to the new TCP/IP MSC link.

The use of AUTOCONN=N, PERSISTENT=N, and a non-zero IDLETO is disallowed for RMTIMSCON connections used by MSC. IMS Connect resets the values (Example 6-33).

Example 6-33  IMS Connect automatic parameter changes

```
HWSX0920W VALUE OF PARAMETER AUTOCONN IN STATEMENT RMTIMSCON=HWSI12B 
CHANGED TO N FROM Y BECAUSE PERSISTENT=N WAS SPECIFIED ; M=XCFG
HWSX0920W VALUE OF PARAMETER PERSISTENT IN STATEMENT RMTIMSCON=HWSI12B 
CHANGED TO Y FROM N BECAUSE THE RMTIMSCON IS USED BY MSC; M=XCFG
HWSX0920W VALUE OF PARAMETER IDLETO IN STATEMENT RMTIMSCON=HWSI12B 
CHANGED TO 0 FROM 60000 BECAUSE THE RMTIMSCON IS USED BY MSC; M=XCFG
```

6.2.4 New and modified IMS Connect commands

The command processing for IMS Connect has had a number of new and updated commands to support TCP/IP MSC links and RMTIMSCON connections. WE have grouped them in the following sections:

- New and updated IMS Connect WTOR reply commands
- New and updated IMS Connect z/OS modify commands
- New type-2 IMS Connect Single Point of Control commands

New and updated IMS Connect WTOR reply commands

The commands in the following sections are updated or added as replies to the IMS Connect WTOR.

The VIEWHWS command

The VIEWHWS command is updated to add details of the MSC links and RMTIMSCON connections. Example 6-34 on page 176 shows the additional output.
Example 6-34  Output from the VIEWHWS command

HWSC0001I  MSC=MSC2A2B  STATUS=ACTIVE
HWSC0001I  RMTPLKID=MSC2B2A
HWSC0001I  LCLIMSID=I12A  RMTIMSID=I12B  GENIMSID=
  AFFINITY=
HWSC0001I  IMSPLEX=IM12X
HWSC0001I  MEMBER=HWS112A1  TARGET MEMBER=IM12X
HWSC0001I  RMTIMSCON=HWS112B1
HWSC0001I  IP-ADDRESS=009.012.006.009  PORT=7202
HWSC0001I  HOSTNAME=WTSC64.ITSO.IBM.COM

HWSC0001I  LINK  PARTNERID  STATUS  SENDCLNT  RECVCLNT
HWSC0001I  DFLS0002  BA  ACTIVE  MSCCE084  MSCE934B

HWSC0001I  RMTIMSCON=HWS112B1  STATUS=ACTIVE
HWSC0001I  IP-ADDRESS=009.012.006.009  PORT=7202
HWSC0001I  HOSTNAME=WTSC64.ITSO.IBM.COM

HWSC0001I  AUTOCONN=N  PERSISTENT=Y
HWSC0001I  IDLETO=0
HWSC0001I  RESVSOC=10  NUMSOC=1
HWSC0001I  SENDCLNT LCLPLKID STATUS  SECOND SENDPORT
HWSC0001I  MSCCE084 MSC2A2B  CONN  3189  8438
HWSC0001I  TOTAL SENDCLNTS=1  RECV=0  CONN=1  XMIT=0  OTHER=0

The VIEWMSC ALL command
The new VIEWMSC ALL command shows the status of IMS Connect MSC connections. Example 6-35 shows the output.

Example 6-35  Output from the VIEWMSC ALL command

R 719,VIEWMSC ALL
IEE600I  REPLY TO 719 IS;VIEWMSC ALL
HWSC0001I  MSC=MSC2A2B  STATUS=ACTIVE
HWSC0001I  RMTPLKID=MSC2B2A
HWSC0001I  LCLIMSID=I12A  RMTIMSID=I12B  GENIMSID=
  AFFINITY=
HWSC0001I  IMSPLEX=IM12X
HWSC0001I  MEMBER=HWS112A1  TARGET MEMBER=IM12X
HWSC0001I  RMTIMSCON=HWS112B1
HWSC0001I  IP-ADDRESS=009.012.006.009  PORT=7202
HWSC0001I  HOSTNAME=WTSC64.ITSO.IBM.COM

HWSC0001I  LINK  PARTNERID  STATUS  SENDCLNT  RECVCLNT
HWSC0001I  DFLS0002  BA  ACTIVE  MSCCE084  MSCE934B

The VIEWMSC mscid command
The new VIEWMSC mscid command shows a named MSC connection. Example 6-35 shows the output, but only one single MSC connection is displayed.

The VIEWRMT ALL command
The new VIEWRMT ALL command shows the status of all RMTIMSCON connections. Example 6-11 on page 165 shows the output.
The VIEWRMT rmtimscon command
The new VIEWRMT rmtimscon command shows the status of a specific RMTIMSCON connection. Example 6-12 on page 165 shows the output.

The STOPLINK linkname command
The new STOPLINK linkname command stops communication on an MSC link. To find the linkname, use the VIEWMSC command because there is no VIEWLINK command. Example 6-36 shows the output.

Example 6-36  Output from the STOPLINK linkname command
R 724,STOPLINK DFSL0002
IEE600I REPLY TO 724 IS;STOPLINK DFSL0002
HWSF3310I LOGICAL LINK DFSL0002 TERMINATED; MSC=MSC2A2B , M=DSCM
HWSST3525I THE SEND CLIENT MSC70986 WAS TERMINATED FOR REMOTE IMS
CONNECT HWS12B1; M=TCVC

The STOPLINK linkname lclplkid command
The new STOPLINK linkname lclplkid command stops communications on an MSC link (qualified with the MSC local physical link name). Example 6-37 shows the output.

Example 6-37  Output from the STOPLINK DFSL0001 MSC2A2B command
R 725,STOPLINK DFSL0002 MSC2A2B
IEE600I REPLY TO 725 IS;STOPLINK DFSL0002 MSC2A2B
HWSF3310I LOGICAL LINK DFSL0002 TERMINATED; MSC=MSC2A2B , M=DSCM
HWSST3525I THE SEND CLIENT MSC1028B WAS TERMINATED FOR REMOTE IMS
CONNECT HWS12B1; M=TCVC

The STARTMSC lclplkid command
The new STARTMSC lclplkid command starts communications on an MSC link. Example 6-38 shows the output.

Example 6-38  Output from the STARTMSC MSC2A2B command
R 727,STARTMSC MSC2A2B
IEE600I REPLY TO 727 IS;STARTMSC MSC2A2B
HWSF3300I COMMUNICATIONS ON MSC PHYSICAL LINK MSC2A2B STARTED; M=ISC1

The STOPMSC lclplkid command
The new STOPMSC lclplkid command terminates communications on an MSC link. Example 6-39 shows the output.

Example 6-39  Output from the STOPMSC MSC2A2B command
R 726,STOPMSC MSC2A2B
IEE600I REPLY TO 726 IS;STOPMSC MSC2A2B
HWSF3360I THE TRANSMIT THREAD TERMINATED FOR MSC PHYSICAL LINK MSC2A2B ; M=IXMT
HWSST3525I THE SEND CLIENT MSCADD4E WAS TERMINATED FOR REMOTE IMS
CONNECT HWS12B1; M=TCVC
HWSF3360I THE RECEIVE  THREAD TERMINATED FOR MSC PHYSICAL LINK MSC2A2B ; M=IREC
HWSF3305I COMMUNICATIONS ON MSC PHYSICAL LINK MSC2A2B STOPPED;
M=DSCM
The STARTRMT rmtimscon command

The new STARTRMT rmtimscon command activates an IMS Connect RMTIMSCON connection. Example 6-40 shows the output.

Example 6-40 Output from the STARTRMT HWSI12B1 command

R 703,STARTRMT HWSI12B1
IEE600I REPLY TO 703 IS;STARTRMT HWSI12B1
HWST3500I COMMUNICATIONS WITH REMOTE IMS CONNECT HWSI12B1 STARTED;
M=TSCH

The STOPRMT rmtimscon command

The new STOPRMT rmtimscon command stops an RMTIMSCON connection. Example 6-41 shows the output.

Example 6-41 Output from the STOPRMT HWSI12B1 command

R 702,STOPRMT HWSI12B1
IEE600I REPLY TO 702 IS;STOPRMT HWSI12B1
HWST3505I COMMUNICATIONS WITH REMOTE IMS CONNECT HWSI12B1 STOPPED;
M=TSCH
HWST3525I THE SEND CLIENT MSCE6706 WAS TERMINATED FOR REMOTE IMS CONNECT HWSI12B1; M=TCVC
DFS3176E 19:04:23 IMS CONNECT ERROR MESSAGE RECEIVED, I12A
DFS000I SENDMSG, RETCODE = 0C000018, RSCNCODE = 0006408, LOSTSESS = NONE, I12A
DFS000I MODULE = DFSTC7A0, LINK = 002, DFSLO002 I12A
DFS2169I 19:04:23 DISCONNECTION COMPLETED ON LINK 0002 I12A

Notice that IMS reacted by disconnecting the MSC link, as shown in message DFS2169I.

The STOPSCLN rmtimscon sendclient command

The new STOPSCLN rmtimscon sendclient command terminates the send sockets on the connection to the remote IMS Connect instance. Example 6-42 shows the output.

Example 6-42 Output from the STOPSCLN MSCxxxxx command

R 723,STOPSCLN HWSI12B1 MSC7FDC9
IEE600I REPLY TO 723 IS;STOPSCLN HWSI12B1 MSC7FDC9
HWST3525I THE SEND CLIENT MSC7FDC9 WAS TERMINATED FOR REMOTE IMS CONNECT HWSI12B1; M=TCVC

New and updated IMS Connect z/OS modify commands

IMS Connect has also added or modified the new type-2 modify commands in the following sections.

QUERY MEMBER TYPE(IMSCON) SHOW(ALL)

The QUERY MEMBER TYPE(IMSCON) SHOW(ALL) command is updated to add the new status about MSC links and RMTIMSCON connections. The output is equivalent to the VIEWHWS command. Example 6-34 on page 176 shows the output.

QUERY MSC NAME(*)

The new QUERY MSC NAME(*) command shows the status of all MSC links. The output is equivalent to the VIEWMSC ALL command. Example 6-35 on page 176 shows the output.
QUERY MSC NAME(mscid)
The new QRY MSC NAME(mscid) command shows the status of a specific MSC link. The output is equivalent to the VIEWMSC mscid command. Example 6-35 on page 176 shows the output, but only one MSC link is displayed.

QUERY RMTIMSCON NAME(*)
The new QUERY RMTIMSCON NAME(*) command is equivalent to the VIEWRMT ALL command. Example 6-11 on page 165 shows the output.

QUERY RMTIMSCON NAME(rmtinm)
The new QUERY RMTIMSCON NAME(rmtinm) command is equivalent to the VIEWRMT rmtimscon command except that it only displays the listed RMTIMSCON. Example 6-12 on page 165 shows the output.

DELETE LINK NAME(linkname)
The new DELETE LINK NAME(linkname) command is equivalent to the STOPLINK xxx command. Example 6-43 shows the output. You must determine the MSC link name from a VIEWMSC ALL, VIEWMSC xxx, or QUERY MSC NAME(...) command. There is no VIEWLINK or QUERY LINK NAME(...) command.

Example 6-43  Output from the DELETE LINK NAME(DFSL0002) command
F IM12AHW1,DELETE LINK NAME(DFSL0002)
HWSF3310I LOGICAL LINK DFSL0002 TERMINATED; MSC=MSC2A2B , M=DSCM

DELETE LINK NAME(linkname) LCLPLKID(lclplkid)
The new DELETE LINK NAME(linkname) LCLPLKID(lclplkid) command is equivalent to the STOPLINK linkname lclplkid command. Example 6-37 on page 177 shows the output.

UPDATE MSC NAME(lclplkid) START(COMM)
The new UPDATE MSC NAME(lclplkid) command is equivalent to the STARTMSC lclplkid command. Example 6-38 on page 177 shows the output.

UPDATE MSC NAME(lclplkid) STOP(COMM)
The new UPDATE MSC NAME(lclplkid) command is equivalent to the STOPMSC lclplkid command. Example 6-39 on page 177 shows the output.

UPDATE RMTIMSCON NAME(rmtinm) START(COMM)
The UPDATE RMTIMSCON NAME(rmtinm) START(COMM) is the equivalent to the STARTRMT rmtimscon command. Example 6-40 on page 178 shows the output.

UPDATE RMTIMSCON NAME(rmtinm) STOP(COMM)
The UPDATE RMTIMSCON NAME(rmtinm) STOP(COMM) is the equivalent to the STOPRMT rmtimscon command. Example 6-41 on page 178 shows the output.

DELETE RMTIMSCON NAME(rmtinm) SENDCLNT(clientid)
The DELETE RMTIMSCON NAME(rmtinm) SENDCLNT(clientid) is the equivalent to the STOPSCLN rmtimscon clientid command. Example 6-42 on page 178 shows the output.

New type-2 IMS Connect Single Point of Control commands
As part of the support to allow IMS Connect to be operated from an IMS SPOC, the commands in the following sections are added.
**QUERY IMSCON TYPE(CONFIG) SHOW(ALL | showparm)**

The new `QUERY IMSCON TYPE(CONFIG) SHOW(...)` command show the status of the IMS Connect system. Example 6-44 shows the output.

Example 6-44  Output from the QRY IMSCON TYPE(CONFIG) SHOW(ALL) command

<table>
<thead>
<tr>
<th>MbrName</th>
<th>CC</th>
<th>Version</th>
<th>IconID</th>
<th>IPAddress</th>
<th>MaxSoc</th>
<th>TimeOut</th>
<th>NumSoc</th>
<th>WarnInc</th>
<th>UidCache</th>
<th>UidAge</th>
<th>RACF</th>
<th>PswdMc</th>
<th>RRS</th>
<th>RRSStat</th>
<th>Recorder</th>
<th>SMem</th>
<th>Cm0Atoq</th>
<th>Adapter</th>
<th>ODBMAC</th>
<th>ODBMTO</th>
</tr>
</thead>
<tbody>
<tr>
<td>HWSI12A1</td>
<td>0</td>
<td>V12</td>
<td>HWSI12A1</td>
<td>009.012.006.070</td>
<td>50</td>
<td>0</td>
<td>5</td>
<td>80</td>
<td>5 Y</td>
<td>2147483647</td>
<td>N</td>
<td>Y</td>
<td>REGISTERED</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>18000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HWSI12B1</td>
<td>0</td>
<td>V12</td>
<td>HWSI12B1</td>
<td>009.012.006.009</td>
<td>50</td>
<td>0</td>
<td>5</td>
<td>80</td>
<td>5 Y</td>
<td>2147483647</td>
<td>N</td>
<td>Y</td>
<td>REGISTERED</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>18000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HWSI12C1</td>
<td>0</td>
<td>V12</td>
<td>HWSI12C1</td>
<td>009.012.006.070</td>
<td>50</td>
<td>0</td>
<td>5</td>
<td>80</td>
<td>5 Y</td>
<td>2147483647</td>
<td>N</td>
<td>Y</td>
<td>REGISTERED</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>18000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HWSI12D1</td>
<td>0</td>
<td>V12</td>
<td>HWSI12D1</td>
<td>009.012.006.009</td>
<td>50</td>
<td>0</td>
<td>5</td>
<td>80</td>
<td>5 Y</td>
<td>2147483647</td>
<td>N</td>
<td>Y</td>
<td>REGISTERED</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>18000</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**QUERY IMSCON TYPE(MSC) NAME(*) SHOW(ALL | showparm)**

The new `QUERY IMSCON TYPE(MSC) NAME(*) SHOW(...)` command shows the status of the MSC links in IMS Connect. Example 6-45 shows the output.

Example 6-45  Output from the QRY IMSCON TYPE(MSC) NAME(*) SHOW(ALL) command

<table>
<thead>
<tr>
<th>MscName</th>
<th>MbrName</th>
<th>CC</th>
<th>CCText</th>
<th>Link</th>
<th>Partner</th>
<th>SendClnt</th>
<th>RecvClnt</th>
<th>LinkStatus</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSC2A2B</td>
<td>HWSI12A1</td>
<td>0</td>
<td></td>
<td>MSC2B2A</td>
<td>I12A</td>
<td>I12B</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>HWSI12A1</td>
<td>IM12X</td>
<td>HWSI12B1</td>
<td>9.12.6.9</td>
<td>WTSC64.ITSO.IBM.COM</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>DFSL0002</td>
<td>BA</td>
<td>MSC51A86</td>
<td>ACTIVE</td>
<td></td>
</tr>
<tr>
<td>MSC2B2A</td>
<td>HWSI12B1</td>
<td>0</td>
<td></td>
<td>DFSL0002</td>
<td>BA</td>
<td>MSC51A86</td>
<td>ACTIVE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>HWSI12B1</td>
<td>IM12X</td>
<td>HWSI12A1</td>
<td>9.12.6.70</td>
<td>WTSC63.ITSO.IBM.COM</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>DFSL0002</td>
<td>BA</td>
<td>MSC51A86</td>
<td>ACTIVE</td>
<td></td>
</tr>
</tbody>
</table>

**QUERY IMSCON TYPE(MSC) NAME(mscid) SHOW(ALL | showparm)**

The new `QUERY IMSCON TYPE(MSC) NAME(mscid) SHOW(...)` command shows the status of a specific MSC link in IMS Connect. The output is equivalent only the MSC link matching the name pattern are displayed. Example 6-46 shows the output restricted to just the SHOW(LINK) option.

Example 6-46  Output from the QRY IMSCON TYPE(MSC) NAME(MSC2B2A) SHOW(LINK) command

<table>
<thead>
<tr>
<th>MscName</th>
<th>MbrName</th>
<th>CC</th>
<th>CCText</th>
<th>Link</th>
<th>Partner</th>
<th>SendClnt</th>
<th>RecvClnt</th>
<th>LinkStatus</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### QUERY IMSCON TYPE(RMTIMSCON) NAME(*) SHOW(ALL / showparm)

The new `QUERY IMSCON TYPE(RMTIMSCON) NAME(*) SHOW(...)` command shows the status of all RMTIMSCON connections in IMS Connect. Example 6-47 shows the output.

**Example 6-47   Output from the QRY IMSCON TYPE(RMTIMSCON) NAME(*) SHOW(ALL) command**

| RmtImsCon | MbrName | CC IPAddress | HostName              | Port | AutoConn | Persist | IdleTO | ResvSoc | NumSoc | Appl | UserID | Status | TotSClnts | TotRecv | TotConn | TotXmit | TotOther | SendClnt | LclPlkID | Second | SendPort | SendStatus |
|-----------|---------|--------------|-----------------------|------|----------|---------|--------|---------|--------|------|--------|--------|----------|---------|-------|-------|--------|----------|----------|--------|--------|----------|-----------|
| HWSI12B1  | HWSI12A1| 0 9.12.6.9    | WTSC64.ITSO.IBM.COM    | 7202 | N        | Y       | 0      | 10      | 1      |      |        | ACTIVE | 1         | 0       | 1      | 0       | 0        |          |         |        |          |           |
| HWSI12B1  | HWSI12A1| 0             |                       |      |          |         |        |         |        |      |        |        |           |         |       |        |        |          |          |        |        |          |           |
| MSCE6706  | MSC2A2B | 1750          | 8405                 |      |          |         |        |         |        |      |        |        |           |         |       |        |        |          |          |        |        |          |           |

### QUERY IMSCON TYPE(SENDCLNT) NAME(*) SHOW(ALL)

The new `QUERY IMSCON TYPE(SENDCLNT) NAME(*) SHOW(...)` command shows details about the TCP/IP sending clients. Example 6-48 shows the output.

**Example 6-48   Output from the QRY IMSCON TYPE(SENDCLNT) NAME(*) SHOW(ALL) command**

<table>
<thead>
<tr>
<th>SendClnt</th>
<th>MbrName</th>
<th>CC UserID</th>
<th>MscName</th>
<th>Second</th>
<th>SendPort</th>
<th>RmtImsCon</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSC15F4C</td>
<td>HWSI12A1</td>
<td>0</td>
<td>MSC2A2B</td>
<td>3030</td>
<td>8137</td>
<td>HWSI12B1</td>
<td>CONN</td>
</tr>
<tr>
<td>MSC1C844</td>
<td>HWSI12B1</td>
<td>0</td>
<td>MSC2B2A</td>
<td>3030</td>
<td>35474</td>
<td>HWSI12A1</td>
<td>CONN</td>
</tr>
</tbody>
</table>
**UPDATE IMSCON TYPE(LINK) NAME(linkname) STOP(COMM)**

The new `UPDATE IMSCON TYPE(LINK) NAME(linkname) STOP(COMM)` command stops communication on an MSC link. You first need to find the link name using the `QRY IMSCON TYPE(MSC) NAME(MSC2B2A) SHOW(LINK)` command as shown in Example 6-46 on page 180. Example 6-49 shows the results of running this command.

**Example 6-49  Output from UPD IMSCON TYPE(LINK) NAME(DFSLLxxxx) STOP(COMM)**

Response for: UPDATE IMSCON TYPE(LINK) NAME(DFSL0002) STOP(COMM)

<table>
<thead>
<tr>
<th>Link</th>
<th>MscName</th>
<th>MbrName</th>
<th>CC</th>
<th>CCText</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFSL0002</td>
<td>MSC2A2B</td>
<td>HWS12A1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>DFSL0002</td>
<td>MSC2B2A</td>
<td>HWS12B1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

**UPDATE IMSCON TYPE(LINK) NAME(linkname) MSC(lclplkid) STOP(COMM)**

The new `UPDATE IMSCON TYPE(LINK) NAME(linkname) MSC(lclplkid) STOP(COMM)` command terminates communications on a named MSC link. In some cases, more than one system can have the same IMS generated LINK name so that you can qualify the command with the MSC name. Example 6-50 shows the output.

**Example 6-50  Output from the UPD IMSCON TYPE(LINK) NAME(x) MSC(m) STOP(COMM) command**

Response for: UPDATE IMSCON TYPE(LINK) NAME(DFSL0002) MSC(MSC2A2B) STOP(COMM)

<table>
<thead>
<tr>
<th>Link</th>
<th>MscName</th>
<th>MbrName</th>
<th>CC</th>
<th>CCText</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFSL0002</td>
<td>MSC2A2B</td>
<td>HWS12A1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

**UPDATE IMSCON TYPE(MSC) NAME(lclplkid) START(COMM)**

The new `UPDATE IMSCON TYPE(MSC) NAME(lclplkid) START(COMM)` command starts communications for an MSC link. Example 6-51 shows the output.

**Example 6-51  Output from UPD IMSCON TYPE(MSC) NAME(MSC2A2B) START(COMM)**

Response for: UPDATE IMSCON TYPE(MSC) NAME(MSC2A2B) START(COMM)

<table>
<thead>
<tr>
<th>MscName</th>
<th>MbrName</th>
<th>CC</th>
<th>CCText</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSC2A2B</td>
<td>HWS12A1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

**UPDATE IMSCON TYPE(MSC) NAME(lclplkid) STOP(COMM)**

The new `UPDATE IMSCON TYPE(MSC) NAME(lclplkid) STOP(COMM)` command terminates communication for an MSC link. Example 6-52 shows the output.

**Example 6-52  Output from UPD IMSCON TYPE(MSC) NAME(MSC2A2B) STOP(COMM)**

Response for: UPDATE IMSCON TYPE(MSC) NAME(MSC2A2B) STOP(COMM)

<table>
<thead>
<tr>
<th>MscName</th>
<th>MbrName</th>
<th>CC</th>
<th>CCText</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSC2A2B</td>
<td>HWS12A1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

**UPDATE IMSCON TYPE(RMTIMSCON) NAME(rmtinm) START(COMM)**

The new `UPDATE IMSCON TYPE(RMTIMSCON) NAME(rmtinm) START(COMM)` command initiates communications from the local IMS Connect to a remote IMS Connect system. Example 6-53 on page 183 shows the output.
Example 6-53  Output from UPD IMSCON TYPE(RMTIMSCON) NAME(rmtinm) START(COMM)

Response for: UPDATE IMSCON TYPE(RMTIMSCON) NAME(HWSI12A1) START(COMM)
RmtImsCon MbrName            CC CCText
----------------------------------------------------------------------
HWSI12A1  HWSI12B1            0

**UPDATE IMSCON TYPE(RMTIMSCON) NAME(rmtinm) STOP(COMM)**

The new **UPDATE IMSCON TYPE(RMTIMSCON) NAME(rmtinm) STOP(COMM)** command stops communications to a remote IMS Connect system. Example 6-54 shows the output.

Example 6-54  Output from UPD IMSCON TYPE(RMTIMSCON) NAME(rmtinm) STOP(COMM)

Response for: UPDATE IMSCON TYPE(RMTIMSCON) NAME(HWSI12A1) STOP(COMM)
RmtImsCon MbrName            CC CCText
----------------------------------------------------------------------
HWSI12A1  HWSI12B1            0

**UPDATE IMSCON TYPE(SENDCLNT) NAME(cl) RMTIMSCON(rmt) STOP(COMM)**

The new **UPDATE IMSCON TYPE(SENDCLNT) NAME(clientid) RMTIMSCON(rmtinm) STOP(COMM)** command deactivates a TCP/IP sending client in IMS Connect. Example 6-55 shows the output.

Example 6-55  Output from UPD IMSCON TYPE (SENDCLNT) NAME(cl) RMTIMSCON(rmt) STOP(COMM)

Response for: UPDATE IMSCON TYPE(SENDCLNT) NAME(MSCCCAC6) RMTIMSCON(HWSI12A1) STOP(COMM)
SendClnt RmtImsCon MbrName            CC CCText
MSCCCAC6 HWSI12A1  HWSI12B1            0

6.2.5 Generic MSC name support

Figure 6-4 on page 184 shows four IMS systems in three separate IMSPlexes. Each physical path between two IMS systems has an **MSPLINK** macro and a corresponding MSC configuration statement in the local IMS connect to define the path within the IMSPlex. IMS1 and IMS2 are using shared queues. IMS3 and IMS4 have an MSC link to connect to the shared queues systems. To configure that link using TCP/IP MSC links, you need to use generic MSC support.
Additional IMS definitions

To connect the shared queues systems, you must define a new TCP/IP **MSPLINK** in the queue sharing IMS system. We already have the definitions in the original systems as shown in Example 6-22 on page 170 and Example 6-23 on page 171.

In the new queue sharing IMS system, we add the **MSPLINK**, **MSLINK**, and **MSNAME** as shown in Example 6-56. The existing definition remains unchanged.

**Example 6-56  IMS stage 1 macros for a TCP/IP MSC link (shared queues)**

<table>
<thead>
<tr>
<th>LINKCB</th>
<th>MSPLINK TYPE=TCPIP,NAME=I12B,SESSION=2,BUFSIZE=2048, X</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LCLICON=HWCF12C1, X</td>
</tr>
<tr>
<td></td>
<td>LCLPLKID=MSC2C2B</td>
</tr>
<tr>
<td>MSLINK</td>
<td>PARTNER=CB,MSPLINK=LINKCB</td>
</tr>
<tr>
<td>MSCBC</td>
<td>MSNAME SYSID=(102,103)</td>
</tr>
</tbody>
</table>

If this is the first time you add an MSC link with **TYPE=TCPIP**, you must run a stage1 or stage2 ALL or NUCLEUS system definition and restart IMS to make the MSC link available.

You must also update the IMS.PROCLIB(DFSDCxxx) member for both queue sharing IMS systems to define their generic IMS ID. This update requires an IMS WARM restart to activate the generic IMS ID. Example 6-57 shows the new specification.

**Example 6-57  Additional statement in IMS.PROCLIB(DFSDCxxx)**

```plaintext
GENIMSID=ACIM,
```

This generic IMS ID is only needed in the queue sharing systems. The /DISPLAY ACTIVE command shows the generic IMS ID when it is activated.
New IMS type-2 commands

Two new type-2 commands are available to control generic MSC links. No equivalent type-1 commands are available.

**UPDATE MSPLINK NAME(msplinkname | *) START(GENLOGON)**

The new *UPDATE MSPLINK NAME(...) START(GENLOGON)* command activates a generic MSC link. Example 6-58 shows the output.

Example 6-58  Output from the UPD MSPLINK NAME(*) START(GENLOGON) command

<table>
<thead>
<tr>
<th>MSPLink</th>
<th>MbrName</th>
<th>CC</th>
<th>CCText</th>
</tr>
</thead>
<tbody>
<tr>
<td>LINKBA</td>
<td>I12A</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>LINKCB</td>
<td>I12C</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

**UPDATE MSPLINK NAME(msplinkname | *) STOP(GENLOGON)**

The new *UPDATE MSPLINK NAME(...) STOP(GENLOGON)* command stops a generic MSC link. Example 6-59 shows the output.

Example 6-59  Output from the UPD MSPLINK NAME(*) STOP(GENLOGON) command

<table>
<thead>
<tr>
<th>MSPLink</th>
<th>MbrName</th>
<th>CC</th>
<th>CCText</th>
</tr>
</thead>
<tbody>
<tr>
<td>LINKBA</td>
<td>I12A</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>LINKCB</td>
<td>I12C</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Additional IMS Connect definitions

To support the new generic MSC links, we add a second MSC link to our existing configuration. Example 6-60 shows the modified definition for the link from I12A to I12B and the new definition for I12C to I12B.

Example 6-60  Modified MSC definitions in the local IMS Connect

```plaintext
MSC=(LCLPLKID=MSC2A2B,RMTPLKID=MSC2B2A,
    LCLIMS=I12A,RMTIMS=I12B,
    GENIMSID=ACIM,
    IMSPLEX=(MEMBER=HWSI12A1,TMEMBER=IM12X),
    RMTIMSCON=HWSI12B1)

MSC=(LCLPLKID=MSC2C2B,RMTPLKID=MSC2B2C,
    LCLIMS=I12C,RMTIMS=I12B,
    GENIMSID=ACIM,
    IMSPLEX=(MEMBER=HWSI12A1,TMEMBER=IM12X),
    RMTIMSCON=HWSI12B1)

RMTIMSCON=(ID=HWSI12B1,
           HOSTNAME=WTSC64.ITSO.IBM.COM,PORT=7202,
           AUTOCONN=Y,REVSOC=10)
```

The shared queues system that is running for I12A and I12C now has a generic IMS ID of ACIM. The one IMS Connect is used by both IMS systems.
The IMS Connect configuration for the remote system must also change for generic support. Example 6-61 shows that we have updated the MSC definition to use the generic IMS ID rather than a specific RMTIMS system.

Example 6-61  Modified MSC definitions in the remote IMS Connect

MSC=(LCLPLKID=MSC2B2A,RMTPLKID=MSC2A2B,
   LCLIMS=I12B,RMTIMS=ACIM,
   IMSPLEX=(MEMBER=HWSII2B1,TMEMBER=IM12X),
   RMTIMSCON=HWSI12A1)

RMTIMSCON=(ID=HWSI12A1,
   HOSTNAME=WTSC63.ITSO.IBM.COM,PORT=7102,
   AUTOCONN=Y,RESVSOC=10)

The RMTIMSCON definitions are unchanged but are included in both examples for clarity.

6.2.6  IMS Connect security using RACF PassTickets

The only security that you can use for an MSC link using TCP/IP is the PassTicket support for connection security. This security is implemented by defining USERID=xxx and APPL=yyy on the RMTIMSCON definitions in IMS Connect.

You must create PTKTDATA and APPL definitions in the RACF database. At the local system, you need PTKTDATA so that system can generate a PassTicket (Example 6-62).

Example 6-62  Local RACF definitions

SETROPTS CLASSACT(PTKTDATA)
SETROPTS RACLIST(PTKTDATA)
RDEFINE PTKTDATA APPLI12B SSIGNON(KEYSIGNED(E001193519561977)) UACC(N)
SETROPTS REFRESH RACLIST(PTKTDATA)

At the remote system, create PTKTDATA (used to verify the incoming PassTicket) and an APPL, and permit the remote user ID read access to the APPL (Example 6-63).

Example 6-63  Remote RACF definitions

DELUSER USER002
ADDUSER USER002 PASSWORD(PSWD0002) TSO(ACCTNUM(D1001) PROC(TPROC02))

SETROPTS CLASSACT(PTKTDATA)
SETROPTS RACLIST(PTKTDATA)
RDEFINE PTKTDATA APPLI12B SSIGNON(KEYSIGNED(E001193519561977)) UACC(N)
SETROPTS REFRESH RACLIST(PTKTDATA)

SETROPTS CLASSACT(APPL)
SETROPTS RACLIST(APPL)
RDEFINE APPL APPLI12B UACC(N)
PERMIT APPLI12B ACCESS(READ) CLASS(APPL) ID(USER002)
SETROPTS RACLIST(APPL) REFRESH
RLIST APPL APPLI12B AU

KEYMASKED value: The value used for KEYMASKED must be identical on both systems.
You must then update the **RMTIMSCON** definitions to use the remote USERID and APPL created at the remote system (Example 6-64).

**Example 6-64  Changes to the RMTIMSCON definition for security**

```
RMTIMSCON=(ID=HWSI12B1, 
  HOSTNAME=WTSC64.ITSO.IBM.COM,PORT=7202, 
  PERSISTENT=Y,RESVSOC=1, 
  USERID=USER002,APPL=APPLI12B)
```

### 6.2.7 Configuration summary for MSC using TCP/IP

Figure 6-5 on page 188 shows two IMS systems connected with a TCP/IP non-generic MSC link.

IMS1 has **MSPLINK** with the new parameters for TCP/IP. The NAME matches the **RMTIMS** parameter defined on the MSC definition in the local IMS Connect.

The **LCLICON** matches the **MEMBER** parameter of the **IMSPLEX** statement in the MSC definition in the local IMS Connect. The **LCLPLKID** matches the **LCLPLKID** parameter in the MSC definition.

The local IMS Connect has the MSC definition and the **RMTIMSCON** definition. On the MSC definition, the LCLIMS name must match the IMSID of the local IMS system. The **RMTIMS** defines the IMS ID of the remote system (or the generic name if GENIMSID is used). The **RMTIMSCON** name must match the ID parameter of the **RMTIMSCON** definition. On the **RMTIMSCON** definition you define the ID and the TCP/IP parameters so that you can find the remote system within the network.

IMS2 has **MSPLINK**, which points to NAME, LCLICON, and LCLPLKID.

The remote IMS Connect has the MSC definition with **RMTIMS**, **LCLIMS**, **LCLPLKID**, and **RMTIMSCON**. It also has the **RMTIMSCON** definition to define the other IMS Connect.

No connection security is defined on this link.
6.3 Other IMS Connect changes

The following minor changes have also been made to IMS and IMS Connect in version 12:

- IMS Connect Recorder Trace has new tracing points.
- IMS Connect can now use RACF user ID caching to reduce storage use and reduce the number of calls made to RACF.
- IMS Connect now supports RACF return codes and reason codes for security violations.
- With XML Converter Refresh, a module can be refreshed without restarting IMS Connect.
- Partial read status support helps with clients that have not completed sending data.
- Load modules for exits are packaged in addition to sample exits source.
- QUERY MSLINK STATISTICS changes for TCP/IP MSC links to allow additional statistics.

We provide a brief description in the following sections.

6.3.1 IMS Connect Recorder Trace

IMS 12 adds a new level of tracing to IMS Connect. It adds records for TCP/IP and MSC, and cross-system coupling facility (XCF) sends and receives at the trace points listed in Table 6-1.

Table 6-1  IMS Connect additional recorder trace entries

<table>
<thead>
<tr>
<th>Trace point (eyecatcher)</th>
<th>What is traced</th>
<th>Tracing method</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICONTR</td>
<td>TCP/IP Receive</td>
<td>BPE</td>
</tr>
<tr>
<td>ICONTS</td>
<td>TCP/IP Send</td>
<td>BPE</td>
</tr>
</tbody>
</table>
Of the new records, those capturing the entire message, and therefore generating a lot of data, are only recorded in the Base Primitive Environment (BPE) External trace. They have the following IDs (eyecatchers):

- ICONTR TCP/IP Receive
- ICONTS TCP/IP Send
- ICONIR IMS OTMA Receive
- ICONIS IMS OTMA Send

The new records generating small amounts of data are captured by using the existing, old method. They have the IDs (eyecatchers):

- ICONMS MSC send
- ICONMR MSC receive
- ICONRR Remote IMS Connect to local IMS Connect

The existing trace points are still recorded in both the new recorder trace and the existing HWSRCORD trace:

- ICONRC: User Msg Exit Receive
- ICONSN: User Msg Exit XMIT

The support for a BPE-based recorder trace was included in IMS 11.

**Tip:** Remove the existing recorder trace, and replace it with the new BPE-based trace.

### Defining the new trace in IMS Connect

Before you can use the new IMS Connect recorder trace, complete these steps:

1. Update IMS Connect JCL to remove the HWSRCORD data set (Example 6-65).

   **Example 6-65  Remove the old recorder trace**

   ```
   //IM12AHW1 PROC RGN=0M,TME=1440,SOUT=*,
   // BPECFG=BPEHWS2A,HWSCFG=HWCF12A1
   ...
   //HWSRCORD DD DSN=IMS12Q.IMS12A.HWSRCDR,DISP=SHR /* Not used */
   ```

2. Update the BPE configuration to add a TRCLEV and EXTTRACE definition to define the GDG base for the new recorder trace (Example 6-66).

   **Example 6-66  Updated IMS.PROCLIB(BPEHWSxx) member**

   ```
   TRCLEV=(RCTR,MEDIUM,Hws,EXTERNAL=YES)
   #
   # DEFINITION FOR NEW STYLE RECORDER TRACE
   #
   EXTTRACE(
   ```
3. Define the GDG base data set by using **IDCAMS** or option 6 of ISPF (Example 6-67).

**Example 6-67  Defining the GDG base data set**

```bash
//STEP1  EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=* 
//SYSIN DD *
DEFINE GDG( 
   NAME('IMS12Q.IMS12A.HWS.RCTR') 
   LIMIT(10) 
   SCRATCH 
) 
```

4. Restart IMS Connect, which then creates a recorder trace data set.

**New commands**

When you want to activate the recorder trace, you must issue a modify command (Example 6-68) that is actioned by BPE. Regardless of the specification in the BPE configuration, the recorder trace is always inactive when IMS Connect starts. It can only be activated by an explicit **UPDATE TRACETABLE** command.

**Example 6-68  BPE command to activate the recorder trace**

```bash
F IM12AHW1,UPDATE TRACETABLE NAME(RCTR) OWNER(HWS) LEVEL(HIGH) EXTERNAL(YES) 
```

To terminate the recorder trace, you issue another BPE **UPDATE** command (Example 6-69).

**Example 6-69  BPE command to terminate the recorder trace**

```bash
F IM12AHW1,UPDATE TRACETABLE NAME(RCTR) OWNER(HWS) LEVEL(NONE) EXTERNAL(NO) 
```

### 6.3.2 RACF user ID caching

User ID caching is a new function in IMS 12 to reduce IMS Connect storage usage and reduce the number of calls made to RACF to check user IDs. If the same user ID connects from more than one client or on more than one port, IMS Connect searches its cache to find security credentials (assuming they have not expired due to aging). If the user ID is not found, only then does it call RACF.

**New configuration definition**

Example 6-70 on page 191 shows the new specification for UIDCACHE and UIDAGE. UIDAGE gives better control for when IMS Connect removes credentials from storage and calls RACF to renew them. UIDCACHE can be controlled by an external command. UIDAGE can only be controlled by updating the HWSCFG member and restarting IMS Connect.
Chapter 6. Connectivity enhancements

Example 6-70  IMS Connect configuration member

HWS=(ID=HWSI12A1,
     PSWDMC=R,RACF=Y,
     UIDCACHE=Y,UIDAGE=500,
     RRS=Y,XIBAREA=50)

Example 6-71 shows the QUERY MEMBER (or equivalent VIEWHWS) command with the current values for UIDCACHE and UIDAGE.

Example 6-71  Output from the QUERY MEMBER command

F IMI2DHW1,QUERY MEMBER TYPE(IMSCON) SHOW(ALL)
HWSC0001I HWS ID=HWSI12D1 RACF=N PSWDMC=R
HWSC0001I UIDCACHE=Y UIDAGE=500
HWSC0001I MAXSOC=50 TIMEOUT=0
HWSC0001I NUMSOC=5 WARNSOC=80% WARNINC=5%
HWSC0001I RRS=Y STATUS=ACTIVE
HWSC0001I VERSION=V12 IP-ADDRESS=009.012.006.070
HWSC0001I SUPER MEMBER NAME= CM0 ACK TOQ=

New IMS Connect commands

Three new commands control user ID caching. Table 6-2 shows the commands to control user ID caching and to refresh the in-storage data for a user ID. No support is available for using a generic user ID on the refresh command.

<table>
<thead>
<tr>
<th>Type-2 command</th>
<th>WTOR reply</th>
<th>Modify command</th>
</tr>
</thead>
<tbody>
<tr>
<td>UPD IMSCON TYPE(CONFIG) SET(UIDCACHE(ON))</td>
<td>SETUIDC ON</td>
<td>UPDATE MEMBER TYPE(IMSCON) SET(UIDCACHE(ON))</td>
</tr>
<tr>
<td>UPD IMSCON TYPE(CONFIG) SET(UIDCACHE(OFF))</td>
<td>SETUIDC OFF</td>
<td>UPDATE MEMBER TYPE(IMSCON) SET(UIDCACHE(OFF))</td>
</tr>
<tr>
<td>UPDATE IMSCON TYPE(RACFUID) NAME(userid) OPTION(REFRESH)</td>
<td>REFRESH RACFUID NAME(userid)</td>
<td>UPDATE RACFUID NAME(userid) OPTION(REFRESH)</td>
</tr>
</tbody>
</table>

6.3.3 RACF return codes

New support is in the IMS Connect exits to pass any bad return code from RACF back to the client. This process is done by extending the header for the *REQSTS* RSM when it is returned to the client. Example 6-72 shows the new RACFRC field in the RSM as mapped by IMS12Q.MACLIB(HWSIMSCB).

Example 6-72  New layout for RSM

<table>
<thead>
<tr>
<th>RSMMask</th>
<th>DSECT</th>
<th>Request status message dsect</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSM_Len</td>
<td>DS H</td>
<td>Length of RSM</td>
</tr>
<tr>
<td>RSM_FLGS</td>
<td>DS 0H</td>
<td>RSM FLAG BYTES</td>
</tr>
<tr>
<td>RSM_FLG1</td>
<td>DS X</td>
<td>FLAG BYTE</td>
</tr>
<tr>
<td>RSM_AMSG</td>
<td>EQU X'80'</td>
<td>ASYNCH MSG Q'D IN IMS</td>
</tr>
<tr>
<td>RSM_CONV</td>
<td>EQU X'40'</td>
<td>CONVERSATIONAL OUTPUT</td>
</tr>
<tr>
<td>RSM_ACK_NAK</td>
<td>EQU X'20'</td>
<td>ACK/NAK REQUIRED</td>
</tr>
<tr>
<td>RSM_RACFRC</td>
<td>DS 0X</td>
<td>RETURN CODE FROM RACF</td>
</tr>
<tr>
<td>RSM_OTMARSN</td>
<td>DS X</td>
<td>REASON CODE FROM OTMA</td>
</tr>
</tbody>
</table>
The REXX application was amended to send in a bad password. The HWCSCFG was updated to have RACF=Y. In addition, the REXX error handler was updated to display the header when IMS Connect returns a *REQSTS* RSM.

Example 6-73 shows the new REXX code used to handle the *REQSTS*. RSM_RetCode and RSM_RsnCod were already being interpreted.

**Example 6-73  REXX handling for *REQSTS* RSM**

```rexx
if pos('*REQSTS*',page) > 0 then do
  parse var page rsm_header '*REQSTS*' +8 retc +4 rsn
  say '*REQSTS*' "RC:" c2d(retc) 'x', "RSN:" c2d(rsn) 'x'
  say "HDR:" c2x(rsm_header)
  parse var rsm_header llll +4 rsm_len +2 rsm_FLG1 +1 rsm_RACFRC
  say "LLLL:" c2d(llll) 'x'
  say "RSM_Len:" c2d(rsm_len) 'x'
  say "RSM_FLG1:" c2d(rsm_FLG1) 'x'
  say "RSM_RACFRC:" c2d(rsm_RACFRC) 'x'
end
```

Example 6-74 shows that, after the RACF error, IMS Connect returns LLLL=24 (x'00000018') because HWSSMPL1, RSM_Len=20 (x'14'), RSM_FLG1=x'00', and RSM_RACFRC=x'08' are used.

**Example 6-74  Output from IMS Connect**

```
*REQSTS* RC: 8 '8'x RSN: 40 '28'x
HDR: 0000001800000000
LLLL: 24 '18'x
RSM_Len: 20 '14'x
RSM_FLG1: 0 '0'x
RSM_RACFRC: 8 '8'x
```

### 6.3.4 XML Converter Refresh

IMS 12 adds a new function so that an XML converter module can be refreshed without restarting IMS Connect. One new command can be used as a WTOR reply, a modify command, or from an IMS SPOC to refresh an XML converter module (Table 6-3).

**Table 6-3  XML converter refresh command**

<table>
<thead>
<tr>
<th>Type-2 command</th>
<th>WTOR reply</th>
<th>Modify command</th>
</tr>
</thead>
<tbody>
<tr>
<td>UPD IMSCON TYPE(CONVERTER) NAME(modname) OPTION(REFRESH)</td>
<td>REFRESH CONVERTER NAME(modname)</td>
<td>UPDATE CONVERTER NAME(modname) OPTION(REFRESH)</td>
</tr>
</tbody>
</table>
6.3.5 Partial read status

Partial read is a TCP/IP status where the client sending data has not sent the whole message that the listener is expecting. This condition can lead to unexpected hangs in IMS Connect or in the client process.

IMS Connect in IMS 12 now has a function to deal with clients that have not completed their send of data or have the wrong value for the length passed to the IMS Connect exit. You can see which clients are waiting in with a partial read.

Example 6-75 shows the output from a VIEWPORT command with a client using HWSSMPL1 that has an invalid full length value (LLLL). The client is hanging waiting for IMS Connect. IMS Connect is waiting for the client to send the missing data.

Example 6-75 Output from the VIEWPORT command with a partial read client

```
R 822,VIEWPORT 7400
IEE6001I REPLY TO 822 IS;VIEWPORT 7400
HWSC0001 PORT=7400 STATUS=ACTIVE KEEPAV=0 NUMSOC=2 EDIT=
TIMEOUT=0
HWSC0001I CLIENTID USERID TRancode DATAstore STATUS
SECOND CLNTPORT IP-ADDRESS APSB-TOKEN
HWSC0001I DELDUMMY READ
386 9180 127.0.0.0.001
HWSC0001I TOTAL CLIENTS=1 RECv=0 READ=1 CONN=0 XMIT=0 OTHER=0
```

Example 6-76 shows the same status by using the QUERY IMSCON command from the IMS batch SPOC. You can also use the QRY IMSCON TYPE(PORT) NAME(7400) SHOW(STATUS,CLIENT) command to reduce the response to just the information you need.

Example 6-76 Output from the QRY IMSCON TYPE(PORT) command with a partial read client

```
Response for: QRY IMSCON TYPE(PORT) NAME(7400) SHOW(ALL)
Port MbrName CC CCText KeepAv NumSoc Edit TimeOut Status TotClnts TotRecv TotRead TotConn TotXmit TotOther ClientID UserID Trancode DataStore ApsbToken
7400 HWSI12D1 0 0 2 0 ACTIVE 1 0 1 0 0 0
7400 HWSI12D1 0 DELDUMMY
READ 262 9180 127.0.0.1
```

Example 6-77 shows usage of the STOPCLNT command to terminate the connection from the remote client and break the hang condition.

Example 6-77 The STOPCLNT command

```
R 832,STOPCLNT 7400 DELDUMMY
IEE6001 REPLY TO 832 IS;STOPCLNT 7400 DELDUMMY
HWSS0761I TCPIP COMMUNICATION WITH CLIENT=7400 DELDUMMY STOPPED;
M=SCCM
```
Example 6-78 shows the same client being terminated by using the new **UPDATE IMSCON TYPE(CLIENT)** command.

**Example 6-78   The UPD IMSCON TYPE(CLIENT) command to terminate a partial read client**

<table>
<thead>
<tr>
<th>Response for: UPD IMSCON TYPE(CLIENT) NAME(DELDUMMY) PORT(7400) STOP(COMM)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ClientID</td>
</tr>
<tr>
<td>----------</td>
</tr>
<tr>
<td>DELDUMMY</td>
</tr>
</tbody>
</table>

### 6.3.6 Load modules for exits

IBM supplies samples for some IMS Connect exits, including HWSUINIT, HWSJAVA0, HWSSMPL0, and HWSSMPL1. These exits are working examples, and many customers use the IBM supplied exits without modifications.

In IMS 12, the exits are being repackaged as load modules and sample source. Users who want to modify the IBM supplied samples can still do so. Customer who assemble and bind the current samples without modification can remove a task from their migration and implementation plans.

If you need to assemble the sample source for any reason, then the JCL used to for this task must include IMS.SDFSMAC, SYS1.MACLIB, and SYS1.MODGEN. Otherwise, the assembly will fail.

### 6.3.7 QUERY MSLINK STATISTICS changes for TCP/IP MSC links

IMS 12 has added new statistics to the output from a **QRY MSLINK SHOW(STATISTICS)** command. The new values (shown in Table 6-4) are added for TCP/IP MSC links. They include the high, low, and total send I/O times for the local and remote SCI, IMS Connect, and TCP/IP components.

#### Table 6-4 New TCP/IP MSC link statistics

<table>
<thead>
<tr>
<th>Short name</th>
<th>Long name</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>HLISIOT</td>
<td>HiLocIconSendIOTime</td>
<td>The longest interval of time that the local IMS Connect instance is required to process a message from SCI and send it to TCP/IP.</td>
</tr>
<tr>
<td>HLSSIOT</td>
<td>HiLocSciSendIOTime</td>
<td>The longest interval of time that the local SCI instance is required to process a message from IMS and send it to the local IMS Connect.</td>
</tr>
<tr>
<td>HRISIOT</td>
<td>HiRmtIconSendIOTime</td>
<td>The longest interval of time that the remote IMS Connect instance is required to process a message from TCP/IP and send it to the remote SCI.</td>
</tr>
<tr>
<td>HRSSIOT</td>
<td>HiRmtSciSendIOTime</td>
<td>The longest interval of time that the remote SCI instance is required to process a message from the remote IMS Connect instance and send it to the remote IMS system.</td>
</tr>
<tr>
<td>HTCSIOT</td>
<td>HiTcpipSendIOTime</td>
<td>The longest interval of time that a message is required to travel from the local IMS Connect instance to the remote IMS Connect instance on the TCP/IP network.</td>
</tr>
<tr>
<td>LLISIOT</td>
<td>LowLocIconSendIOTime</td>
<td>The shortest interval of time that the local IMS Connect instance is required to process a message from SCI and send it to TCP/IP.</td>
</tr>
<tr>
<td>LRISIOT</td>
<td>LowRmtIconSendIOTime</td>
<td>The shortest interval of time that the remote IMS Connect instance is required to process a message from TCP/IP and send it to the remote SCI.</td>
</tr>
</tbody>
</table>
Chapter 6. Connectivity enhancements

6.4 IMS Connect type-2 Single Point of Control commands

IMS Connect is enhanced in IMS 12 to support a new type-2 command interface from the IMS SPOC. The following groups of commands are added:

- Display commands
- Start commands
- Stop and close commands
- Set, reset, and refresh commands

6.4.1 Display commands

Table 6-5 shows the new IMS Connect type-2 SPOC commands and the equivalent WTOR reply or type-2 modify command.

<table>
<thead>
<tr>
<th>Short name</th>
<th>Long name</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>LRSSIOT</td>
<td>LowRmtSciSendIOTime</td>
<td>The shortest interval of time that the remote SCI instance is required to process a message from the remote IMS Connect instance and send it to the remote IMS system.</td>
</tr>
<tr>
<td>LTCSIOT</td>
<td>LowTcpipSendIOTime</td>
<td>The shortest interval of time that a message is required to travel from the local IMS Connect instance to the remote IMS Connect instance on the TCP/IP network.</td>
</tr>
<tr>
<td>TLISIOT</td>
<td>TotLociconSendIOTime</td>
<td>The total amount of time that the local IMS Connect instance is required to process all messages from SCI and send them to TCP/IP.</td>
</tr>
<tr>
<td>TLSSIOT</td>
<td>TotLocSciSendIOTime</td>
<td>The total amount of time that the local SCI instance is required to process all messages from IMS and send them to the local IMS Connect.</td>
</tr>
<tr>
<td>TRISIOT</td>
<td>TotRmticonSendIOTime</td>
<td>The total amount of time that the remote IMS Connect instance required to process all messages from TCP/IP and send them to the remote SCI.</td>
</tr>
<tr>
<td>TRSSIOT</td>
<td>TotRmtSciSendIOTime</td>
<td>The total amount of time that the remote SCI instance is required to process all messages from the remote IMS Connect instance and send them to the remote IMS system.</td>
</tr>
<tr>
<td>TTCSIOT</td>
<td>TotTcpipSendIOTime</td>
<td>The sum total of the amount of time that all messages is required to travel from the local IMS Connect instance to the remote IMS Connect instance on the TCP/IP network.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type-2 command</th>
<th>WTOR reply</th>
<th>Modify command</th>
</tr>
</thead>
<tbody>
<tr>
<td>QUERY IMSCON TYPE(ALIAS) NAME(*) SHOW(ALL</td>
<td>showparm)</td>
<td>VIEWIA ALL</td>
</tr>
<tr>
<td>QUERY IMSCON TYPE(ALIAS) NAME(alias) SHOW(ALL</td>
<td>showparm)</td>
<td>VIEWIA alias</td>
</tr>
<tr>
<td>QUERY IMSCON TYPE(ALIAS) NAME(alias) ODBM(odbmname)</td>
<td>VIEWIA alias odbmname</td>
<td>Not applicable</td>
</tr>
<tr>
<td>QUERY IMSCON TYPE(CONFIG) SHOW(ALL</td>
<td>showparm)</td>
<td>VIEWHWS</td>
</tr>
<tr>
<td>QUERY IMSCON TYPE(DATASTORE) NAME(*) SHOW(ALL</td>
<td>showparm)</td>
<td>VIEWDS ALL</td>
</tr>
</tbody>
</table>
### 6.4.2 Start commands

Table 6-6 shows the new IMS Connect type-2 SPOC commands and the equivalent WTOR reply or type-2 modify command.

**Table 6-6  IMS Connect type-2 start commands**

<table>
<thead>
<tr>
<th>Type-2 command</th>
<th>WTOR reply</th>
<th>Modify command</th>
</tr>
</thead>
<tbody>
<tr>
<td>QUERY IMSCON TYPE(DATASTORE) NAME(datastore) SHOW(ALL</td>
<td>viewparm)</td>
<td>VIEWDS datastore</td>
</tr>
<tr>
<td>QUERY IMSCON TYPE(IMSPLEX) NAME(*) SHOW(ALL</td>
<td>showparm)</td>
<td>VIEWIP ALL</td>
</tr>
<tr>
<td>QUERY IMSCON TYPE(IMSPLEX) NAME(IMSplexname) SHOW(ALL</td>
<td>showparm)</td>
<td>VIEWIP IMSplexname</td>
</tr>
<tr>
<td>QUERY IMSCON TYPE(MSC) NAME(*) SHOW(ALL</td>
<td>showparm)</td>
<td>VIEWMSC ALL</td>
</tr>
<tr>
<td>QUERY IMSCON TYPE(MSC) NAME(mscid) SHOW(ALL</td>
<td>showparm)</td>
<td>VIEWMSC mscid</td>
</tr>
<tr>
<td>QUERY IMSCON TYPE(PORT) NAME(*) SHOW(ALL</td>
<td>showparm)</td>
<td>VIEWPORT ALL</td>
</tr>
<tr>
<td>QUERY IMSCON TYPE(PORT) NAME(portname) SHOW(ALL</td>
<td>showparm)</td>
<td>VIEWPORT portname</td>
</tr>
<tr>
<td>QUERY IMSCON TYPE(PORT) NAME(LOCAL) SHOW(ALL</td>
<td>showparm)</td>
<td>VIEWPORT LOCAL</td>
</tr>
<tr>
<td>QUERY IMSCON TYPE(RMTIMSCON) NAME(*) SHOW(ALL</td>
<td>showparm)</td>
<td>VIEWRMT ALL</td>
</tr>
<tr>
<td>QUERY IMSCON TYPE(RMTIMSCON) NAME(rmtimscon) SHOW(ALL</td>
<td>showparm)</td>
<td>VIEWRMT rmtimscon</td>
</tr>
<tr>
<td>QUERY IMSCON TYPE(UOR) NAME(*) SHOW(ALL</td>
<td>showparm)</td>
<td>VIEWUOR ALL</td>
</tr>
<tr>
<td>QUERY IMSCON TYPE(UOR) NAME(uorid) SHOW(ALL</td>
<td>showparm)</td>
<td>VIEWUOR uorid</td>
</tr>
</tbody>
</table>
6.4.3 Stop and close commands

Table 6-7 shows the new IMS Connect type-2 SPOC commands and the equivalent WTOR reply or type-2 modify command.

<table>
<thead>
<tr>
<th>Type-2 command</th>
<th>WTOR reply</th>
<th>Modify command</th>
</tr>
</thead>
<tbody>
<tr>
<td>UPDATE IMSCON TYPE(ODBM) NAME(odbmname) START(COMM)</td>
<td>STARTOD odbmname</td>
<td>Not applicable</td>
</tr>
<tr>
<td>UPDATE IMSCON TYPE(RMTIMSCON) NAME(rmtimscon) START(COMM)</td>
<td>STARTRMT rmtimscon</td>
<td>UPDATE RMTIMSCON NAME(rmtimscon) START(COMM)</td>
</tr>
<tr>
<td>UPDATE IMSCON TYPE(PORT) NAME(portname) START(COMM)</td>
<td>OPENPORT portname or STARTPT portname</td>
<td>UPDATE PORT NAME(portname) START(COMM)</td>
</tr>
</tbody>
</table>

Table 6-7  IMS Connect type-2 stop or close commands

<table>
<thead>
<tr>
<th>Type-2 command</th>
<th>WTOR reply</th>
<th>Modify command</th>
</tr>
</thead>
<tbody>
<tr>
<td>UPDATE IMSCON TYPE(ALIAS) NAME(alias) ODBM(odbmname) STOP(COMM)</td>
<td>STOPIA alias odbmname</td>
<td>Not applicable</td>
</tr>
<tr>
<td>UPDATE IMSCON TYPE(CLIENT) NAME(client_name) PORT(portname) STOP(COMM)</td>
<td>STOPCLNT portname clientid</td>
<td>DELETE PORT NAME(portName) CLIENT(clientName)</td>
</tr>
<tr>
<td>UPDATE IMSCON TYPE(CONFIG) SHUTDOWN(COMM)</td>
<td>CLOSEHWS</td>
<td>SHUTDOWN MEMBER</td>
</tr>
<tr>
<td>UPDATE IMSCON TYPE(CONFIG) SHUTDOWN(COMM) OPTION(FORCE)</td>
<td>CLOSEHWS FORCE</td>
<td>SHUTDOWN MEMBER OPTION(FORCE)</td>
</tr>
<tr>
<td>UPDATE IMSCON TYPE(CONFIG) SHUTDOWN(COMM) OPTION(QUIESCE)</td>
<td>CLOSEHWS QUIESCE</td>
<td>SHUTDOWN MEMBER OPTION(QUIESCE)</td>
</tr>
<tr>
<td>UPDATE IMSCON TYPE(DATASTORE) NAME(datastore) STOP(COMM)</td>
<td>STOPDS datastore</td>
<td>UPDATE DATASTORE NAME(datastore) STOP(COMM)</td>
</tr>
<tr>
<td>UPDATE IMSCON TYPE(IMSPLEX) NAME(IMSplexname) STOP(COMM)</td>
<td>STOPIP IMSplexname</td>
<td>UPDATE IMSPLEX NAME(IMSplexname) STOP(COMM)</td>
</tr>
<tr>
<td>UPDATE IMSCON TYPE(LINK) NAME(linkname) STOP(COMM)</td>
<td>STOPLINK linkname</td>
<td>DELETE LINK NAME(linkname)</td>
</tr>
<tr>
<td>UPDATE IMSCON TYPE(MSC) NAME(lclplkid) STOP(COMM)</td>
<td>STOPMSC lclplkid</td>
<td>UPDATE MSC NAME(lclplkid) STOP(COMM)</td>
</tr>
<tr>
<td>UPDATE IMSCON TYPE(ODBM) NAME(odbmname) STOP(COMM)</td>
<td>STOPOD odbmname</td>
<td>Not applicable</td>
</tr>
<tr>
<td>UPDATE IMSCON TYPE(RMTIMSCON) NAME(rmtimscon) STOP(COMM)</td>
<td>STOPRMT rmtimscon</td>
<td>UPDATE RMTIMSCON NAME(rmtimscon) STOP(COMM)</td>
</tr>
<tr>
<td>UPDATE IMSCON TYPE(SENDCLNT) NAME(sendclient_name) RMTIMSCON(rmtimscon) STOP(COMM)</td>
<td>STOPSCN rmtimscon sendclient</td>
<td>DELETE RMTIMSCON NAME(rmtimscon) SENDCLNT(clientid)</td>
</tr>
<tr>
<td>UPDATE IMSCON TYPE(PORT) NAME(portname) STOP(COMM)</td>
<td>STOPPORT portname</td>
<td>UPDATE PORT NAME(portname) STOP(COMM)</td>
</tr>
<tr>
<td>UPDATE IMSCON TYPE(CONFIG) STOP(RECORDER)</td>
<td>RECORDER CLOSE</td>
<td>UPDATE MEMBER TYPE(IMSCON) STOP(TRACE)</td>
</tr>
</tbody>
</table>
### 6.4.4 Set, reset, and refresh commands

Table 6-8 shows the new IMS Connect type-2 SPOC commands and the equivalent WTOR reply or type-2 modify command.

**Table 6-8  IMS Connect type-2 set, reset, and refresh commands**

<table>
<thead>
<tr>
<th>Type-2 command</th>
<th>WTOR reply</th>
<th>Modify command</th>
</tr>
</thead>
<tbody>
<tr>
<td>UPDATE IMSCON TYPE(CONFIG) SET(AUTO(ON))</td>
<td>SETAUTO YES</td>
<td>UPDATE MEMBER TYPE(IMSCON) SET(AUTO(ON))</td>
</tr>
<tr>
<td>UPDATE IMSCON TYPE(CONFIG) SET(AUTO(OFF))</td>
<td>SETAUTO NO</td>
<td>UPDATE MEMBER TYPE(IMSCON) SET(AUTO(OFF))</td>
</tr>
<tr>
<td>UPDATE IMSCON TYPE(CONFIG) SET(PSWDMC(ON))</td>
<td>SETPWMC ON</td>
<td>UPDATE MEMBER TYPE(IMSCON) SET(PSWDMC(ON))</td>
</tr>
<tr>
<td>UPDATE IMSCON TYPE(CONFIG) SET(PSWDMC(OFF))</td>
<td>SETPWMC OFF</td>
<td>UPDATE MEMBER TYPE(IMSCON) SET(PSWDMC(OFF))</td>
</tr>
<tr>
<td>UPDATE IMSCON TYPE(CONFIG) SET(EIT(OFF))</td>
<td>SETEIT OFF</td>
<td>UPDATE MEMBER TYPE(IMSCON) SET(EIT(OFF))</td>
</tr>
<tr>
<td>UPDATE IMSCON TYPE(CONFIG) SET(RACF(ON))</td>
<td>SETRACF ON</td>
<td>UPDATE MEMBER TYPE(IMSCON) SET(RACF(ON))</td>
</tr>
<tr>
<td>UPDATE IMSCON TYPE(CONFIG) SET(RACF(OFF))</td>
<td>SETRACF OFF</td>
<td>UPDATE MEMBER TYPE(IMSCON) SET(RACF(OFF))</td>
</tr>
<tr>
<td>UPDATE IMSCON TYPE CONFIG SET(RRS(ON))</td>
<td>SETRRS ON</td>
<td>Not applicable</td>
</tr>
<tr>
<td>UPDATE IMSCON TYPE CONFIG SET(RRS(OFF))</td>
<td>SETRRS OFF</td>
<td>Not applicable</td>
</tr>
<tr>
<td>UPDATE IMSCON TYPE(CONFIG) SET(UIDCACHE(ON))</td>
<td>SETUIDC ON</td>
<td>UPDATE MEMBER TYPE(IMSCON) SET(UIDCACHE(ON))</td>
</tr>
<tr>
<td>UPDATE IMSCON TYPE(CONFIG) SET(UIDCACHE(OFF))</td>
<td>SETUIDC OFF</td>
<td>UPDATE MEMBER TYPE(IMSCON) SET(UIDCACHE(OFF))</td>
</tr>
<tr>
<td>UPDATE IMSCON TYPE(CONVERTER) NAME(converter_name) OPTION(REFRESH)</td>
<td>REFRESH CONVERTER NAME(converter_name)</td>
<td>UPDATE CONVERTER NAME(converter_name) OPTION(REFRESH)</td>
</tr>
<tr>
<td>UPDATE IMSCON TYPE(RACFUID) NAME(userid) OPTION(REFRESH)</td>
<td>REFRESH RACFUID NAME(userid)</td>
<td>UPDATE RACFUID NAME(userid) OPTION(REFRESH)</td>
</tr>
</tbody>
</table>
IMS repository

The IMS repository is a new capability in IBM Information Management System (IMS) 12. This repository stores various types of data related to resource definitions and descriptors in a centralized store that is accessible to all IMS systems in the IMSplex.

Dynamic resource definition (DRD) exploits this new function by using the IMSRSC type of repository to simplify management of MODBLKS resources among multiple IMS systems. DRD also maintains the resource definitions of all IMS systems in a single centralized store. The IMSRSC repository is a strategic alternative to the resource definition data set (RDDS).

This chapter includes the following sections:

- Introducing and explaining the value of the repository
- Components of the repository
- Repository Server
- Repository data sets
- Common Service Layer
- Batch utilities
- Migrating to the repository
- IMSRSC repository access
- Using the repository in business operations
- Security considerations
- DRD user interface enhancements
- IVP enhancements for repository
- Value of using the IMS repository with DRD
7.1 Introducing and explaining the value of the repository

Users of IMS DRD can now use the repository to manage MODBLKS resources in a single centralized location. From these centralized data sets, the resource definitions can be stored and retrieved by different IMS systems existing together in an IMSplex. They can also be shared among the systems and be dynamically updated. The main goal of the IMS repository is to simplify resource management, easing the overhead involved in making and keeping track of definitional changes among IMS systems.

If you are already using DRD with separate RDDS for each IMS system, you can perform a few simple migration steps to begin using the repository instead. This will eliminate the need to manually coordinate and manage several RDDSs across different IMS systems. If you have never implemented DRD, you can use the definitions that exist in your MODBLKS data set as a starting point, eventually porting them to the new repository.

Similar to the RDDS, the IMSRSC repository contains MODBLKS definitions of the following resources:

- Databases
- Database descriptors
- Programs
- Program descriptors
- Routing codes
- Routing code descriptors
- Transactions
- Transaction descriptors

Implementation of the repository is in alignment with the future direction of IMS in that it provides a simplified, more dynamic method of managing resource definitions. Using DRD with the repository provides the highest availability for your IMS system MODBLKS resources. You no longer need to perform sysgen or online change to manage these resources. The repository centralizes all of these definitions in a single location.

7.2 Components of the repository

Several components work together to facilitate use of the repository. This section explores these components, which include the IMSRSC repository that contains stored resource definitions, the RS, RS catalog repository, required Common Service Layer (CSL) elements, and the utilities that interact with the IMSRSC repository itself. Throughout this section, as each component is explained, see Figure 7-1 on page 201 to help you understand how all of these moving parts work together.
7.3 Repository Server

The user repository that contains MODBLKS stored resource definitions is managed by a new address space called the Repository Server (RS). The RS, in turn, works with the RM address space whenever access to the user repository is requested. It also has its own repository called the RS catalog repository, which is explained in 7.4.1, “Repository Server catalog repository” on page 202. A single master RS manages access to the repository data sets, but several subordinate Repository Servers can take over for the master RS if it fails. The master RS is the Repository Server that gains access to the user repository first. It is required to have one master RS per IMSplex, residing on any LPAR, even if it is on a different LPAR than the one on which an IMS system is running. Having a subordinate Repository Servers is optional but is recommended for availability.

The Repository Server is responsible for managing registrations and connections to the repository. It ensures data integrity within the repository, and restricts access to the repository so that only authorized users can retrieve data from it. The RS also has an audit trail capability which you can set up to track certain events that occur when attempts to access the repository occur. Finally, the RS provides a tracing capability that uses the Base Primitive Environment (BPE), because it is based on this component.

As shown in Figure 7-1, the RS communicates with the Resource Manager component of the Common Service Layer using z/OS cross-system coupling facility (XCF) services to process incoming requests from RM and from the batch ADMIN utility. The RM utilities can be used to populate the repository and to read from the repository. The RM utilities are useful in migration and fallback because they work with both the RDDS and repository. The batch ADMIN utility performs administration tasks for both the user and the RS catalog repository.
For more information about these sets of utilities, see 7.8.2, “Offline access through RM utilities” on page 228, and “Batch ADMIN commands” on page 241.

7.4 Repository data sets

When you use DRD with the repository, two different types of repositories are used. One type is called the Repository Server catalog repository, which manages the information contained in the user repositories. The other type is called the user repository. The IMSRSC repository is a user repository that contains the MODBLKS resource definitions associated with the IMS systems.

Both types of repositories consist of two pairs of VSAM key-sequenced data sets (KSDS): a primary repository data set pair and a secondary repository data set pair. Each data set pair is made up of an index data set that contains all of the search fields for members that exist in the repository (including the member name). Each data set pair is also made up of a member data set that contains the member data that is indexed by the index data set. In addition to its primary and secondary repository data sets, the user repository can have a spare repository data set for availability.

**RS catalog repository:** The RS catalog repository does not have spare capability.

The following sections explore these two types of repositories further, starting with the RS catalog repository followed by the IMSRSC repository.

7.4.1 Repository Server catalog repository

The Repository Server has its own repository, which maintains the user repositories that contain the MODBLKS stored resource definitions, establishing their name references. The information contained in the RS catalog repository includes user repository names, their status (such as whether they are currently started or stopped), and the date they were last updated (with the identity of the updater). The RS is the only address space that can communicate with its catalog repository, and it does so when administrative tasks are performed such as adding a user repository name to the RS catalog repository or updating an existing one.

The catalog repository must be defined before any user repositories are defined, and it will consist of two pairs of VSAM key sequenced data sets: a primary pair and a secondary or duplex pair.

7.4.2 IMSRSC repository data sets

The IMSRSC repository is shared among multiple IMS systems within an Ites the need to manage individual Resource Definition Data Sets (RDDSs) for each IMS system, simplifying the resource management process.

Like the RS catalog repository, the IMSRSC repository consists of a primary and secondary pair of VSAM KSDS repository data sets, but it also has an optional spare pair. If a repository write failure occurs on the primary or secondary data sets, the data set that still contains valid data is copied to the spare data set and the failed copy is marked as discarded. In the event of a read error, the remaining valid data set is read and no copy to spare is initiated. For information about the recovery process, see “Recovery activities” on page 257.
To distinguish between these different types of repository data sets, a “state” is assigned to each one. The primary repository data set is referred to as COPY1, the secondary repository data set is referred to as COPY2, and the spare repository data set is referred to as SPARE. If a write error occurs on the primary or secondary repository data set, its state is changed from either COPY1 or COPY2 to DISCARD. You can display the status for each of the repository data sets with a command as explained in 7.8.4, “Direct repository access using z/OS modify interface” on page 230.

You can have more than one IMSRSC repository per Repository Server, for example for test and production IMS systems. However, it is recommended to have one Repository Server per IMSplex so typically you will have one IMSRSC repository per Repository Server per IMSplex.

So far, the IMSRSC repository has been explained in general terms of it containing MODBLKS stored resource definitions. The following sections examine the contents within the IMSRSC repository from a structural perspective.

Repository data
An IMSRSC repository keeps track of the MODBLKS stored resource definitions for each IMS system in the IMSplex. So, how does it do this? For each DRD-enabled IMS system that is using the repository, the following two entities are in the IMSRSC repository:

- **IMS resource lists**
  These lists contain the names of resources that are defined to each IMS. Up to eight resource lists can be in the repository for each IMS, one for each resource type: DB, DBDESC, TRAN, TRANDESC, PGM, PGMDESC, RTC, and RTCDESC. If an IMS does not have a resource of a certain type, the resource list for that type of resource does not exist in the repository. The DBCTL system does not have a transaction or routing code resource and descriptor resource list.

- **Resource definitions**
  The resource definition of each resource is defined in the repository. The resource definition consists of the generic definition that applies to all IMS systems that have the resource defined. If an IMS system has one or more attributes that are different from the generic definition, a specific definition is maintained for each IMS. The entire resource definition for a resource with the generic and specific sections is stored as one entity. Each resource definition is stored as one entity, with the key and other information in the repository index data set and the attribute information in the repository member data set.

You can query the resource definitions from the repository and the IMS systems that own the resources by using the type-2 QUERY command for the resources in question. For more information, see “QUERY for resources and descriptors” on page 234, and Example 7-37 on page 234.

When IMS restarts, it reads its own IMS resource list in the repository to determine the list of resource names to be autoimported. The stored resource definitions of the resources owned by the IMS are read from the repository during autoimport. When these definitions are read into the running IMS system, they are referred to as **runtime resource definitions**.
7.5 Common Service Layer

Using DRD with the repository also requires a Common Service Layer (CSL) environment and requires that the following address spaces are available for both single-IMS IMSplexes or multiple-IMS IMSplexes:

- Operations Manager (OM) to process type-2 commands for repository functions
- Resource Manager (RM) to gain access to the Repository Server (which in turn can access the IMSRSC repository or the RS catalog repository using z/OS XCF services)
- Structured Call Interface (SCI) to enable communication between members of the IMSplex

**Important:** SCI is not used for communication between RM and the Repository Server. Communication between these two address spaces is handled by using z/OS XCF services.

A single point of control (SPOC) is also required for entering certain types of repository commands. For example, you can issue the `EXPORT DEFN TARGET(REPO)` command by using a SPOC to write stored resource definitions to the repository.

**Repository Server in command output:** The Repository Server can be included in the command output for a `QUERY IMSPLEX` command, and is shown as an active member of the IMSplex. However, to include the Repository Server in this command output, it must first register with SCI. However, registration is optional. This is the only instance in which SCI will access the RS component of the repository environment. RS registers to SCI if `IMSPLEX(NAME=)` is specified in the FRPCFG member.

If you are already familiar with the CSL and with the RM address space in particular, you might be aware of the ability of RM to work in conjunction with a coupling facility resource structure for certain functions. If a resource structure is available in your repository environment, it will be used to provide repository name and repository type consistency within the IMSplex.

7.6 Batch utilities

Several utilities are provided for use with the repository environment. One is a batch `ADMIN` utility named FRPBATCH, which performs administrative tasks such as adding a repository to the RS catalog repository, updating an existing repository, listing the attributes of a given repository, or starting and stopping a repository.

Other utilities, provided by RM, are available for assistance in migrating to the repository environment and for falling back from it. The RDDS to Repository utility (`CSLURP10`) populates a user repository with the contents of a specified RDDS, and the Repository to RDDS RM utility (`CSLURP20`) reverts definitions from a repository back to an RDDS.

For information about both of these utilities, see 7.8.2, “Offline access through RM utilities” on page 228, and “Batch ADMIN commands” on page 241.
Chapter 7. IMS repository

7.7 Migrating to the repository

Before IMS 12, using the DRD capability required the use of RDDSs. If you are already using DRD with RDDSs, you can replace your multiple RDDSs with a single, shared repository. Even if you have not yet implemented DRD in your shop, you can do so now and use the repository instead of RDDSs for storing your resource definitions. (In this scenario, a small amount of additional setup is required.)

This section explores the implementation steps for using the repository in IMS 12 for both environments: one in which DRD has already been implemented with RDDSs, and one in which DRD will be implemented for the first time.

To begin, the following section outlines the general setup steps that apply to both of these environments.

7.7.1 Setting up the migration

We begin our setup with a walkthrough of allocating the data sets that are required for repository usage, followed by defining the PROCLIB members that are read upon initialization of the address spaces that are part of the repository environment. This section explains the following tasks:

- Allocating the repository data sets
- Setting up the Repository Server
- Setting up the Common Service Layer
- Setting up IMS
- Ensuring parameter consistency

Allocating the repository data sets

We first must allocate the physical data set pairs (VSAM KSDSs) required to use the repository, including the following repositories:

IMSRSC repository  This contains MODBLKS stored resource definitions for our IMS system or systems.

Repository Server catalog repository  This contains information about the user repository, manages the Repository Server functions, and provides a link between the user repository name and VSAM data sets.

Regardless of the repository data set type, remember that each repository data set consists of an index data set and a member data set. The IMSRSC repository can contain a spare in addition to the primary and secondary data set, but a spare is not permitted with the Repository Server catalog repository. Therefore, a maximum of 10 data sets must be defined: two each for the primary, secondary and spare IMSRSC repository data sets, and two each for the primary and secondary RS catalog repository data sets. If a spare is not used, then the user only needs eight data sets.

Tip: Define the primary, secondary and spare data set pairs on different volumes, to ensure availability. Also make sure that the size of the secondary index and member data sets is greater than the size of the primary index and member data sets. Finally, the size of the spare index and member data sets should be greater than the size of the secondary index and member data sets.
Example 7-1 shows the job control language (JCL) used in our test environment for allocating these data sets. We used the following data set names for our user repositories:

- Primary user repository index data set: IMS12Q.IMS12X.REPO.IMSPRI.RID
- Primary user repository member data set: IMS12Q.IMS12X.REPO.IMSPRI.RMD
- Secondary user repository index data set: IMS12Q.IMS12X.REPO.IMSSEC.RID
- Secondary user repository member data set: IMS12Q.IMS12X.REPO.IMSSEC.RMD
- Spare user repository index data set: IMS12Q.IMS12X.REPO.IMSSPR.RID
- Spare user repository member data set: IMS12Q.IMS12X.REPO.IMSSPR.RMD

In addition, for our RS catalog repositories we used the following data set names to maintain consistency with our naming convention:

- Primary Repository Server catalog repository index data set: 
  IMS12Q.IMS12X.REPO.CATPRI.RID
- Primary Repository Server catalog repository member data set: 
  IMS12Q.IMS12X.REPO.CATPRI.RMD
- Secondary Repository Server catalog repository index data set: 
  IMS12Q.IMS12X.REPO.CATSEC.RID
- Secondary Repository Server catalog repository member data set: 
  IMS12Q.IMS12X.REPO.CATSEC.RMD

Example 7-1 JCL for allocating user repository and RS catalog repository data sets

```assembly
//**********************************************************************
//* FUNCTION: ALLOCATE DATA SETS NEEDED FOR THE REPOSITORY USAGE FOR DRD
//**********************************************************************
//RDSALLOC JOB ACTINFO1,'PGMRNAME',CLASS=A,MSGCLASS=H,MSGLEVEL=(1,1),
// NOTIFY=&SYSUID,REGION=128M
//*
/*JOBPARM S=SC64
// JCLLIB ORDER=IMS12Q.PROCLIB
//*
/*/ ALLOCATE DATA SETS
//ALLOCATE EXEC PGM=IDCAMS,DYNAMNBR=200
//SYSPRINT DD SYSOUT=* 
//SYSIN DD SYSOUT=* 
DEFINE CLUSTER(NAME(IMS12Q.IMS12X.REPO.CATPRI.RID) REUSE INDEXED -
  KEYS(128,0) FREESPACE(10 10) RECORDSIZE(282 282) -
  SHAREOPTIONS(2 3) CONTROLINTERVALSIZE(8192) -
  VOLUMES(SBOXI5) CYLINDERS(1 1))
DEFINE CLUSTER(NAME(IMS12Q.IMS12X.REPO.CATPRI.RMD) REUSE INDEXED -
  KEYS(12,0) FREESPACE(20 20) RECORDSIZE(8185 8185) -
  SHAREOPTIONS(2 3) CONTROLINTERVALSIZE(8192) -
  VOLUMES(SBOXI5) CYLINDERS(1 1))
DEFINE CLUSTER(NAME(IMS12Q.IMS12X.REPO.CATSEC.RID) REUSE INDEXED -
  KEYS(128,0) FREESPACE(10 10) RECORDSIZE(282 282) -
  SHAREOPTIONS(2 3) CONTROLINTERVALSIZE(8192) -
  VOLUMES(SBOXI5) CYLINDERS(2 1))
DEFINE CLUSTER(NAME(IMS12Q.IMS12X.REPO.CATSEC.RMD) REUSE INDEXED -
  KEYS(12,0) FREESPACE(20 20) RECORDSIZE(8185 8185) -
  SHAREOPTIONS(2 3) CONTROLINTERVALSIZE(8192) -
  VOLUMES(SBOXI5) CYLINDERS(2 1))
DEFINE CLUSTER(NAME(IMS12Q.IMS12X.REPO.IMSPRI.RID) REUSE INDEXED -
  KEYS(128,0) FREESPACE(10 10) RECORDSIZE(282 282) -
  SHAREOPTIONS(2 3) CONTROLINTERVALSIZE(8192) -
```
Now that the user repository data sets and RS catalog repository data sets have both been allocated, the next step is to set up the Repository Server.

### Setting up the Repository Server

The RS is a BPE-based address space, therefore it requires that a BPE configuration member is defined. In addition, the Repository Server is configured by the parameters specified in the FRPCFG member of PROCLIB. Both of these members are read during initialization of the RS address space.

#### BPE configuration member

Define the BPE configuration member to specify trace settings for the RS address space. This requires a z/OS PPT entry for BPE, which will most likely already be present in your shop.

Example 7-2 shows the BPE configuration member that we used in our test environment for our Repository Server address space.

Example 7-2   BPE configuration member for Repository Server address space

```
# DEFINITIONS FOR REPO TRACES
TRCLEV=(*,HIGH,REPO,PAGES=300) / * DEFAULT ALL TRACES TO HIGH */
```

For more information about BPE, see IMS Version 12 System Administration, SC19-3020.

#### FRPCFG configuration member

Define your FRPCFG configuration member using an 8-character name. In our test system, we named our member FRP12XRP, which is referenced later in the startup procedure for the Repository Server address space in Example 7-4 on page 208.
Example 7-3 shows the FRPCFG configuration member that we used in our test system; the name of our IMSplex is IM12X.

**Important:** The IMSplex that you specify in the FRPCFG configuration member must be the same IMSplex that you specify in your Common Service Layer configuration member (either in the CSL section in the DFSDFxxx member or in the DFSCGxxx member) and the CSL initialization members for SCI, RM, and OM.

Also note in the example that we have specified the names of the RS catalog repository primary and secondary data sets that we allocated in the previous section. The RSNAME= parameter represents the REPOID that will be appended to all messages issued by the Repository Server. The characters “RP” will be added as a suffix to the RSNAME that you specify. Our REPOID will be IMS12XRPRP because we specified RSNAME=IMS12X. Because IMSPLEX(NAME=) is specified, RS will attempt to register to the SCI address space on the same LPAR as the RS if SCI is available.

Another important parameter to specify is the XCF group name, which must be the same for RM, in the CSLRIxxx member, the RS FRPCFG parameter XCF_GROUP_NAME, and on the XCFGROUP= in the FRPBACTH JCL.

**Important:** The XCF group name specified on the XCF_GROUP_NAME= parameter in the FRPCFG member must match the XCF group name specified in both the RM initialization member CSLRIxxx and in the FRPBATCH (batch ADMIN) utility JCL that defines the user repositories to the RS catalog repository (using the ADD function).

For a complete description of the other parameter values included in the FRPCFG configuration member, see IMS Version 12 System Administration, SC19-3020.

**Example 7-3** FRPCFG configuration member parameter definitions

```
XCF_THREADS=8
MAX_COMMUNICATION_RETRY=32
MBR_CORE_MAX=1024
IMSPLEX(NAME=IM12X)
RSNAME=IMS12X
PRIMARY_CATALOG_REPOSITORY_INDEX=IMS12Q.IMS12X.REPO.CATPRI.RID
PRIMARY_CATALOG_REPOSITORY_MEMBER=IMS12Q.IMS12X.REPO.CATPRI.RMD
SECONDARY_CATALOG_REPOSITORY_INDEX=IMS12Q.IMS12X.REPO.CATSEC.RID
SECONDARY_CATALOG_REPOSITORY_MEMBER=IMS12Q.IMS12X.REPO.CATSEC.RMD
VSAM_BUFNO=128
VSAM_BUFSIZE=8
XCF_GROUP_NAME=IM12XREP
AUDIT=NO
AUDIT_FAIL=CONTINUE
AUDIT_LEVEL=HIGH
AUDIT_DEFAULT=NOAUDIT
```

Next, we must create the startup procedure JCL for the Repository Server, so that we can later start this address space as a task. Example 7-4 shows the RS startup procedure that we used in our test environment.

**Example 7-4** Repository Server startup procedure JCL

```
//IM12XREP PROC RGN=128M,BPECFG=BPERECFN,
//     FRPCFG=FRP12XRPRP,SOUT='*
```
//RESPPROC EXEC PGM=BPEINI00,REGION=&RGN,
//PARM='BPEINIT=FRPINI00,BPECNF=&BPECNF,FRPCNF=&FRPCNF'
//*
//STEPLIB DD DSN=IMS12Q.SDFSRESL,DISP=SHR
//*
//FRPPRINT DD SYSOUT=&SOUT
//SYSPRINT DD SYSOUT=* 
//SYSUDUMP DD SYSOUT=* 

Setting up the Common Service Layer

CSL initialization PROCLIB members must be set up when using the RS. The SCI initialization member (CSLSIxxx) and the OM initialization member (CSLOMxxx) simply need to contain standard CSL parameters, where the RM initialization PROCLIB member (CSLRIxxx) contains the standard CSL parameters plus some unique parameters for using the IMSRSC repository and communication with the RS. In this section, we focus on the parameters in these PROCLIB members that are new in IMS 12. For more information about other parameters not expanded upon here, see IMS Version 12 System Definition, GC19-3021.

Resource Manager: If you are already using the SCI and OM components of the CSL in your environment, you only need to complete the setup steps in this section for Resource Manager. Then you can skip to “Resource Manager” on page 211.

Structured Call Interface

Define the CSLSIxxx PROCLIB member and the startup procedure JCL for this address space. Later, you start the SCI address space as a task using the startup procedure, which then reads the initialization member that you defined.

Example 7-5 shows the SCI initialization member that we used in our test system.

Example 7-5 SCI initialization PROCLIB member CSLSI12D

*--------------------------------------------------------------------*
* SCI INITIALIZATION PROCLIB MEMBER CSLSI12D *
*--------------------------------------------------------------------*
ARMRST=N, /* SHOULD ARM RESTART SCI ON FAILURE */
SCINAME=IMS12D, /* SCI NAME (SCIID = IMS12DSC */
IMSPLEX(NAME=IM12X) /* IMSPLEX NAME (CSLIM12X) */
* FORCE(ALL,SHUTDOWN) /* TO CLEAN UP THE GLOBAL INTERFACE STORAGE */
*--------------------------------------------------------------------*
* END OF MEMBER CSLSI12D *
*--------------------------------------------------------------------*

Example 7-6 shows the SCI startup procedure that we used in our test system (used later when we start the SCI address space).

Example 7-6 SCI startup procedure JCL

//*----------------------------------------------------------------------*
//* PARAMETERS:*
//* BPECNF - NAME OF BPE MEMBER*
//* SCIINIT - SUFFIX FOR YOUR CSLSIxxx MEMBER == CSLSI12D*
//* ARMRST - INDICATES IF ARM SHOULD BE USED*
//*----------------------------------------------------------------------*
Operations Manager

Define the CSLOIxxx PROCLIB member and the startup procedure JCL for this address space. Later you start the OM address space as a task using the startup procedure, which then reads the initialization member that you defined.

Example 7-7 shows the OM initialization member that we used in our test system.

Example 7-7  OM initialization PROCLIB member CSLOI12D

```
*--------------------------------------------------------------------*
*  OM INITIALIZATION PROCLIB MEMBER - CSLOI12D                     *
*--------------------------------------------------------------------*
ARMRST=N,                      /* SHOULD ARM RESTART OM ON FAILURE */
CMDLANG=ENU,                   /* USE ENGLISH FOR COMMAND DESC */
CMDSEC=N,                      /* NO COMMAND SECURITY */
OMNAME=IMS12D,                 /* OM NAME (OMID = IMS12DOM) */
IMSPLEX(
   NAME=IM12X),               /* IMSPLEX NAME (CSLIM12X) */
   AUDITLOG=SYSLOG.OM2Q01.LOG), /* MVS LOG STREAM */
CMDTEXTDSN=IMS12Q.SDFSDATA     /* CMD TEXT DATASET */
*--------------------------------------------------------------------*
*  END OF MEMBER CSLOI12D                                           *
*--------------------------------------------------------------------*

Example 7-8 shows the OM startup procedure that we used in our test system (used later when we start the OM address space).

Example 7-8  OM startup procedure JCL

```
//IM12DOM  PROC RGN=128M,TME=1440,SOUT=*,
//       BPECFG=BPECONF,
//       OMINIT=12D,
//       CSLOI=CSLOINI0,
//       PARM1='ARMRST=N,CMDSEC=N'
//*------------------------------------------------------------------*
//*   OM                                                             *
//*------------------------------------------------------------------*
//* PARAMETERS:                                                     *
//* BPECFG - NAME OF BPE MEMBER                                     *
//* OMINIT - SUFFIX FOR YOUR CSLOIxxx MEMBER                        *
//* ARMRST - INDICATES IF ARM SHOULD BE USED                        *
//* CMDLANG - LANGUAGE FOR COMMAND DESCRIPTION TEXT                *
```
Resource Manager

Define the CSLRIxxx PROCLIB member and the startup procedure JCL for this address space. Later you start the RM address space as a task using the startup procedure, which then reads the initialization member that you defined.

Example 7-9 shows the RM initialization member that we used in our test system.

Example 7-9   RM initialization PROCLIB member CSLRIRMX

ARMRST=N,
RMNAME=RMX,
IMSPLEX(NAME=IM12X)
<SECTION=REPOSITORY>
REPOSITORY=(NAME=IMS12XRP,TYPE=IMSRSC,GROUP=IM12XREP,
AUDITACCESS=NOAUDIT)

The REPOSITORY section shown after the other RM initialization parameters is new for IMS 12. Here, we define the name of the IMSRSC repository, the XCF group name, and optional audit log access parameters. There can only be one REPOSITORY= statement. The parameter values (new in this IMS release) have the following meanings:

REPOSITORY=()  This value defines the IMS repository parameters for RM initialization. It is specified within a section with the header <SECTION=REPOSITORY>.

NAME=  This value specifies the repository name that is managed by RM. This name must be same as the repository name defined to the RS. A repository is defined to the RS with the batch ADMIN utility ADD command (see “ADD command” on page 241). The repository name can be up to 44 characters long and can contain the alphanumeric characters (A–Z, 0 - 9) and the following symbols: period (.) at sign (@), number sign (#), underscore (_), and dollar sign ($). The alphabetic characters A–Z can be uppercase only.

Restriction: A repository name of CATALOG cannot be specified, because it is reserved for use by the RS.

TYPE=  This value specifies the repository type. The only valid value is IMSRSC.

GROUP=  This value specifies the Repository Server z/OS XCF group name. This value must be the same as the XCF group name specified on the XCF_GROUP_NAME parameter of the FRPCFG member. RM and the RS must be in the same XCF group. The value must be eight characters padded on the right with blanks. Valid characters are A-Z (uppercase only), 0 - 9, and the following symbols: number sign (#), dollar sign ($), and at sign (@).

AUDITACCESS=  This is an optional parameter. It specifies the repository audit access level for the specified repository. If this value is not specified, the audit access level
defaults to the level of auditing set by the `AUDIT_DEFAULT` parameter in the FRPCFG member of the IMS PROCLIB data set. The `AUDITACCESS` parameter has the following valid values:

**DEFAULT** Use the rule specified in the `AUDIT_DEFAULT` parameter of the FRPCFG member.

**NOAUDIT** No auditing of member access.

**SECURITY** Audit security failures only.

**UPDATE** Audit member access with update intent.

**READ** Audit member access with read or update intent.

**SYSTEMREAD** Audit member access with system-level read, read, or update intent. A read of the resource definition by the system before the update request is identified as a “system read” request.

You can specify `AUDITACCESS` in the CSLRlxxx member if you have more than one IMSRSC repository being managed by the RS address space and want to have different audit level specifications for each repository. Example 7-10 shows the RM startup procedure that we used in our test system (executed later when we started the RM address space).

**Example 7-10  RM startup procedure JCL**

```plaintext
// *------------------------------------------------------------------*
//   RM                                                            *
// *------------------------------------------------------------------*
// * PARAMETERS:                                                   *
// * BPECFG  - NAME OF BPE MEMBER                                   *
// * RMINIT  - SUFFIX FOR YOUR CSLRlXXX MEMBER                     *
// * ARMST  - INDICATES IF ARM SHOULD BE USED                      *
// * RMNAME  - NAME OF THE RM BEING STARTED                         *
// *------------------------------------------------------------------*
//* IEFPROC EXEC PGM=BPEINI00,REGION=3000K,  *
// PARM=('BPECFG=BPECONFG','BPEINIT=CSLRINI0','RMINIT=RMX')   *
//* STEPLIB DD DSN=IMS12Q.SDFSRESL,DISP=SHR  *
//PROCLIB DD DSN=IMS12Q.PROCLIB,DISP=SHR  *
//SYSPRINT DD SYSOUT=*  *
//SYSUDUMP DD SYSOUT=*  *
//*
```

**Setting up IMS**

Upon initialization, IMS reads the DFSDFxxx PROCLIB member (Example 7-11) to apply specific processing options for various functions. This member contains multiple sections that apply to these functions. Here, we focus on the sections that apply to the CSL, DRD, and repository, again with emphasis on the parameters that are new in IMS 12.
Example 7-11  System Definition PROCLIB member DFSDF12D

```
* --------------------------------------------------------------- *
* MEMBER DFSDF12D                                              *
* DFSDF= PARAMETER ON CONTROL REGION EXECUTION OR DFSPBXXX       *
* --------------------------------------------------------------- *
* PARAMETERS IN DFSQQXXX AND DFGCGXXX MEMBERS                   *
* OVERRIDE THE ATTRIBUTES DEFINED IN THE DFSDFXXX MEMBER        *
* --------------------------------------------------------------- *
* COMMON SERVICE LAYER SECTION                                 *
* --------------------------------------------------------------- *
*<SECTION=COMMON_SERVICE_LAYER>                               *
IMSPLEX=IM12X        /* IMSPLEX NAME                        */
MODBLKS=DYN          /* DRD ENABLED;MODBLKS OLC DISABLE */
* --------------------------------------------------------------- *
* DYNAMIC RESOURCES SECTION                                   *
* --------------------------------------------------------------- *
*<SECTION=DYNAMIC_RESOURCES>                                  *
AUTOEXPORT=AUTO,     AUTOIMPORT=AUTO,    RDDSDSN=(
  IMS12Q.IMS12D.RDDS1,
  IMS12Q.IMS12D.RDDS2,
  IMS12Q.IMS12D.RDDS3)
* --------------------------------------------------------------- *
* REPOSITORY SECTION                                           *
* --------------------------------------------------------------- *
*<SECTION=REPOSITORY>                                          *
REPOSITORY=(TYPE=IMSRSC)
* --------------------------------------------------------------- *
* END OF MEMBER DFSDF12D                                        *
```

For each of the following sections, refer back to Example 7-11 on page 213. For more information about other parameters that are not expanded upon here, see IMS Version 12 System Definition, GC19-3021.

**Common Service Layer section**

CSL enablement parameters have historically been defined in the DFSCGxxx PROCLIB member, but it is now recommended to define them in the DFSDFxxx member in the CSL section flagged by a `<SECTION=COMMON_SERVICE_LAYER>` header. If parameters are specified in both members, the DFSCGxxx member will take precedence.

Define the following parameters in the CSL section of the DFSDFxxx member for repository enablement:

- **MODBLKS=DYN** to indicate DRD usage (versus MODBLKS online change)
- **RMENV=Y** to indicate that an RM address space will be used

**Dynamic Resource Definition section**

The **AUTOIMPORT=**AUTO or **AUTOIMPORT=**REPO parameter is important to define in the DRD section of the DFSDFxxx member (flagged by a `<SECTION=DYNAMIC_RESOURCES>` header) for autoimport from the repository.
The **AUTOIMPORT** parameter specifies that resource and descriptor definitions are automatically imported during IMS cold start. Automatic import from the IMS repository is enabled if all of the following conditions are true:

- IMS is enabled with DRD.
- The repository section of the DFSDFxxx member is defined with **TYPE=IMSRSC**.
- IMS is enabled with RM services.
- The CSLRIxxx member is defined for the repository with **TYPE=IMSRSC**.
- The repository contains stored resource definitions for the IMS system.
- RM is started with the repository enabled.
- The RS address space is started and available.

Otherwise, automatic import is attempted from the RDDS or MODBLKS if certain other conditions are true.

The **AUTOIMPORT** parameter specifies that stored resource and descriptor definitions are imported (automatic import) from the IMSRSC repository. In this case, the CSLRIxxx member and the REPOSITORY section of the DFSDFxxx member must also be defined with **TYPE=IMSRSC** parameter. Take into account the following considerations when the **AUTOIMPORT** parameter is specified:

- If the repository does not contain the stored resource definitions for the IMS, then IMS comes up with no resources. IMS issues the **DFS4404I** message if the repository is empty.
- If the IMS resource list in the repository is not empty, IMS processes the resource definitions returned by the RM.
- If an error occurs while processing the returned definitions, the action that is taken is based on the IMPORTERR parameter setting.
- If there is an error reading from the repository, other than if the IMS resource list is not found, a **DFS4401E** message is issued with the RM return and reason code. Action is taken based on the REPOERR parameter setting.
- If the **AUTOIMPORT** parameter is specified and no REPOSITORY section is defined, or the REPOSITORY statement for the repository is not defined, the **DFS4403E** message is issued. IMS initialization abends with U0071 with return code X'27'. The **DFS2930** message is issued with completion code 27,2108 before the abend.
- If the **AUTOIMPORT** parameter is specified and **RMENV=N** is specified in the CSL section, IMS initialization abends with U0071 with return code X'27', because IMS cannot access the repository for the stored resource definitions. IMS requires the CSL RM address space to access the repository. The **DFS2930** message is issued with completion code 27,210C before the abend.

For detailed information about automatic import from these other sources, see *IMS Version 12 System Definition*, GC19-3021.
Repository section

There is a new repository section flagged by a `<SECTION=REPOISTORY>` header that specifies the type of IMS repository that will be used. As mentioned earlier, the type of repository used with DRD is `TYPE=IMSRSC`, which is currently the only valid option. There can be only one `TYPE=` statement within this section.

Ensuring parameter consistency

Two major parameters tie together the processing for a Repository Server. The first parameter is the z/OS XCF group name, which is used for communications with the Repository Server. The same XCF group name must be used in all the member definitions that use a particular repository.

The second parameter is the actual name of the IMSRSC repository that holds the DRD definitions for the IMSplex. Figure 7-2 illustrates each parameter in more detail.

![Diagram of Repository Parameter Consistency Requirements](image)

**XCF group name**

The parameter for the XCF group name specifies the Repository Server z/OS cross-system coupling facility group name. Resource Manager and the Repository Server must be in the same XCF group. The value must be eight characters and padded on the right with blanks.
More specifically, the same XCF group name must be specified in the Repository Server FRPCFG configuration member, RM initialization CSLRIxxx member, and the FRPBATCH JCL for invoking the batch ADMIN utility. It is important to ensure that the XCF group name parameter value is consistent in each of these places for a given repository.

**IMSRSC repository**

The parameter for the IMSRSC repository specifies the IMSRSC repository name that is managed by RM. The repository name can be up to 44 characters long. The name of the IMSRSC repository is defined in the RM initialization PROCLIB member (CSLRIxxx), and must match the repository name specified when you invoke the batch **ADMIN** utility.

Now that you have defined the PROCLIB members required for the repository environment, you can begin initializing the address spaces associated with them.

### 7.7.2 Initializing the SCI and OM Common Service Layer components

SCI is the first address space that must be initialized in an IMSplex. This is because every other address space that attempts to join an IMSplex must first register with SCI before being permitted to do so. In our test environment, we started our SCI address space named IM12DSCI as a task using the startup procedure JCL in Example 7-6 on page 209.

Next, we initialized our OM address space named IM12DOM using the startup procedure JCL in Example 7-8 on page 210.

### 7.7.3 Initializing the Repository Server

Now that we have initialized the SCI and OM address spaces, we can now initialize our RS address space. The RS is generally initialized and started before RM before any client can connect to the repository. We initialized our RS named IM12XREP by using the startup procedure JCL in Example 7-4 on page 208. It attained master RS status because no other RS currently exists in the IMSplex and this is reflected in a FRP2002I message, shown in Example 7-12.

#### Example 7-12  FRP messages displayed upon Repository Server initialization

```
IEF695I START IM12XREP WITH JOBNAME IM12XREP IS ASSIGNED TO USER STC
$HASP373 IM12XREP_STARTED
IEF403I IM12XREP - STARTED - TIME=22.15.58 - ASID=00C5 - SC64
FRP2006I - Server starting: Release 1.2.0, XCF group IM12XREP
FRP2012I - Opening repository: CATALOG
FRP2016I - Repository opened: CATALOG
FRP2002I - Master repository server status obtained IMS12XRP
FRP2025I - Server start completed IMS12XRP
FRP2026I - XCF group IM12XREP joined successfully IMS12XRP
```

**Master and subordinate Repository Servers**

If the master RS fails, a subordinate RS can take over for it to maintain availability. Any RS address space initialized after a master RS is active will become a subordinate RS. It will wait in an initialization state until it detects that the master RS has failed. At this point, the subordinate RS will complete the startup process. Because you can have multiple subordinate RS address spaces, the first one to complete this startup process will become the new master. The others will remain subordinate RSs.
Now that we have initialized our RS address space, we can define our user repository to the RS catalog repository.

### 7.7.4 Defining the IMSRSC repository to the Repository Server catalog

To define an IMSRSC repository to the RS catalog repository, use the batch `ADMIN` utility to issue the `ADD` command. After the IMSRSC repository has been added, you can then start it by using a `START` command. To add and start our IMSRSC repository named IMS12XRP, we used the JCL shown in Example 7-13.

**Example 7-13  Defining an IMSRSC repository to the RS catalog repository with `FRPBATCH`**

```jcl
//DEFNREPO JOB ACTINFO1,
// 'PGMRNAME',
// CLASS=A,
// MSGCLASS=H,MSGLEVEL=(1,1),
// NOTIFY=IMSR2,
// REGION=128M
//*
/*JOBPARM S=SC64
// JCLLIB ORDER=IMS12Q.PROCLIB
//*
//REPOADD EXEC PGM=FRPBATCH,PARM='XCFGROUP=IM12XREP'
//STEPLIB DD DISP=SHR,DSN=IMS12Q.SDFSRESL
//SYSPRINT DD SYSOUT=*  
//SYSIN DD *
ADD REPOSITORY(IMS12XRP) +
   REPDSNIRID(IMS12Q.IMS12X.REPO.IMSPRI.RID) +
   REPDSN1RMD(IMS12Q.IMS12X.REPO.IMSPRI.RMD) +
   REPDSN2RID(IMS12Q.IMS12X.REPO.IMSSEC.RID) +
   REPDSN2RMD(IMS12Q.IMS12X.REPO.IMSSEC.RMD) +
   REPDSN3RID(IMS12Q.IMS12X.REPO.IMSSPR.RID) +
   REPDSN3RMD(IMS12Q.IMS12X.REPO.IMSSPR.RMD) +
   AUTOOPEN(YES)
START REPOSITORY(IMS12XRP)
//*
```

We named our IMS12XRP IMSRSC repository on the `REPOSITORY()` parameter, and we specified the names of our IMSRSC repository primary, secondary, and spare index and member data sets that we previously allocated in Example 7-1 on page 206.

**Repository name:** The repository name is converted to uppercase if it is specified with anything else.

You can also control whether the repository data sets you are specifying with this command are opened when the repository is started (AUTOOPEN(YES) is the default, which we have included for clarity) or when a user first connects to it (AUTOOPEN(NO)). Lastly, after adding our user repository to the RS catalog repository, we started it with the `START` command.

Now that we have defined our IMSRSC repository to the RS catalog repository, we can initialize the RM address space.
7.7.5 Initializing the RM Common Service Layer component

In our test environment, we started our RM address space named IM12XRM as a task using the startup procedure JCL in Example 7-10 on page 212.

When the RM address space is initializing, RM attempts to connect to the repository name specified in the CSLRlxxx member on the REPOSITORY= statement. In our case, it connects to the IMSRSC repository defined to the RS catalog repository (Example 7-13 on page 217). RM then issues message CSL2500I (Example 7-14) that we received when we initialized RM in our test environment. If the repository is empty and this is the first RM initialized repository, you also see a CSL2501I message.

Example 7-14   RM successfully connecting to IMSRSC repository

IEF695I START IM12XRM WITH JOBNAME IM12XRM IS ASSIGNED TO $HASP373 IM12XRM STARTED
IEF403I IM12XRM - STARTED - TIME=22.16.48 - ASID=0080 - SC64
CSL2500I RM RMXRMR CONNECT REQUEST SUCCESSFUL FOR 893
REPOSITORY NAME=IMSRSC
REPOSITORY TYPE=IMSRSC  RMXRMR
CSL0020I RM READY RMXRMR

Attention: If you are already using RM in your shop, you do not need to restart the RM address space to enable it for repository usage. Instead, you can issue a type-2 UPDATE RM command to dynamically enable it using the following syntax:

UPDATE RM TYPE(REPO) REPOTYPE(IMSRSC) SET(REPO(Y))

Ensure that your CSLRlxxx and DFSDFxxx members have been defined with the required repository enablement parameters before issuing this command.

With APAR PM41952 applied: If you started RM before the RS is started, RM initialization will issue a CSL2502A highlighted message and will attempt to register to RS every 5 seconds until it is successful or RM is terminated. Also, if RS is started, but the repository name RM is trying to connect to is not defined to the RS or is not available, RM will issue a CSL2503A highlighted message and attempt to connect to the repository every 5 seconds.

Now that RM has been started, we can begin working with the IMSRSC repository in various ways. We begin by populating the IMSRSC repository with resource definitions.

7.7.6 Populating the IMSRSC repository

You can populate the IMSRSC repository in several ways. The best method is determined by factors such as whether IMS is active or inactive, and whether you have already implemented DRD in your shop. This section reviews the multiple ways that you can populate the IMSRSC repository, beginning with using the RDDS.

Populating the repository from RDDS
To populate the IMSRSC repository with the contents of an RDDS, you can run the RDDS to Repository utility (CSLURP10). This method is the recommended way to populate the repository if you have already implemented DRD in your shop and your IMS is currently inactive. The idea here is to prepare the repository offline by first copying the stored resource definitions in the RDDS to the IMSRSC repository, and then restarting your IMS system with AUTOIMPORT=auto specified in its DFSDFxxx member so that it will read in the resource definitions from the IMSRSC repository.
If AUTOIMPORT is not used on a cold start or IMS is restarting with a warm or emergency restart, you issue the IMPORT command to read the DRD definitions from the repository.

**Tip:** Choose the most current system RDDS for input to the RDDS to Repository utility (CSLURP10). You can determine which system RDDS is the most current by browsing each RDDS and viewing the timestamp information, which is included in clear text in the header record. Determine the system RDDS names by viewing the RDDSDSN=() parameter in the DFSDFxxx member of the IMS system.

To test this utility in our environment we used the most current system RDDS associated with our IM12D system, which was IMS12Q.IMS12D.RDDS1. Example 7-15 shows the JCL that we used to run the RDDS to Repository utility (CSLURP10).

**Example 7-15  Populating an IMSRSC repository from an RDDS**

```jcl
//POPREPOS JOB ACTINFO1,
// 'PGMRNAME',
// CLASS=A,
// MSGCLASS=H,MSGLEVEL=(1,1),
// NOTIFY=IMSR2,
// REGION=128M
//*
/*JOBPARM S=SC64
// JCLLIB ORDER=IMS12Q.PROCLIB
//*********************************************************************
//* FUNCTION: Populate data into the IMS repository from RDDS
//*********************************************************************
/*
/STEP1 EXEC PGM=CSLURP10,MEMLIMIT=4G
/STEPLIB DD  DSN=IMS12Q.SDFSRESL,DISP=SHR
/SYSUDUMP DD SYSOUT=* 
/SYSPRINT DD SYSOUT=* 
/RDDSDSN DD DISP=SHR,DSN=IMS12Q.IMS12D.RDDS1
/SYSIN DD *
  IMSPLEX(NAME=IM12X IMSID(I12D))
/*
```

After we ran the utility, we browsed the output to confirm that the definitions were successfully copied from our specified RDDS to the IMSRSC repository which is shown in Example 7-16.

**Example 7-16  RDDS to Repository utility (CSLURP10) job output**

```
CSL2603I CSLURP10 IS PROCESSING RDDS IMS12Q.IMS12D.RDDS1
CSL2618I CSLURP10 IS PROCESSING PLEX=CSLIM12X, IMSID LIST FROM SYSIN, I12D
CSL2620I CSLURP10 SUCCESSFUL REGISTRATION WITH RM, RMNAME=RMXRM
CSL2600I CSLURP10 WRITE TO REPOSITORY WAS SUCCESSFUL FOR REPOTYPE=IMSRSC ,
  REPONAME=IMS12XR

**** SUMMARY ****

<table>
<thead>
<tr>
<th>DB</th>
<th>COUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>:15</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DBDESC</th>
<th>COUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>:0</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PGM</th>
<th>COUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>:53</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PGMDESC</th>
<th>COUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>:0</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RTC</th>
<th>COUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>:3</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>RTCDESC</th>
<th>COUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>:0</td>
<td></td>
</tr>
</tbody>
</table>
```
If you had already implemented DRD in your shop and you want to populate the IMSRSC repository while IMS is active, you can do so dynamically by using a type-2 EXPORT command, as explained in the following section.

### Populating the repository from current IMS runtime resource definitions

If you are migrating an active DRD-enabled IMS to the repository, you can capture its runtime resource definitions by issuing a type-2 EXPORT command. To use this method of populating the repository, it should be empty at this point. Use the following syntax to ensure that all of its runtime resource definitions are captured:

```
EXPORT DEFN TARGET(REPO) TYPE(ALL) NAME(*) OPTION(ALLRSP)
```

**Important:** Route the command to the IMS system whose runtime resource definitions you want to capture. If you do not specify any command routing, ensure that you include the IMSID on the `SET(IMSID())` parameter.

We issued this flavor of the EXPORT command and routed it to our repository-enabled IMS (I12D); the output is shown in Figure 7-3.
Populating the repository from MODBLKS

If you are migrating to DRD for the first time, you can populate the IMSRSC repository using the MODBLKS as a starting point.

Attention: Migration from MODBLKS online change to DRD requires an IMS cold start and some other general setup. For more information about migrating from MODBLKS online change to DRD, see IMS Version 12 System Definition, GC19-3021.

Currently no utility is available that uses a MODBLKS data set as input and directly generates an equivalent IMSRSC repository. Two utilities must be used: one that creates a non-system RDDS from MODBLKS, and the other that generates an IMSRSC repository from the non-system RDDS created with the first utility.

As you can see, a non-system RDDS must be used as a temporary bridge between the MODBLKS and IMSRSC repository. Begin by allocating a non-system RDDS for use with these utilities. Then use the Create RDDS from MODBLKS utility (DFSURCM0) to generate an RDDS with contents equivalent to your MODBLKS data set. You can then use this RDDS, in turn, as input to the RDDS to Repository utility (CSLURP10) to generate equivalent IMSRSC.
repository contents. For more information about these utilities, see *IMS Version 12 System Utilities*, SC19-3023.

We begin by allocating a non-system RDDS named IMSR2.TEMPRDDS using the JCL shown in Example 7-17.

*Example 7-17  JCL to allocate a non-system RDDS*

```
//RDDSALLC JOB  CLASS=A,MSGCLASS=H,MSGLEVEL=(1,1)
//S1       EXEC PGM=IEBGENER
//SYSOUT   DD  SYSOUT=* 
//SYSPRINT DD  SYSOUT=* 
//SYSUT1   DD  DUMMY,BLKSIZE=32760,RECFM=VB 
//SYSUT2   DD  DSN=IMSR2.TEMPRDDS, 
//          DCB=(RECFM=VB,LRECL=32756,BLKSIZE=32760), 
//          UNIT=SYSDA, 
//          DISP=(),CATLG),SPACE=(TRK,(10,10)) 
//SYSIN    DD  DUMMY 
```

To populate our IMSRSC repository with the stored definitions in our MODBLKS, we ran the JCL shown in Example 7-18. This JCL invoked the Create RDDS from MODBLKS utility (*DFSURCM0*) and the RDDS to Repository utility (*CSLURP10*).

*Tip: The SUFFIX= parameter (can be abbreviated as SUF=) shown in Example 7-18 specifies a 1-character value for the suffix that is associated with the members in the MODBLKS data set. Ensure that this parameter is specified correctly by confirming the suffix character for the members that exist within your MODBLKS data set. The SUFFIX default value is 0.*

*Example 7-18  Populating an IMSRSC repository from MODBLKS using DFSURCM0 and CSLURP10*

```
//POPREPOS JOB ACTINFO1, 
// 'PGMRNAME', 
// CLASS=A, 
// MSGCLASS=H,MSGLEVEL=(1,1), 
// NOTIFY=IMSR2, 
// REGION=128M 
// * 
//*/JOBPARM S=SC64 
// JCLLIB ORDER=IMS12Q.PROCLIB 
//********************************************************************* 
//(************ Populating data into the IMS repository 
//(********************************************************************* 
//STEP1    EXEC PGM=DFSURCM0 
//STEPLIB  DD  DSN=IMS12Q.SDFSRESL,DISP=SHR 
//MODBLKS  DD DISP=SHR,DSN=IMS12Q.IMS12D.MODBLKSA 
//RDDSDSN  DD DSN=IMSR2.TEMPRDDS,DISP=SHR, 
//          UNIT=SYSDA, VOL=SER=SBOX79, 
//          SPACE=(CYL,(1,1),RLSE), 
//          DCB=(LRECL=32756,BLKSIZE=32760,RECFM=VB) 
//SYSPRINT DD SYSOUT=*, 
//          DCB=(LRECL=133,BLKSIZE=6118,RECFM=FBA) 
//REPORT   DD SYSOUT=*, 
//          DCB=(LRECL=133,BLKSIZE=6118,RECFM=FBA) 
//CONTROL  DD * 
//IMSID=I12D 
```
SYSTYPE=DBDC
SUF=D

/*
//STEP2    EXEC PGM=CSLURP10,MEMLIMIT=4G
//STEP1B DD DSN=IMS12Q.SDFSRESL,DISP=SHR
//SYSUDUMP DD SYSOUT=* 
//SYSPRINT DD SYSOUT=* 
//RDDSDDSN DD DISP=SHR, DSN=IMSR2.TEMPRDDS
//SYSIN DD *
IMSPLEX(NAME=IM12X IMSID(I12D))
//*/

After the job is run, we browse the output (Example 7-19).

Example 7-19  Job output from running DFSURCM0 and CSLURP10

* ******************************************
*  RDDS BUILD FROM MODBLKS / DFSURCM0  DATE: 2011/214  TIME: 18:04
* ******************************************

* CONTROL: IMSID=I12D
* CONTROL: SYSTYPE=DBDC
* CONTROL: SUF=D

* RDDS DATA SET NAME:
  * IMSR2.TEMPRDDS

* MODBLKS DATA SET NAME:
  * IMS12Q.IMS12D.MODBLKSA

* IMSID USED . . : I12D
* SUFFIX USED . . : D
* SYSTYPE USED . . : DBDC

* TRAN MEMBER NAME. . . . . . . . : DFSSMBOD
  * SIZE OF MEMBER . . . . . . . : 6440
  * SIZE OF MODBLKS BLOCK . . . . . . : 184
  * SIZE OF CHKP BLOCK . . . . . . : 184
  * SIZE OF RSCX EXTENSION . . . . . : 60
  * TOTAL NUMBER OF BLOCKS . . . . . : 35
  * BLOCKS PER RDDS RECORD . . . . . : 134

* DB MEMBER NAME. . . . . . . . : DFSDDIRD
  * SIZE OF MEMBER . . . . . . . : 2520
  * SIZE OF MODBLKS BLOCK . . . . . . : 168
  * SIZE OF CHKP BLOCK . . . . . . : 51
  * SIZE OF RSCX EXTENSION . . . . . : 60
  * TOTAL NUMBER OF BLOCKS . . . . . : 15
  * BLOCKS PER RDDS RECORD . . . . . : 64

* PGM MEMBER NAME. . . . . . . . : DFSPDIRD
  * SIZE OF MEMBER . . . . . . . : 5712
  * SIZE OF MODBLKS BLOCK . . . . . . : 112
  * SIZE OF CHKP BLOCK . . . . . . : 24
  * SIZE OF RSCX EXTENSION . . . . . : 60
  * TOTAL NUMBER OF BLOCKS . . . . . : 50
** IBM IMS Version 12 Technical Overview

* BLOCKS PER RDDS RECORD . . : 88
* * BLOCK FOR DBF#FPUO PDIR WAS REMOVED
* * RTC MEMBER NAME. . . . . : DFSRCTED
* SIZE OF MEMBER . . . . . . : 144
* SIZE OF MODBLKS BLOCK. . . : 48
* SIZE OF CHKP BLOCK . . . . : 28
* SIZE OF RSCX EXTENSION . . : 60
* TOTAL NUMBER OF BLOCKS . . : 3
* BLOCKS PER RDDS RECORD . . : 88
*
* RDDS BLKSIZE : 32760
*
* *************************************************
* RDDS BUILD INFORMATION SUMMARY
* *************************************************
* TOTAL # OF NON COMMENT CONTROL STATEMENTS READ . . : 3
* TOTAL # OF TRANACT EDIT ROUTINE CONTROL STMTS . . : 0
* TOTAL # OF RDDS RECORDS WRITTEN. . . . . . . . . . . . . : 6
* TOTAL # OF TRAN ENTRIES IN MODBLKS . . . . . . . . : 35
* TOTAL # OF TRAN RDDS RECORDS WRITTEN . . . . : 1
* TOTAL # OF DB ENTRIES IN MODBLKS . . . . . . . . : 15
* TOTAL # OF DB RDDS RECORDS WRITTEN . . . . : 1
* TOTAL # OF PGM ENTRIES IN MODBLKS . . . . . . . . : 50
* TOTAL # OF PGM RDDS RECORDS WRITTEN . . . . : 1
* TOTAL # OF RTC ENTRIES IN MODBLKS . . . . . . . . : 3
* TOTAL # OF RTC RDDS RECORDS WRITTEN . . . : 1
* *************************************************

CSL26031 CSLURP10 IS PROCESSING RDDS IMSR2.TEMPRDDS
CSL26181 CSLURP10 IS PROCESSING PLEX=CSLIM12X, IMSID LIST FROM SYSIN, I12D
CSL26201 CSLURP10 SUCCESSFUL REGISTRATION WITH RM, RMNAME=RMXRM
CSL26001 CSLURP10 WRITE TO REPOSITORY WAS SUCCESSFUL FOR REPOTYPE=IMSRSC ,
REPONAME=IMS12XRP

**** SUMMARY ****

<table>
<thead>
<tr>
<th></th>
<th>COUNT</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>DB</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>DBDESC</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>PGM</td>
<td>50</td>
<td></td>
</tr>
<tr>
<td>PGMDESC</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>RTC</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>RTCDESC</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>TRAN</td>
<td>35</td>
<td></td>
</tr>
<tr>
<td>TRANDESC</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>DB DUPLICATES</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>DBDESC DUPLICATES</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>PGM DUPLICATES</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>PGMDESC DUPLICATES</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>RTC DUPLICATES</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>RTCDESC DUPLICATES</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>TRAN DUPLICATES</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>TRANDESC DUPLICATES</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
Now that your repository has been populated with stored resource definitions, follow the remainder of the DRD migration steps outlined in *IMS Version 12 System Definition*, GC19-3021. When you restart your IMS system with the **AUTOIMPORT=AUTO** parameter specified in its DFSDFxxx member, it reads in the resource definitions from the IMSRSC repository.

### 7.7.7 Starting IMS

The next step is starting your IMS system. In our test environment, we restarted our IMS. Because we specified the **AUTOIMPORT=AUTO** parameter in its DFSDF12D member (see Example 7-11 on page 213), it read in the stored resource definitions from the IMSRSC repository that we previously populated offline (see Example 7-18 on page 222).

**Attention:** If your IMS system is DRD-enabled and you have been using RDDSs up to this point, you do not need to restart your IMS system to enable it for repository usage. Instead, you can issue a type-2 **UPDATE IMS** command to dynamically enable it using the following syntax:

```
UPDATE IMS SET(LCLPARM(REPO(Y) REPOTYPE(IMSRSC))
```

Ensure that your DFSDFxxx member has been defined with the required repository enablement parameters before issuing this command. Afterwards, if you have not already populated your repository, you can use the **EXPORT** command to export all of the runtime resource definitions of your IMS to it at this time.

### 7.7.8 Listing status information for IMSRSC repository with FRPBATCH

Another action you can take before or after you start your IMS system is to display information about a specific IMSRSC repository defined to the RS catalog. You use the batch **ADMIN** utility to issue the **LIST REPOSITORY()** command. To display information about our IMSRSC repository named IMS12XRP, we used the JCL shown in Example 7-20.

**Example 7-20  Batch ADMIN LIST command to display the information of a single IMSRSC repository**

```
//LISTREPO JOB ACTINFO1,
// 'PGMRNAME',
// CLASS=A,
// MSGCLASS=H,MSGLEVEL=(1,1),
// NOTIFY=IMSR2,
// REGION=128M
//*
/*JOBPARM S=SC64
// JCLLIB ORDER=IMS12Q.PROCLIB
//*/
//* FUNCTION: List the information of the IMS repository
//REPOLST EXEC PGM=FRPBATCH,PARM='XCFGROUP=IM12XREP'
//STEPLIB DD DISP=SHR,DSN=IMS12Q.SDFSRESL
//SYSPRINT DD SYSOUT=* 
//SYSIN DD *
//*/
LIST REPOSITORY(IMS12XRP)
```
After the batch ADMIN utility finished executing we browsed the output, which is shown in Example 7-21.

Example 7-21  Output for the batch ADMIN LIST REPOSITORY command

```
LIST REPOSITORY(IMS12XRP)
Repository Name   : IMS12XRP

   Last updated date/time  : 2011/07/26 01:44:31
   Status                 : OPEN
   Auto-open              : YES
   Security Class         : NOT DEFINED

Repository Data Set pair 1
   Index (RID)   : IMS12Q.IMS12X.REPO.IMSPRI.RID
   Member (RMD)  : IMS12Q.IMS12X.REPO.IMSPRI.RMD
   Status        : COPY1

Repository Data Set pair 2
   Index (RID)   : IMS12Q.IMS12X.REPO.IMSSEC.RID
   Member (RMD)  : IMS12Q.IMS12X.REPO.IMSSEC.RMD
   Status        : COPY2

Repository Data Set pair 3
   Index (RID)   : IMS12Q.IMS12X.REPO.IMSSPR.RID
   Member (RMD)  : IMS12Q.IMS12X.REPO.IMSSPR.RMD
   Status        : SPARE

FRP4750I - LIST command processing completed successfully
```

There is also a flavor of the batch ADMIN LIST command that displays general information about all user repositories that are defined to the RS. In Example 7-20 on page 225, simply replace the LIST REPOSITORY() parameter with LIST STATUS. We ran the batch ADMIN LIST STATUS command in our test environment and received the output shown in Example 7-22.

Example 7-22  The batch ADMIN LIST STATUS command to display all IMSRSC repository information

```
LIST STATUS
Repository          Status  Changed     ID      RDS1     RDS2     RDS3
------------------- --------- ---------- ------ ------ ------ ------
IMS12XRP            OPEN     2011/07/26 IMSR3    COPY1    COPY2    SPARE
FRP4750I - LIST command processing completed successfully
```

Notice that the output indicates a status for the primary, secondary, and spare repository data sets. When the user repository data sets are originally defined to the RS catalog repository, the primary starts out with a COPY1 status, the secondary with COPY2 status, and the spare with SPARE status. These statuses will change when a write error occurs on either the primary or secondary repository data sets. In this case, the RS will drive recovery and the statuses for the repository data sets will change.

For example, when a primary repository data set starts out with a COPY1 status and a write error occurs on it, its status will change from COPY1 to DISCARD status. The current status of the spare data set changes from SPARE to COPY1. You must then allocate and define a new repository data set as the new spare and manually set its status to SPARE with the batch ADMIN DSCHANGE command. For more information about this command, see “DSCHANGE command” on page 244.
7.7.9 Cleaning up the DFSDFxxx member

After you successfully migrate an IMS to the repository, remove the RDDDSN() parameter from its DFSDFxxx member because it is no longer required. If you specified AUTOIMPORT=AUTO, then the next time you restart this IMS system, it reads from the repository. Lastly, if you specified AUTOEXPORT=AUTO, you can also remove this definition because it does not apply to an environment in which repository DRD is being used.

**Tip:** You can disable autoexport for an IMS after it is enabled to use the repository by issuing the UPDATE IMS SET(AUTOEXPORT(NO)) command. Otherwise, if your RDDSs are still defined, autoexport will continue to occur at each system checkpoint if definitional changes have been made since the previous system checkpoint.

7.8 IMSRSC repository access

You can access the IMS RS by using the RM address space, online by using type-2 commands or offline by using RM utilities. In addition, you can use facilities that do not use RM to access the repository: the batch ADMIN utility (FRPBATCH), which uses XCF, and a set of RS commands that use the z/OS modify interface. These facilities are used for repository management. This section explores the use of each of these methods for repository access.

7.8.1 Online access through RM

During online operation, several new and modified RM and IMS type-2 commands apply to the repository environment:

- UPDATE RM
- QUERY RM
- UPDATE IMS
- QUERY IMS
- EXPORT DEFN
- IMPORT DEFN
- DELETE DEFN
- QUERY DB/DBDESC/PGM/PGMDESC/TRAN/TRANDESC/RTC/RTCDESC with the following SHOW() parameters:
  - SHOW(DEFN)
  - SHOW(DEFN, IMSID)
  - SHOW(DEFN, LOCAL)
  - SHOW(DEFN, GLOBAL)
  - SHOW(DEFN, IMSID, LOCAL)
  - SHOW(DEFN, IMSID, GLOBAL)
  - SHOW(IMSID)

These commands are used for dynamic enablement and disablement of a repository, to display status of RM and IMS, and to create, update, delete, and query the DRD stored resource definitions in the IMSRSC repository. Remember that the DRD CREATE, UPDATE, and DELETE commands work with runtime definitions, not the stored resource definitions in the repository. For more information about these commands, see 7.9.1, “IMS repository commands” on page 230.
### 7.8.2 Offline access through RM utilities

Two RM utilities are provided to write to and read from the IMSRSC repository, and they can be used for migration and fallback with the repository. They access the repository offline through RM. The RDDS to Repository utility (CSLURP10) populates an IMSRSC repository with the contents of a specified RDDS. The Repository to RDDS utility (CSLURP20) creates definitions from a repository and stores them in an RDDS.

#### CSLURP10

CSLURP10 takes the contents of a system or non-system RDDS and writes these definitions to an IMSRSC repository. This utility can be used to initially populate an IMSRSC repository and update definitions at some later time. Because CSLURP10 uses RM to communicate with the repository, an SCI address space on the LPAR where the utility is being run and an RM address space must be available. Example 7-23 shows a job that runs the CSLURP10 utility to generate an IMSRSC repository.

**Example 7-23 JCL to populate a repository with stored definitions within an RDDS**

```plaintext
//RDDS2RPO JOB,USER,CLASS=A,MSGCLASS=X,NOTIFY=USER
//*
//JOBLIB DD DSN=IMSTESTL.TNUC0,DISP=SHR
//*
//STEP1 EXEC PGM=CSLURP10,MEMLIMIT=4G
//SYSUDUMP DD SYSOUT=*  
//SYSPRINT DD SYSOUT=*  
//*
//***************************************************************/
//* SPECIFY A VALID RDDS DSN FOR INPUT */
//***************************************************************/
//RDDSRSN DD DSN=IMSTESTG.NONSYS.IMSRDDS1,DISP=SHR
//*
//***************************************************************/
//* IMSID MAY BE SPECIFIED ON SYSIN OR DEFAULT TO THE IMSID */
//* ON THE RDDS HEADER RECORD. SUBSTITUTE THE SYSIN STATEMENT */
//* BELOW TO CHANGE BEHAVIOUR */
//***************************************************************/
//SYSIN DD *
IMSPLEX(NAME=PLEX1 IMSID(SYS3,IMS2,IMS3))
//*
```

#### CSLURP20

CSLURP20 generates a non-system RDDS from an IMSRSC repository. It can be used for backup or fallback. Because CSLURP20 uses RM to communicate with the repository, an SCI address space and an RM address space must be available. Example 7-24 shows a job that runs the CSLURP20 utility to generate an RDDS.

**Example 7-24 JCL to populate an RDDS with stored definitions within an IMSRSC repository**

```plaintext
//RPO2DDS JOB,USER,CLASS=A,MSGCLASS=X,NOTIFY=USER
//*
//JOBLIB DD DSN=IMSTESTL.TNUC0,DISP=SHR
//*
//STEP1 EXEC PGM=CSLURP20
//SYSUDUMP DD SYSOUT=*  
//SYSPRINT DD SYSOUT=*  
```
7.8.3 Direct access without RM using the batch ADMIN utility

You can also use the repository batch ADMIN utility (called FRPBATCH) to help manage IMSRSC repositories. Eight commands are provided. This utility does not use RM to access the repository; instead it accesses the repository directly as a batch job using XCF. This information about IMSRSC repositories is kept in the RS catalog repository.

The ADD, UPDATE, RENAME, and DELETE commands provide the capability to manage IMSRSC repository definitions. The DSCHANGE command provides the capability to change a data set disposition (used to set up new SPARE data sets). The LIST command provides a display of IMSRSC information. The START and STOP commands allow and prevent access to an IMSRSC repository.

Example 7-25 shows JCL that you can use to run the batch ADMIN utility. The commands are entered as part of the SYSIN DD statement.

Example 7-25 JCL that executes the ADD, START, and LIST functions of the batch ADMIN utility

```
//FRPBAT EXEC PGM=FRPBATCH,PARM='XCFGROUP=IM12XREP'
//SYSPRINT DD SYSOUT=*  
//SYSIN DD *
ADD REPOSITORY(IMS12XRP) 
  REPDNS1RID(IMS1Q.IMS12X.REPO.IMSPRI.RID) + 
  REPDNS1RMD(IMS1Q.IMS12X.REPO.IMSPRI.RMD) + 
  REPDNS2RID(IMS1Q.IMS12X.REPO.IMSSEC.RID) + 
  REPDNS2RMD(IMS1Q.IMS12X.REPO.IMSSEC.RMD) + 
  REPDNS3RID(IMS1Q.IMS12X.REPO.IMSSPR.RID) + 
  REPDNS3RMD(IMS1Q.IMS12X.REPO.IMSSPR.RMD) + 
  AUTOOPEN(NO) 

START REPOSITORY(IMS12XRP) MAXWAIT(30,CONTINUE) 

LIST REPOSITORY(IMS12XRP) 
/*
7.8.4 Direct repository access using z/OS modify interface

Several RS commands are available to help you manage the RS and IMSRSC repositories. You enter these commands by using the z/OS modify interface:

- **ADMIN** (administrative functions for IMSRSC repositories: change data set disposition, display data sets, and start and stop repositories)
- **AUDIT** (dynamically turn auditing on or off)
- **SECURITY** (refresh in-storage RACF profile definitions)
- **SHUTDOWN** (shut down RS address spaces)
- **STOP** (stop or shut down RS)

7.9 Using the repository in business operations

Now that you understand the repository infrastructure and required setup, the focus shifts to how to use the repository in your business operations.

7.9.1 IMS repository commands

First, we focus on commands and describe the following topics:

- IMS and RM IMSplex commands issued from SPOC
- Security considerations for commands
- Batch ADMIN commands
- Repository Server commands issued through z/OS modify interface
- Repository commands comparison

**IMS and RM IMSplex commands issued from SPOC**

This section describes the new and enhanced commands in IMS 12 that apply to repository DRD. These commands are “IMSplex” commands (also known as *type-2 commands*) that can be issued to one or more IMS systems in an IMSplex through an OM interface. This section explains when it is appropriate to issue each command, describes how each command is useful in different scenarios, and shows the syntax for reference.

For more information about a particular command beyond what is explained here, such as available parameter values and their meanings, see *IMS Version 12 Commands, Volume 1: IMS Commands A-M*, SC19-3009, or *IMS Version 12 Commands, Volume 2: IMS Commands N-V*, SC19-3010.

**UPDATE RM command**

Use the new **UPDATE RM** command to dynamically enable the RM address space to use the repository, or change the audit access level that was originally specified in the FRPCFG member. Example 7-26 shows the command syntax.

*Example 7-26  UPDATE RM command syntax*

```plaintext
UPDATE RM TYPE(REPO) REPOTYPE(IMSRSC) SET(REPO(Y|N) AUDITACCESS())
```

Notice that the command must be issued with **TYPE(REPO)** and **REPOTYPE(IMSRSC)** to indicate that we want to target a repository of type "IMSRSC". These are the only valid values for these parameters.
To dynamically enable RM for repository usage, specify the SET(REPO(Y)) parameter. Then command processing rereads and reprocesses the REPOSITORY section within the CSLRIxxx member. This is why it is important to add repository definitions to the CSLRIxxx member before issuing this command. Also during command processing, RM registers to the RS if RM is not already registered. RM connects to the repository name specified in the REPOSITORY section of the CSLRIxxx member.

If the command is successful at the command master RM, the command master RM communicates the changes to other active RMs in the IMSplex. All RMs in the IMSplex will have the same repository settings. If RM is defined to use the resource structure, the command master RM will update the resource structure with the repository name and repository type that it is connected to. Subsequent RMs that are restarted after the change will ensure that they are connected to the same repository name and repository type as read from the resource structure.

If the RM repository usage is already enabled, you can dynamically disable it by specifying SET(REPO(N)). The CSLRIxxx member is not reread or reprocessed in this case like it is when SET(REPO(Y)) is specified. Therefore, as part of the repository disabling process, you can remove the repository definitions from CSLRIxxx either before or after UPDATE RM is issued with SET(REPO(N)). If the repository definitions specified on the REPOSITORY= statement are still present in CSLRIxxx, any RMs that start after the UPDATE RM … SET(REPO(N)) command is issued will reconnect to the repository during RM startup and enable the IMSRSC repository at all RMs in the IMSplex.

With the AUDITACCESS() parameter, you can dynamically change the audit level settings and override what was originally specified on the AUDIT_DEFAULT parameter in the FRPCFG member. Example 7-27 shows an UPDATE RM command being issued to change the AUDITACCESS parameter value to read.

Example 7-27 UPDATE RM command dynamically updating the AUDITACCESS setting

<table>
<thead>
<tr>
<th>RepositoryType</th>
<th>MbrName</th>
<th>CC</th>
<th>RepositoryName</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMSRSC</td>
<td>RMXRM</td>
<td>0</td>
<td>IMS12XRP</td>
</tr>
</tbody>
</table>

**QUERY RM command**

Issue the QUERY RM command to determine whether an RM address space is enabled for repository usage, or to view the audit access level or status of RM. The RM can have one of the following statuses:

- **CONNECTED**
  - RM is connected to the IMSRSC repository and the repository is available for use.

- **CONNECT-INCOMPLETE**
  - RM successfully connected to the repository, but RM failed to correctly update the repository global entry in the resource structure.

- **DISCONNECT-INCOMPLETE**
  - RM successfully disconnected from the repository, but RM failed to correctly update the repository global entry in the resource structure.

- **NOTAVAIL**
  - The repository is not available.

- **RS-NOTAVAIL**
  - No master RS is available.

- **SPARERECOV**
  - The repository spare recovery process is in progress.

- **SPARERCVERR**
  - The repository spare recovery process resulted in an error and the repository is not available for use.
UPDATE-AUDACC-INCOMPLETE

RM successfully updated the audit access setting in the repository, but RM failed to correctly update the repository global entry in the resource structure.

For information about the repository spare recovery process, see “Recovery activities” on page 257.

Example 7-28 shows the QUERY command syntax.

Example 7-28 QUERY RM command syntax

```
QUERY RM TYPE(REPO) SHOW(ATTRIB|STATUS|ALL)
```

You can filter the information associated with RM by specifying the SHOW() parameter accordingly. For example, display only the repository attributes in RM by using the command shown in Example 7-29. The only attribute value that is displayed when the SHOW(ATTRIB) parameter is applied is the audit access level.

Example 7-29 Displaying the audit access setting using the QUERY RM command

Response for: QUERY RM TYPE(REPO) SHOW(ATTRIB)

```
RepositoryType MbrName    CC AuditAccess RepositoryName
IMSRSC         RMXRM       0 READ        IMS12XRP
```

Alternatively, you can choose to display only the status of RM using the command shown in Example 7-30.

Example 7-30 Displaying the status of RM using the QUERY RM command

Response for: QUERY RM TYPE(REPO) SHOW(STATUS)

```
RepositoryType MbrName    CC Status    RepositoryName
IMSRSC         RMXRM       0 CONNECTED IMS12XRP
```

Lastly, you can display the audit access setting and the status, the name of the IMSRSC repository with which RM is associated, and the repository group in which it is contained. See Example 7-31.

Example 7-31 Displaying all information associated with RM using the QUERY RM command

Response for: QUERY RM TYPE(REPO) SHOW(ALL)

```
RepositoryType MbrName    CC Status    AuditAccess RepositoryName RepositoryGroup
IMSRSC         RMXRM       0 CONNECTED READ        IMS12XRP       IM12XREP
```

**QUERY RM command:** The QUERY RM command can be issued to determine whether RM is repository-enabled in the IMSplex. For example, you can issue this command after an UPDATE RM command is issued to dynamically enable it for repository usage. If the UPDATE RM command failed, you know that RM was not successfully repository-enabled by observing the non-zero UPDATE command completion code. However, keep in mind that you will not know if the UPDATE RM command failed at non-master RMs that received the broadcast message to enable the repository. Because this information is not part of the command response, issuing the QUERY RM command shows whether all RMs are connected to or disconnected from the repository.

For detailed information about the input and output of the QUERY RM command, see *IMS Version 12 Commands, Volume 2: IMS Commands N-V*, SC19-3010.
**UPDATE IMS command**

You can dynamically enable an IMS system to use the repository by issuing the **UPDATE IMS** command, or use it to disable the automatic export capability. Example 7-32 shows the command syntax.

**XRF:** In an XRF environment, the **UPDATE IMS** command is processed both at the XRF active IMS and the XRF alternate IMS to dynamically enable and disable usage of repository.

**Example 7-32   UPDATE IMS command syntax for DRD-related functions**

```
UPDATE IMS SET(LCLPARM())
```

**Important:** Before you issue this command to dynamically enable IMS for repository usage, you must add repository definitions to the DFSDFxxx member and RM must already be repository-enabled.

To dynamically disable automatic export for an IMS system, issue the **UPDATE IMS** command with the **LCLPARM(AUTOEXPORT(N))** parameter included. You take this step after migration to the repository has been completed to reduce I/O overhead that occurs with autoexport.

**IMS cold start:** After automatic export is disabled with this command, an IMS cold start is required to re-enable it.

**QUERY IMS command**

To determine whether an IMS system is enabled for repository usage, issue the **QUERY IMS** command. You can also use this command to determine whether automatic export is enabled for an IMS system. These enhancements were added to existing **QUERY IMS** functionality, essentially to display the information about the possible capabilities that were added with the **UPDATE IMS** command (dynamically enabling repository usage, automatic export, or both).

Example 7-33 shows the command syntax.

**Example 7-33   QUERY IMS command syntax**

```
QUERY IMS TYPE(LCLPARM) SHOW()
```

The **SHOW()** parameter can be used to filter output shown in the command response. For example, Example 7-34 shows the **QUERY IMS** command being issued to display whether the IMS systems in an IMSplex are enabled for repository usage in addition to other information about the repository.

**Example 7-34   Showing repository attributes using the QUERY IMS command**

```
Response for: QUERY IMS TYPE(LCLPARM) SHOW(REPO)
MbrName | CC | RepositoryType | RepositoryName | LastExportTime
--------|----|----------------|----------------|----------------
I12A     | 0  | IMSRSC         | IMS12XRP       |                
I12B     | 0  | IMSRSC         | IMS12XRP       |                
I12C     | 0  | IMSRSC         | IMS12XRP       |                
I12D     | 0  | IMSRSC         | IMS12XRP       | 2011.235 09:12:01.56
```

**Important:** Before you issue this command to dynamically enable IMS for repository usage, you must add repository definitions to the DFSDFxxx member and RM must already be repository-enabled.

**IMS cold start:** After automatic export is disabled with this command, an IMS cold start is required to re-enable it.

**XRF:** In an XRF environment, the **UPDATE IMS** command is processed both at the XRF active IMS and the XRF alternate IMS to dynamically enable and disable usage of repository.

**Important:** Before you issue this command to dynamically enable IMS for repository usage, you must add repository definitions to the DFSDFxxx member and RM must already be repository-enabled.

To dynamically disable automatic export for an IMS system, issue the **UPDATE IMS** command with the **LCLPARM(AUTOEXPORT(N))** parameter included. You take this step after migration to the repository has been completed to reduce I/O overhead that occurs with autoexport.

**IMS cold start:** After automatic export is disabled with this command, an IMS cold start is required to re-enable it.

**QUERY IMS command**

To determine whether an IMS system is enabled for repository usage, issue the **QUERY IMS** command. You can also use this command to determine whether automatic export is enabled for an IMS system. These enhancements were added to existing **QUERY IMS** functionality, essentially to display the information about the possible capabilities that were added with the **UPDATE IMS** command (dynamically enabling repository usage, automatic export, or both).

Example 7-33 shows the command syntax.

**Example 7-33   QUERY IMS command syntax**

```
QUERY IMS TYPE(LCLPARM) SHOW()
```

The **SHOW()** parameter can be used to filter output shown in the command response. For example, Example 7-34 shows the **QUERY IMS** command being issued to display whether the IMS systems in an IMSplex are enabled for repository usage in addition to other information about the repository.

**Example 7-34   Showing repository attributes using the QUERY IMS command**

```
Response for: QUERY IMS TYPE(LCLPARM) SHOW(REPO)
MbrName | CC | RepositoryType | RepositoryName | LastExportTime
--------|----|----------------|----------------|----------------
I12A     | 0  | IMSRSC         | IMS12XRP       |                
I12B     | 0  | IMSRSC         | IMS12XRP       |                
I12C     | 0  | IMSRSC         | IMS12XRP       |                
I12D     | 0  | IMSRSC         | IMS12XRP       | 2011.235 09:12:01.56
```
Example 7-35 shows another sample of the QUERY IMS command, where the SHOW() parameter is specified to display whether automatic export is enabled for the IMS systems.

**Example 7-35  Showing the automatic export setting using the QUERY IMS command**

Response for: `QUERY IMS TYPE(LCLPARM) SHOW(AUTOEXPORT)`

<table>
<thead>
<tr>
<th>MbrName</th>
<th>CC</th>
<th>AutoExport</th>
</tr>
</thead>
<tbody>
<tr>
<td>I12A</td>
<td>0</td>
<td>Y</td>
</tr>
<tr>
<td>I12B</td>
<td>0</td>
<td>Y</td>
</tr>
<tr>
<td>I12C</td>
<td>0</td>
<td>Y</td>
</tr>
<tr>
<td>I12D</td>
<td>0</td>
<td>Y</td>
</tr>
</tbody>
</table>

To display all possible information about the repository, included with other IMS attribute settings, you can specify the SHOW(ALL) parameter with the QUERY IMS command. For example command output, see *IMS Version 12 Commands, Volume 2: IMS Commands N-V*, SC19-3010.

**Tip:** After an UPDATE IMS command is issued to dynamically enable repository usage, you can confirm that the IMS is now repository-enabled by issuing the QUERY IMS command. If IMS cannot be enabled for repository usage, you detect this problem by observing the non-zero UPDATE command completion code.

**QUERY for resources and descriptors**

The QUERY command has been enhanced to show IMS resource and descriptor definitions (names and attribute values) in the repository and at each IMS system. The command shows the stored repository and runtime IMS resource definition information. In addition, the QUERY command now shows specific IMS IDs that have the resource names defined. Example 7-36 highlights the command syntax for the QUERY command.

**Example 7-36  Syntax for displaying attribute values for resources or descriptors**

```
QUERY rsc-type | desc-type NAME() SHOW()
```

To display both repository and local IMS resource definitions using the QUERY command, specify the SHOW(DEFN) parameter (Example 7-37). In this example, we filtered the output to display the attribute values indicating the type of access to the database and the local runtime value for the resident option. The repository definitions can be distinguished from the IMS runtime definitions by the presence of a “Y” in the “Repo” column. The generic definitions that apply to all IMS systems have a Y in the Repo column and blank in the IMSid column. The IMS-specific definitions have the IMS ID in the IMSid column both for the repository and runtime definitions.

**Example 7-37  Displaying both repository and local IMS definition information with QUERY**

Response for: `QUERY DB NAME(*) SHOW(DEFN,ACCTYPE,RESIDENT)`

<table>
<thead>
<tr>
<th>DBName</th>
<th>MbrName</th>
<th>CC</th>
<th>Repo</th>
<th>IMSid</th>
<th>TYPE</th>
<th>Acc</th>
<th>LAcc</th>
<th>Rsnt</th>
<th>LDRsdnt</th>
<th>LRsdnt</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTO12</td>
<td>I12A</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>UPD</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AUTO12</td>
<td>I12B</td>
<td>0</td>
<td>I12B</td>
<td>DL/I</td>
<td>UPD</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AUTO12</td>
<td>I12C</td>
<td>0</td>
<td>I12C</td>
<td>DL/I</td>
<td>UPD</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AUTO12</td>
<td>I12D</td>
<td>0</td>
<td>I12D</td>
<td>DL/I</td>
<td>READ</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBFSAMD1</td>
<td>I12A</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>EXCL</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBFSAMD1</td>
<td>I12A</td>
<td>0</td>
<td>I12A</td>
<td></td>
<td>EXCL</td>
<td>Y</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBFSAMD1</td>
<td>I12B</td>
<td>0</td>
<td>I12B</td>
<td></td>
<td>EXCL</td>
<td>Y</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBFSAMD1</td>
<td>I12C</td>
<td>0</td>
<td>I12C</td>
<td></td>
<td>EXCL</td>
<td>Y</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBFSAMD1</td>
<td>I12D</td>
<td>0</td>
<td>I12D</td>
<td></td>
<td>EXCL</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
To only display IMS resource definitions that are within the repository, include the `SHOW(DEFN,GLOBAL)` parameter in the `QUERY` command (Example 7-38). In the example, we have again chosen to filter the output to display the access type and resident attribute values.

**Example 7-38  Displaying only repository IMS definition information with QUERY**

<table>
<thead>
<tr>
<th>DBName</th>
<th>MbrName</th>
<th>CC</th>
<th>Repo</th>
<th>IMSid</th>
<th>Acc</th>
<th>Rsnt</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTODB</td>
<td>I12A</td>
<td>0</td>
<td>Y</td>
<td>I12A</td>
<td>UPD</td>
<td>N</td>
</tr>
<tr>
<td>DBFSAMD1</td>
<td>I12A</td>
<td>0</td>
<td>Y</td>
<td>EXCL</td>
<td>N</td>
<td>N</td>
</tr>
</tbody>
</table>

Alternatively, to display only IMS resource definitions local to IMS, include the `SHOW(DEFN,LOCAL)` parameter in the `QUERY` command (Example 7-39). These definitions are specific to each IMS system shown in the MbrName and IMSid columns in the output. To maintain consistency with the previous two examples, we have again chosen to filter the output to display the access type and resident attribute values.

**Example 7-39  Displaying only local IMS definition information with QUERY**

<table>
<thead>
<tr>
<th>DBName</th>
<th>AreaName</th>
<th>PartName</th>
<th>MbrName</th>
<th>CC</th>
<th>IMSid</th>
<th>TYPE</th>
<th>LAcc</th>
<th>LDRsdnt</th>
<th>LRsdnt</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTODB</td>
<td>I12A</td>
<td></td>
<td>I12A</td>
<td>0</td>
<td>I12A</td>
<td>DL/I</td>
<td>UPD</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>AUTODB</td>
<td>I12B</td>
<td></td>
<td>I12B</td>
<td>0</td>
<td>I12B</td>
<td>DL/I</td>
<td>UPD</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>AUTODB</td>
<td>I12C</td>
<td></td>
<td>I12C</td>
<td>0</td>
<td>I12C</td>
<td>DL/I</td>
<td>UPD</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>AUTODB</td>
<td>I12D</td>
<td></td>
<td>I12D</td>
<td>0</td>
<td>I12D</td>
<td>DL/I</td>
<td>READ</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>DBFSAMD1</td>
<td>I12A</td>
<td></td>
<td>I12A</td>
<td>0</td>
<td>I12A</td>
<td>EXCL</td>
<td>Y</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>DBFSAMD1</td>
<td>I12B</td>
<td></td>
<td>I12B</td>
<td>0</td>
<td>I12B</td>
<td>EXCL</td>
<td>Y</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>DBFSAMD1</td>
<td>I12C</td>
<td></td>
<td>I12C</td>
<td>0</td>
<td>I12C</td>
<td>EXCL</td>
<td>Y</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>DBFSAMD1</td>
<td>I12D</td>
<td></td>
<td>I12D</td>
<td>0</td>
<td>I12D</td>
<td>EXCL</td>
<td>Y</td>
<td>N</td>
<td>N</td>
</tr>
</tbody>
</table>

If you want to see a list of IMS systems that have the resources specified in the command defined to them, include the `SHOW(IMSID)` parameter in the `QUERY` command. Example 7-40 shows a command that includes this filter.

**Example 7-40  Displaying a list of IMS systems that have a specific resource defined with QUERY**

<table>
<thead>
<tr>
<th>DBName</th>
<th>MbrName</th>
<th>CC</th>
<th>Repo</th>
<th>IMSid</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTODB</td>
<td>I12A</td>
<td>0</td>
<td>Y</td>
<td>I12A</td>
</tr>
<tr>
<td>AUTODB</td>
<td>I12A</td>
<td>0</td>
<td>Y</td>
<td>I12B</td>
</tr>
<tr>
<td>AUTODB</td>
<td>I12A</td>
<td>0</td>
<td>Y</td>
<td>I12D</td>
</tr>
<tr>
<td>AUTODB</td>
<td>I12A</td>
<td>0</td>
<td>Y</td>
<td>I12Z</td>
</tr>
</tbody>
</table>

Finally, to display all IMSIDs that have the specified resource defined, a list of the repository resource definitions and any IMS-specific definitions, include the `SHOW(DEFN,IMSID)` parameter as shown in Example 7-41. This example assumes that APAR PM41761 has been applied.

**Example 7-41  Displaying all IMSIDs that have a specific resource defined with QUERY**

<table>
<thead>
<tr>
<th>DBName</th>
<th>MbrName</th>
<th>CC</th>
<th>Repo</th>
<th>IMSid</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTODB</td>
<td>I12A</td>
<td>0</td>
<td>Y</td>
<td>I12A</td>
</tr>
<tr>
<td>AUTODB</td>
<td>I12A</td>
<td>0</td>
<td>Y</td>
<td>I12B</td>
</tr>
<tr>
<td>AUTODB</td>
<td>I12A</td>
<td>0</td>
<td>Y</td>
<td>I12C</td>
</tr>
<tr>
<td>AUTODB</td>
<td>I12A</td>
<td>0</td>
<td>Y</td>
<td>I12D</td>
</tr>
</tbody>
</table>

To maintain consistency with the previous three examples, we have again chosen to filter the output to display the access type and resident attribute values. In the example command output, you can see that the AUTODB database is defined in the repository for IMSIDs I12B, I12D, and I12Z as a stored resource definition. It is also defined at active IMS systems I12A, I12B, I12C, and I12D as a runtime resource definition. The AUTODB database is not defined in the repository as a stored definition for IMS systems I12A and I12C even though they have a runtime definition in these systems.
Example 7-41   Displaying repository and local IMS definitions with IMSIDs using QUERY

<table>
<thead>
<tr>
<th>DBName</th>
<th>MbrName</th>
<th>CC</th>
<th>Repo</th>
<th>IMSid</th>
<th>TYPE</th>
<th>Acc</th>
<th>LAcc</th>
<th>Rsdent</th>
<th>LRsdnt</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTODB</td>
<td>I12A</td>
<td>0</td>
<td>Y</td>
<td></td>
<td>UPD</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AUTODB</td>
<td>I12A</td>
<td>0</td>
<td>Y</td>
<td>I12B</td>
<td>UPD</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AUTODB</td>
<td>I12A</td>
<td>0</td>
<td>Y</td>
<td>I12D</td>
<td>UPD</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AUTODB</td>
<td>I12A</td>
<td>0</td>
<td>Y</td>
<td>I12Z</td>
<td>UPD</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AUTODB</td>
<td>I12A</td>
<td>0</td>
<td></td>
<td>I12A</td>
<td>DL/I</td>
<td>UPD</td>
<td>N</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AUTODB</td>
<td>I12A</td>
<td>0</td>
<td></td>
<td>I12B</td>
<td>DL/I</td>
<td>UPD</td>
<td>N</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AUTODB</td>
<td>I12A</td>
<td>0</td>
<td></td>
<td>I12C</td>
<td>DL/I</td>
<td>UPD</td>
<td>N</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AUTODB</td>
<td>I12A</td>
<td>0</td>
<td></td>
<td>I12D</td>
<td>DL/I</td>
<td>UPD</td>
<td>N</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**EXPORT command**

Issue the EXPORT command when resources or descriptors have been either created or updated, and they need to be hardened to the repository. If hardening changes to offline stored definitions is part of your change management process, use the EXPORT command when definitional changes occur or at regular intervals during operations.

**Important:** In an XRF environment, the XRF active IMS can export its runtime resource definitions to the repository using the EXPORT command, where the XRF alternate IMS cannot. The XRF alternate IMS will only be able to export its runtime resource definitions to the repository after it takes over for the active IMS. You can, however, update the XRF active and alternate IMS systems' stored resource definitions within the repository by issuing an EXPORT command, specifying both of their IMSIDs on the SET(IMSID()) parameter.

The EXPORT command is processed by a single command master IMS and will write valid resources and descriptors to the repository. Example 7-42 shows the command syntax.

**Example 7-42   EXPORT command syntax**

```
EXPORT DEFN TARGET(REPO) TYPE() NAME() STARTTIME() ENDTIME() SET(IMSID()) OPTION()
```

The EXPORT command can also use data included in QUERY command output. If the QUERY command is issued with the SHOW(TIMESTAMP) parameter included, you can determine the exact time that a resource was created or updated. See the LTimeCreate column header output in Example 7-43.

**Example 7-43   Displaying the time a resource was created with QUERY and SHOW(TIMESTAMP)**

<table>
<thead>
<tr>
<th>PgmName</th>
<th>MbrName</th>
<th>CC</th>
<th>LRgnType</th>
<th>LTimeCreate</th>
</tr>
</thead>
</table>

You are then able to use those timestamp values for the EXPORT STARTTIME() parameter, the ENDTIME() parameter, or both. The STARTTIME() and ENDTIME() parameters can be as specific as tenths and hundredths of a second, and matches the timestamp granularity displayed in QUERY SHOW(TIMESTAMP) command output, so copying the exact values is facilitated in this way. The STARTTIME() and ENDTIME() parameters are optional.
Chapter 7. IMS repository

Example 7-44 shows the EXPORT command with the timestamp information shown in Example 7-43 on page 236. This command ensures that all programs that are created after the specified timestamp are exported to the repository.

Example 7-44  Exporting a resource to the repository

Response for: EXPORT DEFN TARGET(RePO) TYPE(PGM) NAME(*) STARTTIME(2011.242
22:23:13.36)
Name     Type     MbrName    CC
I12A     LIST     I12A        0
PGMADD   PGM      I12A        0

Next, you can indicate which IMS resource lists in the repository should be exported to by listing them on the SET(IMSID()) parameter. This is an optional parameter and if omitted, the runtime resource definitions of the command master IMSID are exported for the command master IMS only. You can control which IMS is selected as command master by using the ROUTE capability of the OM interface from which you are entering the command.

Wildcards are supported for the IMSID. However, the EXPORT command with a wildcard for the IMSID will fail if no IMS resource lists are in the repository that matches the wildcard name. When a specific IMSid is specified, the IMS resource list for the IMS is created if it does not exist and the resource definitions are written to the repository.

When exporting a transaction resource to the repository with a type-2 EXPORT command, ensure that the program resource associated with this transaction already exists within the repository. Otherwise, the EXPORT command fails with a completion code such as the one shown in Example 7-45.

Example 7-45  Failed export attempt due to a missing program resource

Response for: EXPORT DEFN TARGET(RePO) TYPE(TRAN) NAME(TRANINFO)
Name     Type     MbrName    CC CCText
I12A     LIST     I12A      1D0 NOT DONE DUE TO ERROR
TRANINFO TRAN     I12A       67 NO PGM() DEFINED

If this occurs, determine which program should exist in the repository by querying the transaction with a command such as the QUERY TRAN command (Example 7-46).

Example 7-46  Query transaction resource to determine associated program

Response for: QUERY TRAN NAME(TRANINFO) SHOW(PGM)
Trancode MbrName    CC LPSBname
TRANINFO I12D        0 PGMINFO

Lastly, attempt the EXPORT command a second time using a command such as the one shown in Example 7-47.

Example 7-47  Successful export of resources to repository

Response for: EXPORT DEFN TARGET(RePO) TYPE(PGM TRAN) NAME(PGMINFO TRANINFO)
Name     Type     MbrName    CC
I12D     LIST     I12D        0
PGMINFO  PGM      I12D        0
TRANINFO TRAN     I12D        0
A benefit of using repository DRD is the ability to define the resource definitions for an inactive IMS or a brand-new IMS. When the IMS restarts, it reads these stored definitions that were created while it was inactive.

Lastly, when issuing the **EXPORT** command, you can optionally display a line of output for each successfully exported resource and optionally export only the resources and descriptors that had definitional changes since the last **EXPORT** command was issued. Automatic export is not possible when using DRD with the repository. However, by issuing the **EXPORT** command with the **OPTION(CHANGESONLY)** parameter at regular intervals, you can ensure that definitional changes are captured and hardened to the repository.

The **EXPORT** command writes all of the definitions to the repository as a single unit of work, as is the case with RDDS DRD. In either case (that is, using DRD with the repository or with an RDDS), the export fails if one resource is in error, and no other resources will be exported. The difference between these two DRD types is that, when exporting to the repository, only one command master IMS processes the command. This command master IMS creates and updates the definitions for each IMSID specified in the **SET(IMSID())** parameter. With RDDS DRD, each IMS that receives the command will process it. In this case, the command can succeed on some systems and fail on other systems.

In some cases, resources or descriptors cannot be exported, such as when they are IMS system-defined descriptors or HALDB partitions or have invalid names. Validation is done by IMS before export occurs. Validation of resource attributes between associated resources such as transactions and programs, or transactions and routing codes, or routing codes and programs, is performed by RM before the resource definitions are written to the repository. For example, a transaction cannot be written if the associated program does not exist in the repository.

The **EXPORT** command should be used in different ways depending on whether you have a cloned or non-cloned IMS environment. Specifically, in a cloned environment it is recommended to always specify the **SET(IMSID())** parameter in the **EXPORT** command to ensure that all IMS resource lists are exported to. This keeps the definitions in each of the cloned resource lists for IMS consistent with one another. However, in a non-cloned environment, the IMS systems might have differing attribute values for their resources. Therefore, omit the **SET(IMSID())** parameter so that each IMS resource list can be updated one at a time, separately, and the IMS-specific attribute values can be maintained. When omitting the **SET(IMSID())** parameter, remember to route the command to the correct IMS. Because no IMS-specific information is in the command syntax, you can easily reissue this command without modifying the command itself.

For best performance, export only the resources and descriptors that have been changed since they were last hardened to the repository with an **EXPORT** command. You can more easily pinpoint the changed resource or descriptors by including the **OPTION(CHANGESONLY)** parameter. Another method of minimizing the total number of resources or descriptors to be exporting is including the **STARTTIME()** parameter to target only those that have been created or updated after the specified time.

**DELETE DEFN command**

Issue the **DELETE DEFN** command to delete stored resource definitions for one or more IMS systems in the repository. If runtime resource definitions have been deleted from an online IMS system with the **DELETE** command, these deletes can be hardened to the repository with the **DELETE DEFN** command. In this situation, make sure that you specify the same resources or descriptors in this command whose runtime definitions were deleted from the online system.
To delete stored definitions for specific IMS systems that have resource definitions defined in the IMSRSC repository, specify the IMSIDs associated with the IMS resource lists with the `FOR(IMSID())` parameter. As mentioned before, it is appropriate to issue this command when you want to harden runtime definition deletes to the repository. It is recommended that you first delete the runtime resource definitions using the `DELETE` commands at the IMS systems before using the `DELETE DEFN` command to delete the resource definitions from the IMSRSC repository. Example 7-48 shows the command syntax.

**Example 7-48**  **DELETE DEFN command syntax**

```
DELETE DEFN TARGET(REPO) TYPE() NAME() FOR(IMSID()) OPTION()
```

A series of commands is shown in Example 7-49. Assume that the first three commands shown are routed to IMS1 and IMS2. Looking at the example, a program is first queried to determine whether any work in progress exists for it on either IMS system. Then, the scheduling is stopped for the program to prevent any new work in progress from occurring. The example continues to show that the program is deleted from two online IMS systems with the `DELETE` command, then deleted from the IMS resource lists associated with these two systems in the offline repository with the `DELETE DEFN` command.

**Example 7-49**  **Command sequence for deleting runtime and stored IMS resource definitions**

```
QUERY PGM NAME(PGM1) SHOW(WORK)
UPDATE PGM NAME(PGM1) STOP(SCHD)
DELETE PGM NAME(PGM1)
DELETE DEFN TARGET(REPO) TYPE(PGM) NAME(PGM1) FOR(IMSID(IMS1,IMS2))
```

Wildcard support exists for the `FOR(IMSID())` parameter. When issuing the `DELETE DEFN` command with the `NAME(*)` parameter, you can ensure that a line of output is displayed for each resource or descriptor that was processed by including the `OPTION(ALLRSP)` parameter.

**Attention:** Use the `NAME(*)` parameter with care because it can delete all of the resource definitions in the IMSRSC repository and IMS will be restarted with no resources.

**IMPORT command**

The `IMPORT` command reads stored definitions that exist in the repository into running IMS systems. This command can be used if an IMS is restarted with no resources defined to populate the control region with runtime resource definitions. Alternatively, if changes were made to the repository offline and you want to roll the changes to the systems in IMSplex, the `IMPORT` command can be used to accomplish this. An example of when this scenario might arise is when you use the “RDDS to Repository” (`CSLURP10`) utility, introduced in 7.8.2, “Offline access through RM utilities” on page 228, to write resource definitions to the IMSRSC repository that have not been read into any IMS system yet.

Example 7-50 shows the `IMPORT` command syntax.

**Example 7-50**  **IMPORT command syntax**

```
IMPORT DEFN SOURCE() TYPE() NAME() OPTION() SCOPE()
```
When importing stored definitions from the repository, make sure that the `SOURCE(REPO)` parameter is specified so the repository is the data set that is read. Also, ensure that you indicate which resources should be imported using the `TYPE()` and `NAME()` parameters.

You can control the output displayed by the `IMPORT` command by specifying the `OPTION()` parameter accordingly. The `OPTION(ABORT)` and `OPTION(ALLRSP)` parameters used with RDDS DRD import before IMS 12 are now also used with repository DRD import.

The `OPTION(ABORT)` parameter terminates command processing if an error occurs during import. The `OPTION(ALLRSP)` parameter returns a line of output for each resource that was exported and is valid if the `NAME(*)` parameter was also specified in the command. If individual names are specified for the `NAME()` parameter, a line of output is also returned for each resource in this case.

A new `OPTION(UPDATE)` parameter has been added for the `IMPORT` command. With this parameter, an existing runtime resource definition can be updated with a stored resource definition being imported from either the RDDS or the repository.

**Attention:** The `OPTION(UPDATE)` parameter is not the default and must be explicitly specified to update a runtime resource definition with a stored resource definition.

The ROUTE capability of the OM API is used to route commands to specific IMS systems. `ROUTE=ALL` is recommended when the `SCOPE(ALL)` parameter is included. If a ROUTE list is specified (other than `ROUTE=ALL`), the command is processed only by the IMS systems in the list that receives the command. Other IMS systems that have the resources defined but are not included in the ROUTE list will not receive the command and therefore will not be synchronized with the repository.

In IMS 12, the `SCOPE(ALL)` and `SCOPE(ACTIVE)` parameters are the same. The `SCOPE(ACTIVE)` parameter applies the import to only the active IMS systems. The definitions of any inactive IMS system are not synchronized with the definitions of the other IMS systems in the IMSplex when it warmstarts or emergency restarts. To reestablish synchronization, you can issue an `IMPORT` command to import the resources that the other active IMS systems imported while it was inactive. If the inactive IMS is restarted, it is synchronized with the other IMS systems because it reads its entire IMS resource list.

**Security considerations for commands**

If you have set up security in your environment to restrict access to type-2 commands, you know that the RACF `OPERCMD$` class is used to contain resource profiles representing the commands. When you protect type-2 commands in the `OPERCMD$` class, the resource profiles have a certain format.

Table 7-1 shows the resource names associated with the new `DELETE DEFN`, `QUERY RM`, and `UPDATE RM` commands that must be added to the RACF `OPERCMD$` class to prevent unauthorized access. The required RACF permissions are also shown, and the IMSplex name must begin with the characters CSL. For information about restricting access to other elements that are part of the repository environment, see 7.10, “Security considerations” on page 258.

**Table 7-1 New type-2 command RACF definitions for OM**

<table>
<thead>
<tr>
<th>Command</th>
<th>Resource keyword</th>
<th>RACF access authority</th>
<th>Resource name</th>
</tr>
</thead>
<tbody>
<tr>
<td>DELETE</td>
<td>DEFN</td>
<td>UPDATE</td>
<td>IMS.CSLplxname.DEF.DEFN</td>
</tr>
<tr>
<td>QUERY</td>
<td>RM</td>
<td>READ</td>
<td>IMS.CSLplxname.QUERY.RM</td>
</tr>
<tr>
<td>UPDATE</td>
<td>RM</td>
<td>UPDATE</td>
<td>IMS.CSLplxname.UPDATE.RM</td>
</tr>
</tbody>
</table>
Batch ADMIN commands

Batch ADMIN commands are available to manage user repositories. The user repository data sets must be defined before the repository is started. The commands provide the ability to add a repository, list repository details such as status, data set names, start and stop repository for operations, rename or delete a repository, update repository-specific attributes, or change the disposition of the data sets of a repository.

As explained in this section, user repositories are defined to the catalog repository using the batch ADMIN ADD command and are started with the batch ADMIN START command.

Batch ADMIN commands focus on managing the individual user repository. In “Repository Server commands issued through z/OS modify interface” on page 248, you see that the z/OS modify interface commands have a similar function, but are geared toward managing the RS.

Batch ADMIN commands are available by starting the FRPBATCH utility with JCL statements. Table 7-2 summarizes the batch ADMIN commands that are available for managing a user repository.

Table 7-2  Batch ADMIN utility commands and their functions

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADD</td>
<td>Add a user repository to the RS catalog</td>
</tr>
<tr>
<td>UPDATE</td>
<td>Update a user repository definition in the RS catalog</td>
</tr>
<tr>
<td>RENAME</td>
<td>Rename an existing user repository in the RS catalog</td>
</tr>
<tr>
<td>DELETE</td>
<td>Remove a user repository from the RS catalog</td>
</tr>
<tr>
<td>DSCHANGE</td>
<td>Change data set disposition</td>
</tr>
<tr>
<td>LIST</td>
<td>List status information for all user repositories or detailed information for a single user repository</td>
</tr>
<tr>
<td>START</td>
<td>Request the RS to start a user repository</td>
</tr>
<tr>
<td>STOP</td>
<td>Request the RS to stop a user repository</td>
</tr>
</tbody>
</table>

**ADD command**

To define an IMSRSC repository to the RS catalog repository, execute the batch ADMIN ADD command using the syntax shown in Example 7-51. Here, you must specify the IMSRSC repository name in addition to the names of the IMSRSC repository primary and secondary index and member data sets. The remainder of the parameters are optional. The repository name is converted to uppercase if it is specified in lower or mixed case.

**Example 7-51  Syntax for the batch ADMIN utility ADD command**

```
ADD REPOSITORY(repository-name)
REPD1RID(primaryRID-name)
REPD1RMD(primaryRMD-name)
REPD2RID(secondaryRID-name)
REPD2RMD(secondaryRMD-name)
REPD3RID(NULL | spareRID-name)
REPD3RMD(NULL | spareRMD-name)
AUTOOPEN(NO | YES)
SECURITYCLASS(NULL | securityclassname)
```
Notice that you can optionally specify the spare repository index and member data set names here. If nothing is specified, there will be no spare because the default is null.

You can also control whether the repository data sets you are specifying with this command are opened when the repository is started (AUTOOPEN (YES)), which is the default, or when a user first connects to it (AUTOOPEN (NO)).

If you are going to be restricting access to the IMSRSC repository, specify the name of the 8-byte security class that will be used to restrict access here. This class overrides the SAF_CLASS= parameter value in the FRPCFG configuration member, if one was specified. Alternatively, if you want to deactivate repository security, you can specify the SECURITYCLASS(NULL) parameter on the batch ADMIN ADD command to accomplish this. For more information about setting up security for the IMS repository, see 7.10, “Security considerations” on page 258.

We used the JCL shown in Example 7-52 to define a user repository named IMS12XRP to the RS catalog repository.

Example 7-52  JCL to add a user repository to the RS catalog repository

```
/*
//REPOADD EXEC PGM=FRPBATCH,PARM='XCFGROUP=IM12XREP'
//STEPLIB DD DISP=SHR,DSN=IMS12Q.SDFSRESL
//SYSPRINT DD SYSOUT=*  
//SYSin DD * 
ADD REPOSITORY(IMS12XRP)                     +
   REPDSN1RID(IMS12Q.IM12X.REPO.IMSPRI.RID) +
   REPDSN1RMD(IMS12Q.IM12X.REPO.IMSPRI.RMD) +
   REPDSN2RID(IMS12Q.IM12X.REPO.IMSSEC.RID) +
   REPDSN2RMD(IMS12Q.IM12X.REPO.IMSSEC.RMD) +
   REPDSN3RID(IMS12Q.IM12X.REPO.IMSSPR.RID) +
   REPDSN3RMD(IMS12Q.IM12X.REPO.IMSSPR.RMD) +
   AUTOOPEN(YES)
/*
```

**UPDATE command**

Use the batch ADMIN UPDATE command to modify a user repository definition within the RS catalog data sets (specifically, to change the data sets, auto-open option or security class associated with a specific repository). Example 7-53 shows the syntax.

Example 7-53  Syntax for the batch ADMIN utility UPDATE command

```
UPDATE REPOSITORY(repository-name) 
   REPDSN1RID(ds1_rid_dsname | NULL) 
   REPDSN1RMD(ds1_rmd_dsname | NULL) 
   REPDSN2RID(ds2_rid_dsname | NULL) 
   REPDSN2RMD(ds2_rmd_dsname | NULL) 
   REPDSN3RID(ds3_rid_dsname | NULL) 
   REPDSN3RMD(ds3_rmd_dsname | NULL) 
   AUTOOPEN (YES | NO) 
   SECURITYCLASS(securityclassname | NULL)
```

The only required parameter for this command is the REPOSITORY parameter. The parameters associated with this command have the same meaning as they do when issued with the batch ADMIN ADD command, as described in “ADD command” on page 241.
A user repository must be stopped before it can be updated. We used the JCL shown in Example 7-54 to update the primary index and member data set names in the RS catalog repository.

Example 7-54 JCL to update an IMSRSC repository in the RS catalog repository

```plaintext
/*
//REPOUPD EXEC PGM=FRPBATCH,PARM='XCFGROUP=IM12XREP'
//STEPLIB DD DISP=SHR,DSN=IMS12Q.SDFSRESL
//SYSPRINT DD SYSOUT=* 
//SYSEXIN DD * 
UPDATE REPOSITORY(IMS12XRP_NEW) - 
REPODSN1RID(IMS12Q.IM12X.REPO.IMSPRI.RID.NEW) - 
REPODSN1RMD(IMS12Q.IM12X.REPO.IMSPRI.RMD.NEW) - 
AUTOOPEN(YES) 
/*
```

**RENAME command**

You can use the batch ADMIN RENAME command to rename a user repository name defined within the RS catalog repository. Before a user repository can be renamed, you must first stop the repository.

Example 7-55 shows the RENAME command syntax.

Example 7-55 Syntax for the batch ADMIN utility RENAME command

```
RENAME REPOSITORY(repository-name) REPOSITORYNEW(repository-newname)
```

If you rename the IMSRSC repository at the RS, and you have one or more RM systems enabled with the repository, modify the RM to refer to the repository by the new name.

To modify the RM to refer to the repository by the new name, complete the following steps:

1. Disable RM from using the repository:
   ```plaintext
   UPDATE RM TYPE(REPO) REPOTYPE(IMSRSC) SET(REPO(N))
   ```
2. Ensure that all RMs are disabled from using the repository:
   ```plaintext
   QUERY RM TYPE(REPO) SHOW(ALL)
   ```
3. Modify the CSLRIxxx member of the IMS PROCLIB data set at all RMs to have the new repository name.
4. Enable RM to use the repository:
   ```plaintext
   UPDATE RM TYPE(REPO) REPOTYPE(IMSRSC) SET(REPO(Y))
   ```

To test this command in our environment, we used the JCL shown in Example 7-56.

Example 7-56 JCL to stop and rename a user repository in the RS catalog repository

```plaintext
/*
//REPOSRRN EXEC PGM=FRPBATCH,PARM='XCFGROUP=IM12XREP'
//STEPLIB DD DISP=SHR,DSN=IMS12Q.SDFSRESL
//SYSPRINT DD SYSOUT=* 
//SYSEXIN DD * 
STOP REPOSITORY(IMS12XRP)
RENAME REPOSITORY(IMS12XRP) REPOSITORYNEW(IMS12XRP_NEW)
/*
```
**DELETE command**

Use the batch **ADMIN DELETE** command to remove a user repository from the RS catalog repository. After you delete the user repository, you must delete its associated physical data sets in a separate step using the IDCAMS utility or similar method. Example 7-57 shows the **DELETE** command syntax.

*Example 7-57  Syntax for the batch ADMIN utility DELETE command*

```
DELETE REPOSITORY(repository-name)
```

To test this command in our environment, we used the JCL shown in Example 7-58.

*Example 7-58  JCL to delete a user repository from the RS catalog repository*

```
//*
//REPODEL  EXEC PGM=FRPBATCH,PARM='XCFGROUP=IM12XREP'
//STEPLIB  DD DISP=SHR,DSN=IMS12Q.SDFSRESL
//SYSPRINT DD SYSOUT=*  
//SYSIN    DD *
  DELETE REPOSITORY(IMS12XRP_NEW)
```

**DSCHANGE command**

At certain times it is appropriate to change the disposition of a repository data set (RDS) to either SPARE or DISCARD, which can be done using the batch **ADMIN DSCHANGE** command.

If an error occurs on the primary or secondary RDS, recovery is driven by the RS automatically if a spare RDS is present. When this occurs, the user must then allocate and define a new RDS to replace the one that had the failure. Designate this new RDS as the spare RDS, which you can do by using the batch **ADMIN DSCHANGE** command with the **ACTION(SPARSE)** parameter specified (see Example 7-59 for the **DSCHANGE** command syntax). This command changes the disposition of a RDS pair to SPARE.

*Example 7-59  Syntax for the batch ADMIN DSCHANGE command*

```
DSCHANGE REPOSITORY(repository-name) RDS(1 | 2 | 3) ACTION(SPARE | DISCARD)
```

If you want to replace an existing RDS with a different RDS, you must first stop the repository and change the disposition or status of the RDS to DISCARD. This can be done by issuing the batch **ADMIN DSCHANGE** command with the **ACTION(DISCARD)** parameter specified. When an RDS has a disposition of DISCARD, it can be replaced with a newly defined data set.

Consider the following example scenario that begins with three RDSs that each have different dispositions or statuses of COPY1 (primary RDS), COPY2 (secondary RDS), and SPARE (spare RDS). A write error then occurs on the primary RDS and the following events ensue:

- The RS changes the disposition of the primary data set from COPY1 to DISCARD.
- The RS copies the definitions of the secondary data set to the spare data set.
- The RS changes the disposition of the spare data set to the disposition of the previous primary by changing its disposition from SPARE to COPY1.
- The user issues a command to determine which RDS has been discarded (using either the batch **ADMIN LIST** command or the z/OS modify interface **ADMIN,DISPLAY** command).
- The user deletes the discarded data set that was previously the primary data set.
- The user defines a new data set to replace the previous spare, making the new spare larger than the previous spare as this is a best practice.
The user changes the disposition of this new data set to SPARE using either of the following commands:

- Batch **ADMIN** command:
  
  DSCHANGE REPOSITORY(REPO1) RDS((1) ACTION(SPARE))

- z/OS modify interface command:
  
  F REPOSVR1,ADMIN DSCHANGE (REPO1,S,1)

**Value of 1:** In each of these example commands, 1 is specified. In our example scenario, the primary data set failed. Therefore, 1, representing the old primary data set, is specified to set its disposition to SPARE.

**LIST command**

Use the batch **ADMIN LIST** command to display all user repositories and their associated statuses, or the details of a single repository. The statuses that will be shown are listed here:

- User repository name
- User repository status
- Date of last update
- USERID that last updated user repository
- Data set disposition for each RDS

Example 7-60 shows the syntax for the **LIST** command.

**Example 7-60  Syntax for the batch ADMIN utility LIST command**

```
LIST REPOSITORY(repository-name) | STATUS
```

In our test environment, we issued the **LIST STATUS** command using the JCL shown in Example 7-61 to display all user repositories along with information about them.

**Example 7-61  JCL to show user repository details defined to the RS**

```
/*/  //REPOSTA EXEC PGM=FRPBATCH,PARM='XCFGROUP=IM12XREP'
//STEPLIB DD DISP=SHR,DSN=IMS12Q.SDFSRESL
//SYSPRINT DD SYSOUT=*  //SYSIN DD *
  LIST STATUS
/*/  
```

In our case, we only had a single user repository defined. Therefore, it is the only one shown in the output generated from the batch **ADMIN** utility **LIST STATUS** command; see Example 7-62. If additional user repositories were defined in our environment, they might have been included in the output for this command.

**Example 7-62  Output from the batch ADMIN utility LIST STATUS command**

```
<table>
<thead>
<tr>
<th>Repository</th>
<th>Status</th>
<th>Changed</th>
<th>ID</th>
<th>RDS1</th>
<th>RDS2</th>
<th>RDS3</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMS12XRP</td>
<td>OPEN</td>
<td>2011/07/26</td>
<td>IMSR3</td>
<td>COPY1</td>
<td>COPY2</td>
<td>SPARE</td>
</tr>
</tbody>
</table>
```

FRP4750I - LIST command processing completed successfully
To only display details associated with a single repository, include the repository name on the `LIST` command instead of `STATUS`. Example 7-63 shows sample output that might be received from a batch `ADMIN LIST` command when a user repository name is specified. Notice that you can see detailed information about this user repository, such as its status, its auto-open value, and the different RDSs that it contains and their statuses (dispositions).

**Example 7-63  Output from the batch ADMIN utility LIST command when a user repository is specified**

```
LIST REPOSITORY(IM12XRP)
Repository Name . : IM12XRP
Last updated date/time  : 2011/07/27 00:52:46  USRT001
Status . . . . . . . . : STOPPED
Auto-open . . . . . . . : YES
Security Class . . . . : NOT DEFINED

Repository Data Set pair 1
Index (RID) . . : IMS12Q.IM12X.REPO.IMSPRI.RID
Member (RMD)  . : IMS12Q.IM12X.REPO.IMSPRI.RMD
Status . . . . : COPY1

Repository Data Set pair 2
Index (RID) . . : IMS12Q.IM12X.REPO.IMSSEC.RID
Member (RMD)  . : IMS12Q.IM12X.REPO.IMSSEC.RMD
Status . . . . : COPY2

Repository Data Set pair 3
Index (RID) . . : IMS12Q.IM12X.REPO.IMSSPR.RID
Member (RMD)  . : IMS12Q.IM12X.REPO.IMSSPR.RMD
Status . . . . : SPARE

FRP4750I - LIST command processing completed successfully
```

**START command**

Use the batch `ADMIN START` command to start a specific user repository, for example after it has been defined to the RS catalog repository with the batch `ADMIN ADD` command. Example 7-64 shows the command syntax.

**Example 7-64  Syntax for the batch ADMIN utility START command**

```
START REPOSITORY(repository-name) OPEN(YES | NO)
MAXWAIT(seconds,IGNORE | CONTINUE | ABORT)
```

With the optional `OPEN()` parameter, you can override the `AUTOOPEN=` parameter value that was originally specified when the repository was added to or last updated in the RS catalog (with batch `ADMIN ADD` or `UPDATE` commands, respectively). This parameter value indicates whether the RDSs of the user repository will be open when it starts (with `OPEN(YES)`), or when a user first connects to it (with `OPEN(NO)`). You can only override the `AUTOOPEN=` parameter if it was originally specified as `AUTOOPEN=NO`.

You can optionally include the `MAXWAIT` parameter to indicate how many seconds should elapse before a particular action that you also specify is taken. You can specify a wait time of up to 9999 seconds and opt to have the command continue processing with a return code of either 0 or 4, or opt to have it terminate with a return code of 8. By default, if 5 seconds has elapsed after this command has been issued, the command will continue processing and will give a return code of 4.
The syntax of the `MAXWAIT()` parameter has the following values:

- Seconds will fall between a range of 0-9999.
- IGNORE will continue command processing and set rc=0.
- CONTINUE will continue command processing and set rc=4.
- ABORT will terminate command processing and set rc=8.

In our test environment, we started our user repository at the time that we defined it to the RS with the batch `ADMIN` utility `ADD` command, which is shown in Example 7-52 on page 242.

**STOP command**

Use the batch `ADMIN STOP` command to stop a specific user repository that is defined to the RS catalog. A stopped repository will reject user connection attempts. The command also results in the repository being closed and deallocated by the RS. Example 7-65 shows the command syntax. This command has the same `MAXWAIT()` parameter value as the batch `ADMIN START` command.

**Example 7-65  Syntax for the batch ADMIN utility STOP command**

```
STOP REPOSITORY(repository-name)
MAXWAIT(seconds,IGNORE | CONTINUE | ABORT)
```

Much like a `/DBR` command that prevents programs and transactions from accessing a database, the batch `ADMIN STOP` command, when issued with `MAXWAIT(xx,IGNORE)` or `MAXWAIT(xx,CONTINUE)`, can continue processing after xx seconds have elapsed. At this point, the command continues processing (just like the `/DBR` command) and a specific return code is received, determined by whether IGNORE (rc=0) or CONTINUE (rc=4) was specified. Of course, if `ABORT` was specified instead of IGNORE or CONTINUE, the command terminates processing and a rc=8 is received when xx seconds has elapsed.

In our test environment, we issued the batch `ADMIN` utility `STOP` command at the time that we tested the `RENAME` command, because a user repository must be stopped before it can be renamed. A sample of the JCL that we used is shown in Example 7-56 on page 243.

**Summary of batch ADMIN utility commands**

To summarize all of the batch `ADMIN` commands, see Example 7-66. This job issues each command that has been explained in this section.

**Example 7-66  Sequence showcasing each batch ADMIN command**

```
//FRPBAT   EXEC PGM=FRPBATCH,PARM='XCFGROUP=IM12XREP'
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
//*
ADD REPOSITORY(IMS12XRP) -
  REPDSN1RID(IMS12Q.IM12X.REPO.IMSPRI.RID) -
  REPDSN1RMD(IMS12Q.IM12X.REPO.IMSPRI.RMD) -
  REPDSN2RID(IMS12Q.IM12X.REPO.IMSSEC.RID) -
  REPDSN2RMD(IMS12Q.IM12X.REPO.IMSSEC.RMD) -
  REPDSN3RID(IMS12Q.IM12X.REPO.IMSSPR.RID) -
  REPDSN3RMD(IMS12Q.IM12X.REPO.IMSSPR.RMD) -
  AUTOOPEN(YES)
//*
START REPOSITORY(IMS12XRP) MAXWAIT(30,CONTINUE)
//*
LIST REPOSITORY(IMS12XRP)
//*
```
STOP REPOSITORY(IMS12XR) MAXWAIT(30,CONTINUE)
/*
RENAME REPOSITORY(IMS12XR) REPOSITORYNEW(IMS12XR_PROD)
/*
UPDATE REPOSITORY (IMS12XR_PROD) -
REPSN1RID(IMS12Q.IM12XPROD.REPO.IMSPRI.RID) -
REPSN1RMD(IMS12Q.IM12XPROD.REPO.IMSPRI.RMD) -
AUTOOPEN(NO)
/*
DELETE REPOSITORY(IMS12XR_PROD)

The following section highlights the RS z/OS modify interface commands.

**Repository Server commands issued through z/OS modify interface**

RS parameters can be displayed and dynamically updated by using z/OS modify interface commands. Therefore, by using these commands, you can administer the repository from the z/OS console. Although there is some overlap with the batch **ADMIN** utility, these commands are issued from the perspective of the RS rather than that of the individual user repository.

Table 7-3 summarizes the different z/OS modify interface commands that are available to administer the user repositories, again from the RS perspective.

<table>
<thead>
<tr>
<th>Command</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADMIN</td>
<td>Administrative functions such as change data set disposition, display data sets, start and stop repositories</td>
</tr>
<tr>
<td>AUDIT</td>
<td>Dynamically change audit level</td>
</tr>
<tr>
<td>SECURITY</td>
<td>Refresh RACF profile definitions</td>
</tr>
<tr>
<td>SHUTDOWN</td>
<td>Shut down one or more RS address spaces</td>
</tr>
<tr>
<td>STOP</td>
<td>Stop or shut down a specific RS</td>
</tr>
</tbody>
</table>

The following sections explore the details of each command.

**ADMIN command**

Use the z/OS modify interface **ADMIN** command to perform various administrative tasks. The syntax is shown in Example 7-67.

**Example 7-67 Syntax for the z/OS modify interface ADMIN command**

```
F reposervername,ADMIN
DSCHANGE(repositoryname, S|D, 1|2|3)
DISPLAY(repositoryname | <blank>)
START(repositoryname)
STOP(repositoryname)
```

Use the **ADMIN** command with the **DSCHANGE** parameter to change the disposition of a data set contained in a repository to DISCARD or SPARE status. For details about the circumstances under which this situation is appropriate, see “DSCHANGE command” on page 244.
When changing the disposition of a repository data set pair, specify the name of the repository that contains the target data set pair and the new disposition with either of the following values:

S  A new disposition of SPARE
D  A new disposition of DISCARD

The target repository data set pair is indicated by any of the following values:

1  Primary (also known as COPY1)
2  Secondary (also known as COPY2)
3  Spare

Next, you can use the **ADMIN** command with the **DISPLAY** parameter to display a list of user repository names defined to the RS catalog. If a repository name is specified for the **DISPLAY()** parameter, other details such as the RDS names and statuses will be shown. This command is similar to the batch **ADMIN LIST STATUS** command. Sample output for the z/OS modify interface **ADMIN,DISPLAY** command when a user repository name is specified this command is shown in Example 7-68.

**Example 7-68  Output from the z/OS modify interface ADMIN,DISPLAY command**

```plaintext
/* Display the IMSRSC_REPOSITORY via the ADMIN cmd */
/* FRP2100I - ADMIN DISPLAY repository IMSRSC_REPOSITORY */
/* - Last updated date/time : USRT001 */
/* - Status ............ : OPEN */
/* - Auto-open ........... : YES */
/* - Security Class ....... : NOT DEFINED */
/* FRP2101I - ADMIN DISPLAY repository RDS1: */
/* - Index (RID) ....... : IMSTESTS.FRP1.IMSPRI.RID */
/* - Member (RMD) ....... : IMSTESTS.FRP1.IMSPRI.RMD */
/* - Status ............ : COPY1 */
/* FRP2101I - ADMIN DISPLAY repository RDS2: */
/* - Index (RID) ....... : IMSTESTS.FRP1.IMSSEC.RID */
/* - Member (RMD) ....... : IMSTESTS.FRP1.IMSSEC.RMD */
/* - Status ............ : COPY2 */
/* FRP2101I - ADMIN DISPLAY repository RDS3: */
/* - Index (RID) ...... */
/* - Member (RMD) .... */
/* - Status ........... : NONE */
```

Notice that detailed information about this user repository is displayed, such as its status, its auto-open value, and the different RDSs that it contains and their statuses (dispositions).

Lastly, you can use the **ADMIN** command with the **START** or **STOP** parameters to start and stop a repository, respectively.

**AUDIT command**

Use the **AUDIT** command to dynamically change the audit level setting specified on the **AUDIT_LEVEL** parameter in the FRPCFG configuration member, following the syntax shown in Example 7-69.

**Example 7-69  Syntax for the z/OS modify interface AUDIT command**

```plaintext
F reposervername,AUDIT LEVEL(NONE|HIGH) | RESTART
```
To activate the writing of log records, specify **LEVEL(HIGH)**. To deactivate the writing of logs, specify **LEVEL(NONE)**.

When the RS is starting and attempting to connect to the log stream, an error can occur. Depending on what value you specified for **AUDIT_FAIL** in the FRPCFG member, the RS can either continue starting or can terminate. If you specify **AUDIT_FAIL=CONTINUE**, logging is suspended in the event that the RS encounters an error while attempting to connect to the log stream. You can later activate the logging of records by issuing the **z/OS modify interface AUDIT** command with **RESTART** included.

**SECURITY command**

For repository security, if you need to make changes to your RACF (or System Authorization Facility (SAF) equivalent) definitions, they are only active if you refresh the RACF in-storage profiles. Use the **z/OS modify interface SECURITY** command to accomplish this. The command results in refreshing the RACF profiles in storage to reflect the updated profile definitions. Example 7-70 shows the command syntax.

**Example 7-70  Syntax for the **z/OS modify interface SECURITY** command**

```
F reposervername,SECURITY REFRESH
```

When the **z/OS modify interface SECURITY** command is issued, the RACF in-storage profiles are refreshed, but neither the FRPCFG member or the repository definitions are reread.

**SHUTDOWN command**

To shut down one or more RSs, issue the **z/OS modify interface SHUTDOWN** command. Including the optional **ALL** keyword targets all RSs in the same XCF group. Omitting the **ALL** keyword simply targets the specified RS for shutdown. If the master RS is shut down, one of the subordinate servers becomes the new master. Example 7-71 shows the command syntax.

**Example 7-71  Syntax for the **z/OS modify interface SHUTDOWN** command**

```
F reposervername,SHUTDOWN ALL
```

**STOP command**

The **z/OS modify interface STOP** command is also available to shut down a single RS. Here again, if the master RS is shut down, one of the subordinate servers will become the new master. The syntax is **P reposervername**.

**Repository commands comparison**

For clarity, Table 7-4 displays all batch **ADMIN** commands and RS commands and, if applicable, their equivalents.

<table>
<thead>
<tr>
<th>Batch <strong>ADMIN</strong> utility commands</th>
<th>Repository Server <strong>z/OS modify interface commands</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ADD</strong></td>
<td></td>
</tr>
<tr>
<td><strong>LIST</strong></td>
<td><strong>ADMIN DISPLAY</strong></td>
</tr>
<tr>
<td><strong>START</strong></td>
<td><strong>ADMIN START</strong></td>
</tr>
<tr>
<td><strong>STOP</strong> (repository, not RS)</td>
<td><strong>ADMIN STOP</strong></td>
</tr>
<tr>
<td><strong>RENAME</strong></td>
<td></td>
</tr>
</tbody>
</table>
The following section compares RDDS DRD with repository DRD and examines their similarities and differences.

### 7.9.2 Comparison of DRD use with RDDS versus repository

When using DRD with the new repository instead of the RDDS, you must account for certain considerations especially in the areas of deleting, updating, importing, or exporting resources. This section explores each of these areas.

#### Deleting resources

To delete a resource with RDDS DRD, the runtime definition is deleted with a `DELETE` command. This deletion is hardened to the system RDDS of IMS at system checkpoint by using automatic export (if it is enabled) or with an `EXPORT` command including the `OPTION(overwrite)` parameter. The resource is then removed from the stored resource definitions of the system RDDS.

With repository DRD, the runtime resource definition is also deleted with the `DELETE` command, just as with RDDS DRD. However, because automatic export is not supported with repository DRD and the `EXPORT` command cannot be used to harden deletions to the repository, a different step needs to be taken to accomplish this. The `DELETE DEFN` command must be used to remove stored resource definitions from the IMS resource list associated with the runtime resource deletions that occurred in the running system.

First delete the runtime resource definitions at the IMS system using the `DELETE` command, and then use the `DELETE DEFN` command to delete the resource definitions from the IMSRSC repository.

#### Importing resources

When you issue an `IMPORT` command in an RDDS DRD environment by using the `IMPORT DEFN SOURCE(RDDS)` command, each IMS system that receives the command reads the stored resource definitions from the specified RDDS into the control region, where they become runtime resource definitions.

With repository DRD, each IMS that receives the `IMPORT DEFN SOURCE(REPO)` command reads the stored resources definitions from the repository into the control region, where they become runtime resource definitions.

Both RDDS DRD and repository DRD can create new and update existing runtime resource definitions using an `IMPORT` command.
The ability to refresh runtime resource definitions in an IMS system with stored resource definitions contained in an RDDS or repository is new in IMS 12. This action is taken when the IMPORT command is issued with the OPTION(UPDATE) parameter included. This applies to all of the resources and descriptors being imported. They do not exist as a runtime definition. Instead, the stored resource definition will be created in the running system.

For clarity, Table 7-5 summarizes all possible results for an IMPORT DEFN command.

<table>
<thead>
<tr>
<th>Existing runtime definition exists?</th>
<th>OPTION(UPDATE) specified?</th>
<th>IMPORT DEFN result</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>No</td>
<td>Runtime definition created</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>IMPORT DEFN fails</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Runtime definition created</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Runtime definition updated</td>
</tr>
</tbody>
</table>

If the imported definition is for a resource or descriptor that is unknown to IMS, IMS creates the runtime definition for the resource, regardless of whether the OPTION(UPDATE) parameter is specified.

If the imported definition is for a resource or descriptor for which IMS already has a runtime definition (the resource or descriptor already exists in IMS) and the OPTION(UPDATE) parameter is not specified, the definition is not imported and the command fails.

If the imported definition is for a resource or descriptor for which IMS already has a runtime definition and the OPTION(UPDATE) parameter is specified, the existing runtime definition is updated with the attributes from the imported definition.

**Use case scenario for the IMPORT command**

This section shows how the IMPORT command can be useful when issued with the UPDATE option specified. This scenario illustrates the flexibility of the command in that it can create or update runtime resource definitions depending on whether or not a runtime definition exists for a resource being imported. In the example, a repository is being used instead of an RDDS.

The following scenario illustrates the use of the IMPORT DEFN command when the OPTION(UPDATE) parameter is included.

An IMS application program exists on a test IMS (IMST) and on a development IMS (IMSD) that are in the same IMSplex. Changes are made to this application program, requiring new or changed resource definitions on both IMS systems. Testing is required on the test IMS system before definitions are ported to the development IMS.

- On the test IMS system:
  a. Dynamically add new resources with the DRD CREATE command.
  b. Dynamically update existing resources with the DRD UPDATE command.
  c. Export these changes to a repository:

    ```
    EXPORT DEFN TARGET(REPO) NAME(rsc-names) SET(IMSID(IMST,IMSD))
    ```

After successful testing, the definitions can be ported to the development IMS. Issue the following command to the development IMS:

```
IMPORT DEFN SOURCE(REPO) NAME(rsc-names) OPTION(UPDATE)
```
New runtime resource definitions are added to the development IMS. In addition, existing runtime resource definitions are updated in the development IMS.

Notice that the **EXPORT** command specifies the resource definitions that are written to the repository, targeting the test (IMST) and development (IMSD) IMS systems. When the export occurs, the following two actions occur:

- New resources names and types are added to the IMS resource lists of IMST and IMSD within the repository.
- New stored resource definitions are either added to or updated within the repository.

The **EXPORT** command hardens the runtime definitional additions and changes made on IMST to its stored definitions contained in the repository, and updates its IMS resource list. It does the same for IMSD; that is, the stored resource definitions and IMS resource list of IMSD are updated in preparation for a later import after successful testing has been completed on IMST. Another use for the **EXPORT** command is to ensure that a user repository is not empty.

**IMPORT command considerations**

When using the **IMPORT** command with the **UPDATE** option, a runtime definition is created if one does not already exist and an existing runtime definition is updated with the new attributes.

In most cases, when updating an existing runtime resource definition, the resource cannot be in use or the **IMPORT** fails. However, the following transaction attributes can be updated even if the transaction is in use: CLASS, LCT, LPRI, NPRI, MAXRGN, PARLIM, PLCT, PLCTTIME, SEGNO, SEGSZ, and TRANSTAT. The TRANSTAT program attribute can be updated while the program is in use.

If a database definition is to be updated by the **IMPORT** command, access to the database must be stopped before the import is done. This applies to changing the resident value and to changing the access type. When updating the resident value with the **UPDATE DB** command, you must stop access to the database before you issue the command. However, you do not have to stop access to the database when using the **UPDATE DB** command to update the access type. With the **IMPORT** command, you must always stop access to the database before a database definition is to be updated, even if the import is simply updating the access type.

When the **IMPORT** command is issued with the **UPDATE** option, all existing resources affected by the update are quiesced. For example, if the import is updating a transaction definition, the transaction and the associated program are quiesced. While quiesced, a resource cannot be updated or deleted, and in most cases work cannot be run against the resource, which means the resource cannot be scheduled. Certain latches are held during the import process that prevent work from being done. No resources can be scheduled while the resources to be updated are being quiesced. A system checkpoint is not allowed while an import is in progress.

**Exporting resources**

Exporting resource definitions to an RDDS is handled differently than exporting to a repository.

**Exporting to RDDS**

When exporting in an RDDS DRD environment, resources that have been newly created, changed, or deleted can all be hardened to the RDDS by using automatic export if it is enabled. They can also be hardened to the RDDS by issuing an **EXPORT** command, which can either overwrite the entire contents of an RDDS or append to it.

With RDDS DRD, each IMS system has its own dedicated pair of system RDDSs that contain the entire collection of MODBLKS definitions for the system. When an **EXPORT** command is issued in a cloned environment, it should be routed to all of the IMS systems in the IMSplex so that their system RDDSs remain synchronized with the same set of definitions. In this case,
each IMS system that receives the command will process it and perform export. There is 
potential here for one IMS system to fail the EXPORT command, where others can succeed. This 
requires the user to manually resynchronize the system RDDSs by correcting the error and 
reissuing the EXPORT on the IMS that failed. Manual coordination is not always straightforward 
and is not recommended because it requires more effort on the part of the user.

Finally, RDDS DRD exporting only applies to active IMS systems, and there is no way for the 
stored resource definitions of an inactive IMS to be updated. In IMS 12, several of these 
limitations are alleviated when DRD is used with the repository instead of the RDDS.

Exporting to the repository

With repository DRD exporting, only added and changed resources or descriptors can be 
written to the repository. As previously mentioned, automatic export is not supported with this 
type of DRD. To export to a specific IMS resource list within the repository, an EXPORT command 
is required. To export the runtime resources definitions of the IMS that have been added or 
changed since the previous EXPORT command was issued, include the OPTION(CHANGESONLY) 
parameter omitting the SET(IMSID()) parameter so only the targeted IMS's resource definitions 
in the repository are updated with the changes. If any runtime resource definitions were deleted 
from the IMS, issue a DELETE DEFN command with the correct FOR(IMSID()) parameter to 
remove the associated stored resource definitions from the repository.

With repository DRD, instead of each IMS having its own set of system RDDSs, each IMS has 
its own IMS resource list that contains the resource names and resource types defined for the 
IMS. The repository also has resource definitions for each resource defined to the repository.

When an EXPORT command is issued in a repository DRD environment, the resource 
definitions are written to the repository. The SET(IMSID()) parameter determines which IMS 
stored resource definitions and IMS resource lists to update in the repository. Unlike RDDS 
DRD, where each IMS that receives the EXPORT command processes it, a repository DRD 
EXPORT is only processed by one command master IMS. Because the command is processed 
as a single unit of work, all of the specified resource definitions and IMS resource lists are 
updated by the export, or none of them are updated if an error occurs.

The possibility is eliminated that the stored resource definitions of some IMS instances might 
be different from others because only one IMS is processing the command as a single unit of 
work, writing to the shared repository. RDDS DRD export also processes the command as a 
single unit of work, but the difference is that multiple IMSs are each processing the command 
separately, updating different RDDSs, which as previously stated can succeed or fail at the 
different systems resulting in desynchronization.

Repository DRD export can update the stored definitions of an IMS that is inactive, unlike 
RDDS DRD export. These updated stored definitions can be applied when the IMS restarts, but 
remember, export does not handle resource deletions. It only handles additions or changes. To 
deleted stored resource definitions from the repository, a DELETE DEFN command is required.

Parameters specific to DRD type

Certain EXPORT command parameters only apply to a specific type of DRD. For example, the 
following parameters only apply to repository DRD:

- STARTTIME()
- ENDTIME()
- OPTION(CHANGESONLY)

The following parameters only apply to RDDS DRD:

- OPTION(APPEND)
- OPTION(OVERWRITE)
7.9.3 Using DRD with the IMS repository in an online environment

This section explores a few repository DRD usage scenarios that occur in an online environment, including a closer look at the following areas:

- Handling indoubt work in progress when an import or export is interrupted
- Maintaining unique attribute values for specific IMS resources within the repository
- Exporting resources that are dynamically created with the DFSINSX0 user exit

**Indoubt work in progress**

It is possible that IMS, RM, or the RS can unexpectedly terminate during **IMPORT** or **EXPORT** command processing. In this case, it is uncertain whether the unit of work (UOW) command processing completed, making it “indoubt work.”

The point of failure is important in determining whether the command needs to be reissued. Use the `QUERY SHOW(DEFN)` command to determine when the resources or descriptors involved in the import or export were last created, updated, or imported and compare these timestamps to the point of failure. Look for the resource data under the following command output column headers:

- `TimeCreate`
- `TimeUpdate`
- `TimeImport`

This should indicate whether the **IMPORT** or **EXPORT** command should be issued again.

**Use case scenarios: Indoubt**

The following event sequence illustrates a scenario in which **EXPORT** command processing becomes indoubt. Then we explain how to resolve this situation.

1. You enter the following command:
   
   ```
   EXPORT DEFN TARGET(REPO) TYPE(TRAN) NAME(TRANA,TRANB)
   ```

2. IMS terminates during command processing.

3. Work in progress of the **EXPORT** command is indoubt.

4. You enter the `QUERY TRAN` command:
   
   ```
   QUERY TRAN NAME(TRANA,TRANB) SHOW(DEFN,TIMESTAMP)
   ```

5. You check the `TimeCreate` or `TimeUpdate` column in command response data.

Next, we show another event sequence that illustrates a scenario in which **IMPORT** command processing becomes indoubt. Then we explain how to resolve this situation.

1. You enter the following command:
   
   ```
   IMPORT DEFN SOURCE (REPO) TYPE(TRAN) NAME(TRANA,TRANB)
   ```

2. IMS terminates during command processing.

3. Work in progress of the **IMPORT** command is indoubt.

4. You enter the following command:
   
   ```
   QUERY TRAN NAME(TRANA,TRANB) SHOW(DEFN,TIMESTAMP)
   ```

5. You check the `TimeImport` column in the command response data.

In both of these scenarios, if a `QUERY` command response indicates that work in progress was not committed, reissue the **IMPORT/EXPORT** command.
Resources with unique attribute values in repository

In an MSC environment, each IMS system must have unique SIDR and SIDL values for remote transactions and transaction descriptors. To modify these SIDR and SIDL values, the EXPORT command with the appropriate IMSID must be specified on the SET(IMSID) parameter. If the SET(IMSID()) parameter is omitted, the command must be routed to the specific IMS whose SIDR and SIDL definitional values are to be exported. In either case, the resource definitions of the IMS system are updated with the new values.

**Important:** When EXPORT is issued with the SET(IMSID(*)) parameter, every attribute value except for SIDR/SIDL is written to the stored resource definitions of each IMS system, because these values must always be unique to a particular IMS system.

For local transactions and transaction descriptors, the SIDR and SIDL values are saved as 0 in the repository for each IMS. When the stored resource definition is imported from the repository either during AUTOIMPORT processing or during processing of the IMPORT command, the SIDR and SIDL values are set to the lowest local SID value of the IMS system where the runtime resource definition is created.

Exporting resources created with the DFSINSX0 user exit to repository

The Destination Creation (DFSINSX0) user exit can be used to dynamically create transaction resources that will process messages that come into IMS with an unknown destination. To export these dynamically created transactions to the repository, set TRNQ_FC_EXPORT=1 on the exit input parameter list and issue the EXPORT DEFN TARGET(REPO) command. When exporting, include either the NAME() parameter specifying names of dynamically created transactions. If the resource names are unknown, use the STARTTIME() and ENDTIME() parameters to encompass the timeframe in which the exit dynamically created the transaction resources.

Keep in mind that when using DRD with the repository, no automatic export will occur at system checkpoint, unlike as is the case with RDDS DRD. To harden transactions that are dynamically created with the DFSINSX0 user exit to the repository, the only option is to export with an EXPORT command.

**7.9.4 Managing the IMS repository in an offline batch environment**

This section provides scenarios in which the repository can be managed in an offline environment, namely for security updates and recovery procedures.

**Updating security settings**

The use case scenarios in this section illustrate the use of both batch ADMIN and z/OS modify interface commands being used to specify security settings.

If you are going to restrict access to a user repository defined in the RS catalog, you typically designate the SAF class name to be used in the FRPCFG configuration member during initial setup. However, if no security class was specified in the member at that time, you can update it later on using the batch ADMIN UPDATE command.
A sample batch **ADMIN UPDATE** command is shown in Example 7-72. In the example, the security class XFACILIT is designated for use in restricting access to a repository named REPO1.

**Example 7-72 Using the batch ADMIN UPDATE command to designate a SAF security class**

```plaintext
UPDATE REPOSITORY(REPO1) SECURITYCLASS(XFACILIT)
```

After the SAF security class is designated, you can update your RACF definitions to protect your user repository (Example 7-73).

**Example 7-73 Making RACF definitional changes after the security class has been designated**

```plaintext
RDEFINE XFACILIT FRPREP.REPO1 UACC(NONE)
PERMIT FRPREP.REPO1 CLASS(XFACILIT) ID(ANGIE) ACCESS(READ)
```

Here, you can see that the SAF class XFACILIT that we just designated for use in Example 7-72 is specified in the RACF definitions to prevent unauthorized user access to the REPO1 repository. A user ID named ANGIE is then permitted to read it within these RACF definitions.

After the definitional RACF changes are made, the in-storage RACF profiles must be refreshed to reflect these updates. Use the z/OS modify interface **SECURITY** command to accomplish this task (Example 7-74).

**Example 7-74 Refreshing RACF in-storage profiles with z/OS modify interface SECURITY command**

```plaintext
F REPOSVR1,SECURITY REFRESH
```

**Important:** The z/OS modify interface **SECURITY** command must be used any time the RACF definitions are updated.

For more information about implementing repository security, see 7.10, “Security considerations” on page 258.

**Recovery activities**

A repository data set pair (hereafter referred to as RDS) that is identified by the server as having lost integrity is **discarded**. At this time, the disposition of the RDS will be changed to **DISCARD**. If an RDS is discarded due to a write error, then the repository will be stopped at this time to enable recovery. In this event, the RS drives recovery automatically if a spare RDS is available and the only task of the user is to allocate and define a new spare data set, assigning it to SPARE disposition with a **DSCHANGE** command. If no spare RDS is available, the user repository is stopped and administrator intervention is required to restart the user repository.

**Read errors:** Only **read** errors are handled by using the other valid copy (primary or secondary) to access the needed data; there is no spare switching. This is an improvement for reads compared to previous IMS releases, because neither the RDDS nor MODBLKS have a second data set.

**Use-case scenario: Recovery**

Consider a scenario in which a repository named REPO1 contains a primary, a secondary, and a spare data set with dispositions of COPY1, COPY2, and SPARE, respectively. A write
error occurs on the primary data set. The RS will then drive recovery in the following event sequence:

1. The primary data set is automatically changed to the disposition of DISCARD.
2. Definitions of the secondary data set are copied to the spare.
3. Repository Server changes the spare to primary.

The RS changes the disposition of the primary RDS from COPY1 to DISCARD and copies the contents of the secondary RDS to the spare RDS automatically (if a spare is available). At this point, the existing spare becomes the new primary data set that replaces the repository data set that failed.

At this point in the recovery process, the user must take the following actions:

1. Issue either a batch `ADMIN LIST` or z/OS modify interface `ADMIN,DISPLAY` command to determine which RDS has been discarded.
2. Delete and define the discarded primary data sets to replace the old spare.
3. Change the disposition of this new data set to SPARE.

The batch `ADMIN LIST` command or z/OS modify interface command can be issued to show the dispositions, or statuses, of each repository data set. You must then delete the bad repository data set whose disposition is now DISCARD. Allocate and define a new repository data set (ideally, the size should be larger than the previous spare) and assign a disposition of SPARE to this new data set using either the batch `ADMIN` or z/OS modify interface commands shown in Example 7-75 and Example 7-76. Notice that in each command, 1 is specified. In our sample scenario, the primary data set failed and so a 1 representing the old primary data set is specified to set its disposition to SPARE.

Example 7-75 shows the format of the batch `ADMIN DSCHANGE` command.

```
Example 7-75   A batch ADMIN DSCHANGE command setting RDS1 to SPARE disposition
DSCHANGE REPOSITORY(REPO1) RDS((1) ACTION(SPARE))
```

Example 7-76 shows the format of the z/OS modify interface command.

```
Example 7-76   A z/OS modify interface command setting RDS1 to SPARE disposition
F REPOSVR1,ADMIN DSCHANGE (REPO1,S,1)
```

### 7.10 Security considerations

In securing access to RS resources, you must consider the following areas as explained in this section:

- Repository access
- Types of repository security
- Repository security implementation
7.10.1 Repository access

Access to a user repository can be gained either through RM or directly.

Access by using RM

A user repository can be accessed either through RM or directly. If going through RM, the caller will be considered either “authorized” or “non-authorized”. IMS is an authorized caller and therefore so long as RM is authorized, it has access to all repository contents by using commands such as the following examples:

- `EXPORT TARGET(REPO)`
- `IMPORT SOURCE(REPO)`
- `DELETE DEFN`
- `QUERY with SHOW(DEFN)`

**Important:** Another layer of security can be put in place to prevent an unauthorized user from issuing any new repository-specific commands. For information about how to restrict access to these commands, see “Security considerations for commands” on page 240.

However, the RM utilities `CSLURP10` (RDDS to Repository RM utility) and `CSLURP20` (Repository to RDDS RM utility) are non-authorized RM callers. In this case, the utilities do not automatically have authorization for repository access just because RM is authorized, as is the case with IMS. Therefore, the RM utilities require separate authorization to access the repository.

Direct access

Access to a RS can be gained directly (without going through RM) by both the batch `ADMIN` utility or through the z/OS modify interface. As previously mentioned, the batch `ADMIN` utility runs as a JCL job. Therefore, the user ID specified in the JCL can be used for authorization checking to determine whether repository access is allowed. Security for commands entered through the z/OS modify interface can be implemented using standard console security.

7.10.2 Types of repository security

Repository security can be at the connection level or the member level.

Connection security

Connection security is used when a caller attempts to connect to the user repository. This concept applies to both authorized and non-authorized RM callers, which are the IMS and RM utilities (respectively). In either case, the caller must specify a user ID in the JCL, which is checked in RACF to determine whether access to the repository is allowed. For the RM utilities, this user ID is also used for SCI registration.

Member-level security

Security can also be implemented at the member level within a user repository, which only applies to the RM utilities. In this case, you can restrict access to individual members, and separately authorize the user ID associated with the RM utility (again, specified in the JCL) being used to access these members.

Use member-level security if you want to restrict the RM utilities to accessing only certain resources. After you protect the individual resources in a RACF class, permit the user ID specified in the JCL of the utility to access these resources accordingly. For example, the `CSLURP20` utility reads repository resources and copies them to an RDDS. If you want to
prevent the utility from reading certain resources from the repository and then copying them to the RDDS, restrict access to them and only permit the utility to access (read/copy) the desired resources.

7.10.3 Repository security implementation

RS resources, including the following examples, can be restricted from unauthorized access:

- User repository
- Catalog repository
- Members within a user repository
- Audit levels associated with an individual repository

First, choose a class that your RS resources will be protected in. Use either the FACILITY class or your own user-defined class. Use member-level security due to the 39-character profile name length restriction of FACILITY class.

**Tip:** To define a new user-defined class to RACF, add the new class to the RACF Class Descriptor Table (ICHRRCDE) and then update the RACF Router Table (ICHRFR01) with the new class.

The next steps are to protect your resources by defining general resource profiles, and then to grant access to users that have been protected in those resource profiles.

### Protecting Repository Server resources

This section shows how to restrict access to RS resources to authorized users by defining profiles for them. All examples specify a user-defined RACF class named XFACLIT. As previously mentioned, use a user-defined class when defining resource profiles in RACF.

To protect a user repository, define a profile using the format **FRPREP.repositoryname**. For example, if you want to protect a user repository named REPO1, define a resource profile for it using the format shown in Example 7-77.

**Example 7-77  Restricting access to a user repository by defining a profile for it in RACF**

```
RDEFINE XFACLIT FRPREP.REPO1 UACC(NONE)
```

Notice that we have specified that the user repository is protected in the XFACLIT class. Also, by designating a universal access of none with **UACC(NONE)**, explicit permission must be granted in RACF for any user ID to access this user repository.

To protect all user repositories in your environment, define a profile using the format **FRPREP.*** (see Example 7-78).

**Example 7-78  Restricting access to all user repositories by defining a catch-all profile in RACF**

```
RDEFINE XFACLIT FRPREP.* UACC(NONE)
```

To restrict access to the RS catalog repository, use the **FRPREP.CATALOG** format when defining a resource profile for it. Keep in mind that a user can access or update the RS catalog repository by using batch **ADMIN** commands, for example when adding a new user repository to it during initial setup.
Example 7-79 shows a RACF statement that restricts access to the catalog repository in the XFACILIT class, with a universal access of none.

**Example 7-79  Restricting access to the RS catalog repository by defining a profile for it in RACF**

```
RDEFINE XFACILIT FRPREP.CATALOG UACC(NONE)
```

Individual members within a repository can also be restricted from unauthorized access by the RM utilities CSLURP10 (RDDS to Repository RM utility) and CSLURP20 (Repository to RDDS RM utility). To define a resource profile for an individual repository member, use the following format:

```
FRPMEM.repositoryname.DFS.RSC.membername
```

The membername referenced in the required format must consist of the IMSplex name, followed by the resource name and resource type. For example, a transaction named PART that exists in an IMSplex named IMSPLEX1 can be defined in a profile such as the one shown in Example 7-80.

**Example 7-80  Restricting access to an individual member by defining a profile for it in RACF**

```
RDEFINE XFACILIT FRPMEM.REPO1.DFS.RSC.IMSPLEX1.TRAN.PART UACC(NONE)
```

Lastly, you can restrict unauthorized users from modifying audit levels associated with an individual repository. Define a resource profile in RACF using the following format:

```
FRPAUD.repositoryname.DFS.RSC.TYPE
```

Example 7-81 shows a RACF profile definition that restricts the audit level associated with a user repository named REPO1. By designating a universal access of none with UACC(NONE), explicit permission must be granted to a user ID in RACF for that user ID to access modifying the audit level for this user repository.

**Example 7-81  Restricting access to modifying audit levels by defining a profile for it in RACF**

```
RDEFINE XFACILIT FRPAUD.REPO1.DFS.RSC.TYPE UACC(NONE)
```

**Granting user access to Repository Server resources**

After defining resource profiles to protect your RS resources from unauthorized access, the next step involves granting access to the appropriate users. For each resource, use the appropriate format shown in “Protecting Repository Server resources” on page 260.

To grant user authorization to a repository after having restricted access to it (such as in Example 7-77 on page 260), use RACF PERMIT statements such as the ones shown in Example 7-82 on page 261. In the example, we grant read access to a user ID named VIEWER1 and alter access to a user ID named ADMIN1.

**Example 7-82  Granting different levels of repository access to user IDs**

```
PERMIT FRPREP.REPO1 CLASS(XFACILIT) ID(VIEWER1) ACCESS(READ)
PERMIT FRPREP.REPO1 CLASS(XFACILIT) ID(ADMIN1) ACCESS(ALTER)
```

To grant user authorization to a RS repository catalog after having restricted access to it (such as in Example 7-79), use a RACF PERMIT statement such as the one shown in Example 7-83. In the example, we grant alter access to a user ID named ADMIN1 so that it can update the contents of the RS catalog repository.
Example 7-83  Granting alter access to the ADMIN1 user ID

PERMIT FRPREP.CATALOG CLASS(XFACILIT) ID(ADMIN1) ACCESS(ALTER)

To grant access to individual members that have been protected in RACF (such as in Example 7-80), use a PERMIT statement such as the one shown in Example 7-84. In the example, notice that we grant update access to a user ID named USRUTL10 so that it can update a transaction named PART, which exists in the REPO1 user repository within the IMSPLEX1 IMSplex.

Example 7-84  Granting update access to the USRUTL10 user ID for an individual resource

PERMIT FRPMEM.REPO1.DFS.RSC.IMSPLEX1.TRAN.PART CLASS(XFACILIT) ID(USRUTL10) ACCESS(UPDATE)

To grant access to all members that have been protected in RACF, use a catch-all PERMIT statement such as the one shown in Example 7-85. In this example, we grant read access to user ID USRUTL20 for all resources that have been protected in the XFACILIT class.

Example 7-85  Granting read access to the USRUTL20 user ID for all resources

PERMIT FRPMEM.*.*.*.*.*.*.*.* CLASS(XFACILIT) ID(USRUTL20) ACCESS(READ)

Finally, if you have restricted access to the audit level of your user repository (such as in Example 7-81), you can grant access to specific user IDs to allow the user to modify the audit level.

Important: If a user ID needs RACF UPDATE access for individual members that have been restricted from unauthorized access, the user ID also needs RACF UPDATE access for the actual user repository that these members are contained in. Therefore a separate RACF PERMIT statement is required to ensure this access.

To do so, use a RACF PERMIT statement such as the one shown in Example 7-86. In the example, access to modify the audit level of user repository REPO1 is granted to a user ID named USRZOSMI.

Example 7-86  Granting update access to the USRZOSMI user ID to modify the audit level of REPO1

PERMIT FRPAUD.REPO1.DFS.RSC.TYPE CLASS(XFACILIT) ID(USRZOSMI) ACCESS(UPDATE)

Tip: You can group several user IDs together for higher efficiency when defining resource profiles and granting access to them. In this case, PERMITs reference RACF group rather than each individual user ID. The following example illustrates this concept in a series of RACF statements that protect a user repository named REPO1 in the XFACILIT class and subsequently grant access to them:

- RDEFINE XFACILIT FRPREP.REPO1 UACC(NONE)
- ADDGROUP FRPVIEW
- ADDGROUP FRPENDIT
- PERMIT FRPREP.REPO1 CLASS(XFACILIT) ID(FRPVIEW) ACCESS(READ)
- PERMIT FRPREP.REPO1 CLASS(XFACILIT) ID(FRPENDIT) ACCESS(UPDATE)
- CONNECT <VIEWER1> GROUP(FRPVIEW)
- CONNECT <VIEWER2> GROUP(FRPVIEW)
- CONNECT <VIEWER3> GROUP(FRPVIEW)
- CONNECT <UPDATER4> GROUP(FRPENDIT)
- CONNECT <UPDATER5> GROUP(FRPENDIT)
7.11 DRD user interface enhancements

All new and enhanced DRD commands in IMS 12 were incorporated into the DRD user interface (UI). The DRD UI is an ISPF panel-driven interface that assists you in entering various DRD commands. It is in the IMS application called Manage Resources (MR), which can be invoked from the IMS Application Menu.

New panels were added to the DRD UI for the following commands:
- **EXPORT DEFN** (in the initial TARGET panel)
- **IMPORT DEFN** (in the initial SOURCE panel)
- **DELETE DEFN**

In addition, existing panels were enhanced for the following existing commands:
- **IMPORT DEFN** (source RDDS)
- **QUERY DB**
- **QUERY TRAN**
- **QUERY TRAN DESC**
- **QUERY DBDESC**
- **QUERY PGM**
- **QUERY PGMDESC**
- **QUERY RTC**
- **QUERY RTCDESC**

This section shows several example panels for various commands in “list view” unless otherwise noted. This view provides the greatest level of assistance to the user. The alternative view is “syntax view” for more experienced IMS users who are familiar with command format. This section includes only the new panels and panels that were enhanced with additional parameter information. Panels that were not changed, but that now apply to repository DRD (in addition to RDDS DRD), are not shown.

To start the Manage Resources application, follow these steps:
1. Invoke the IMS Application Menu using the following TSO command:
   ```
   ex 'ims12q.sdfsexec(dfsappl)' 'hlq(ims12q)'
   ```
2. In the IMS Application Menu panel (Figure 7-4 on page 264), select option 2 to invoke the Manage Resources application.
The IMS Application Menu

Command ===>

Select an application and press Enter.

1 Single Point of Control (SPOC)
2 Manage resources
3 Reserved for future use
4 HALDB Partition Definition Utility (PDU)
5 Syntax Checker for IMS parameters (SC)
6 Installation Verification Program (IVP)
7 IVP Export Utility (IVPEX)
8 IPCS with IMS Dump Formatter (IPCS)
9 Abend Search and Notification (ASN)

To exit the application, press F3.

F1=Help   F12=Cancel

Figure 7-4  The IMS Application Menu

The IMS Manage Resources panel (Figure 7-5) opens.

The IMS Manage Resources application menu

Several panels for the EXPORT command have been enhanced or added to accommodate the IMS repository function.
In the initial EXPORT panel (Figure 7-6), select the target data set for export. In our test environment, we selected option 2.

![Figure 7-6](image_url)
In the IMS Export Resource panel (Figure 7-7), which is new in IMS 12, you can set all EXPORT command parameter values.

**Tip:** When navigating through the MR application panels, ensure that your cursor is always placed on the field in which data needs to be entered before pressing Enter. If your cursor is elsewhere on the panel, when you press Enter you might receive an error message.

**Figure 7-7** New EXPORT panel in the MR application for setting parameters (list view)

Figure 7-7 shows the panel in list view, which provides the most assistance to the user. The equivalent panel is shown in Figure 7-8 on page 267, but in syntax view. This view assumes the user is already familiar with command format and parameter values.
Several panels for the IMPORT command have been enhanced or added to accommodate the IMS repository function. Figure 7-9 shows the initial IMPORT panel that designates the source data set for import.

In the IMS Import panel (Figure 7-9), select a source. In our test environment, we selected option 2.
In the IMS Import Repository panel (Figure 7-10), which is new in IMS 12, you can set all the IMPORT command parameter values.

In the IMS Import Repository panel (Figure 7-10), which is new in IMS 12, you can set all the IMPORT command parameter values.

Figure 7-10 New IMPORT panel in the MR application for setting parameters (list view)

Figure 7-10 shows the panel in list view, which provides the most assistance to the user.
The equivalent panel is shown in Figure 7-11, but in syntax view. This view assumes the user is already familiar with command format and parameter values.

```
File  Action  Manage resources  SPOC  View  Options  Help

IM12X                       IMS Import Repository

Command ===>

Press Enter to continue.

IMPORT DEFN SOURCE(REPO)
  NAME( )
  TYPE( )
  OPTION( )
  SCOPE( )

F1=Help   F12=Cancel
```

Figure 7-11  New IMPORT panel in the MR application for setting parameters (syntax view)
The panel for the IMPORT command that uses the RDDS as a source was also enhanced in IMS 12. It now includes the new OPTION(UPDATE) parameter to refresh runtime resource definitions with stored resource definitions residing in the RDDS (Figure 7-12).

```
File  Action  Manage resources  SPOC  View  Options  Help
ssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssss
IM12X                        IMS Import Resources

Command ==> sssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssss
Plex . . Route . . Wait ..
Press Enter to continue.

Source . . . . . . : RDDS

Source RDDS data set ..

Resource type
Enter "/" to select types
   ALL     All RSCs & DESCs
   ALLRSC  All resources
   DB      Database resource
   PGM     Program resource
   RTC     Routing code resource
   TRAN    Transaction resource

   ALLDESC All descriptors
   DBDESC  Database descriptor
   PGMDESC Program descriptor
   RTCDESC Routing code descriptor
   TRANDESC Transaction descriptor

OPTION
Enter "/" to select options
   ABORT  Abort import if error
   UPDATE Update runtime defs

ALLRSP  Show all responses

F1=Help   F12=Cancel
```

Figure 7-12  Enhanced IMPORT panel including the new OPTION(UPDATE) parameter
The first MR application panel for the DELETE command was also enhanced to account for the new DELETE DEFN command, and is shown in Figure 7-13.

In the IMS Delete panel, select a source. In our test environment, we selected option 2.

```
File  Action  Manage resources SPOC View Options Help
IM12X                             IMS Delete

Command ==>

Plex . .        Route . .            Wait . .
Select a source. Press Enter to continue.

* DELETE . . . . . . . .  1. RESOURCES
2. DEFN

F1=Help   F12=Cancel
```

* Figure 7-13  Enhanced DELETE panel in the MR application showing the new DEFN keyword
We then arrived at the new panel that was added to the MR application for the **DELETE DEFN** command, which is shown in Figure 7-14.

```
File  Action  Manage resources  SPOC  View  Options  Help

IMIX                          IMS Delete DEFN

Command ===>

Press Enter to continue.

Target . . . . . . : REPO

Resource name . . . .

For IMSID . . . . .

Resource type

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DB</td>
<td>Database resource</td>
<td>DBDESC</td>
<td>Database descriptor</td>
</tr>
<tr>
<td>PGM</td>
<td>Program resource</td>
<td>PGMDESC</td>
<td>Program descriptor</td>
</tr>
<tr>
<td>RTC</td>
<td>Routing code resource</td>
<td>RTCDESC</td>
<td>Routing code descriptor</td>
</tr>
<tr>
<td>TRAN</td>
<td>Transaction resource</td>
<td>TRANDESC</td>
<td>Transaction descriptor</td>
</tr>
</tbody>
</table>

OPTION

Enter "/" to select options

ALLRSP   Show all responses

F1=Help   F12=Cancel

Figure 7-14   New DELETE DEFN panel in the MR application (list view)
The list view panel for the **DELETE DEFN** command is shown in Figure 7-14. The equivalent panel in syntax view is shown in Figure 7-15 on page 273.

![Figure 7-15](image)

```plaintext
DELETE DEFN TARGET(REPO)
  NAME(               )
  IMSID(              )
  TYPE(               )
  OPTION(             )
```

*Figure 7-15  New DELETE DEFN panel in the MR application (syntax view)*
Lastly, the **QUERY** command panels in the MR application were enhanced to account for the new repository-specific parameter values of **SHOW**(DEFN) and **SHOW**(IMSID). In our test environment, we used the MR application to query our databases, and took note of these new parameter values shown in Figure 7-16.

---

![Figure 7-16 Enhanced QUERY panel in the MR application (list view) showing new parameter values](image-url)
The **QUERY** command panel equivalent to what is shown in Figure 7-16 can be seen in Figure 7-17 in syntax view.

![Figure 7-17](image)

### 7.12 IVP enhancements for repository

This section explains how the IMS IVP application was enhanced to support the IMS repository function.

Several new jobs and tasks, including the following examples, were added to the IVP for repository DRD:

- Creation of the RS catalog data sets and the user repository
- Creation of the RS configuration file
- Execution of the RS startup procedure
- JCL to execute the actions:
  - Start an RS
  - Add a user repository to the RS catalog
  - List user repository status information
  - Populate a user repository
  - Rename a user repository in the RS catalog
  - List detailed information for a single user repository
  - Modify and update user repository definitions
  - Delete a user repository in the RS catalog
  - Delete actual RS catalog and user repository data sets
We used the IVP to set up the required elements for our test environment. Figure 7-18 shows a list of repository-specific jobs and tasks within the IVP.

7.13 Value of using the IMS repository with DRD

The value of using the IMS repository with DRD is clearly shown with its full support for populating, managing, storing, sharing, and retrieving a consistent set of DRD stored resource definitions for multiple-IMS IMSplexes and single-IMS IMSplexes all in a centralized location. You will have improved data availability due to the duplexing of stored resource definitions with the ability to have a spare. Migrating to repository DRD is simple because it can be accomplished with commands. An IMS outage is not required to enable or disable the repository function.

When repository DRD is used, operational functionality and flexibility for managing resources across an IMSplex are improved with the following functions:

- Generic resource definition is available along with IMS-specific resource definitions.
- The EXPORT process is a single unit or work for an entire IMSplex (all succeeds or all fails).
- The EXPORT process is controlled by the user (no AUTOEXPORT).
- You can select CHANGESONLY or by time periods.
- Deleting stored resource definitions is controlled by the user.
- You can update by using IMPORT on an existing runtime.
- **EXPORT** is reflected in all IMSs in an IMSplex, whether they are up or down at the time.
- **IMPORT SCOPE(ALL)** allows IMSs that are down to be aware of changed definitions at the next restart.
- You can use **QUERY** to show the stored resource definitions from a repository.
- The **DFSINSX0** user exit supports export to a repository.

The IMS repository is considered a strategic IMS architectural direction. Because all stored resource definitions exist in a single, shared data set, consistency is ensured while eliminating your need to maintain and coordinate multiple sets of RDDSs in a multiple-IMS IMSplex. The architecture of the repository ensures consistency and integrity of data without the need for additional controls.
Installation and migration considerations

This chapter explains the installation and migration tasks to consider if you are migrating from a supported IBM Information Management System (IMS) release to IMS 12.

This chapter includes the following topics:
- Prerequisites and coexistence
- Fix Category HOLDDATA
- Coexistence with IMS 12
- General coexistence considerations
- Installation Verification Program
- Syntax Checker
- Installation and migration tasks
- Review of migration considerations
8.1 Prerequisites and coexistence

This section describes the prerequisites and coexistence for IMS 12. For detailed information, see IMS Version 12 Release Planning, GC19-3019.

8.1.1 Preventive Service Planning

Before you begin installing IMS 12, review the current Preventive Service Planning (PSP) information. The PSP buckets maintain current lists (which have been identified since the package was created) of any recommended or required service for the installation of this package. This service includes software PSP information that contains HIPER and required program temporary fixes (PTFs) against the base release.

If you obtained IMS 12 as part of a Custom-built Product Delivery Option (CBPDO), HOLDDATA is included.

If the CBPDO for IMS 12 is older than two weeks by the time you install the product materials, contact the IBM Support Center or use S/390® SoftwareXcel to obtain the latest PSP bucket information.

You can also obtain the latest PSP bucket information by going to the Preventive Service Planning bucket page at:


For program support, go to the Software Support website at:

http://www.ibm.com/software/support/

Upgrades and Subsets

PSP buckets are identified by Upgrades, which specify product levels, and Subsets, which specify the FMIDs for a product level.

Table 8-1 shows the Upgrade and Subset values for IMS 12.

<table>
<thead>
<tr>
<th>Upgrade</th>
<th>Subset or subsets</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMS1200</td>
<td>HMK1200</td>
<td>IMS V12.01.00 System Services</td>
</tr>
<tr>
<td>IMS1200</td>
<td>JMK1201</td>
<td>IMS V12.01.00 Database Manager</td>
</tr>
<tr>
<td>IMS1200</td>
<td>JMK1202</td>
<td>IMS V12.01.00 Transaction Manager</td>
</tr>
<tr>
<td>IMS1200</td>
<td>JMK1203</td>
<td>IMS V12.01.00 Extended Terminal Option</td>
</tr>
<tr>
<td>IMS1200</td>
<td>JMK1204</td>
<td>IMS V12.01.00 Recovery Level Tracking</td>
</tr>
<tr>
<td>IMS1200</td>
<td>JMK1205</td>
<td>IMS V12.01.00 DB Level Tracking</td>
</tr>
<tr>
<td>IMS1200</td>
<td>JMK1206</td>
<td>IMS V12.01.00 Java On Demand features</td>
</tr>
<tr>
<td>IMS1200</td>
<td>HIR2220/1020</td>
<td>Internal Resource Lock Manager (IRLM) V2.2</td>
</tr>
<tr>
<td>IMS1200</td>
<td>HIR2230</td>
<td>Internal Resource Lock Manager (IRLM) V2.3</td>
</tr>
<tr>
<td></td>
<td>HIR2230/1031</td>
<td></td>
</tr>
</tbody>
</table>
8.1.2 Support procedures

Table 8-2 identifies the function modification IDs (FMID) and component IDs (COMPID) for IMS 12 for use when you have to report any problems to your IBM Support Center.

<table>
<thead>
<tr>
<th>FMID</th>
<th>COMPID</th>
<th>Component name</th>
<th>IBM RETAIN® release</th>
</tr>
</thead>
<tbody>
<tr>
<td>HMK1200</td>
<td>5635A0300</td>
<td>System Services</td>
<td>200</td>
</tr>
<tr>
<td>JMK1201</td>
<td>5635A0300</td>
<td>Database Manager</td>
<td>201</td>
</tr>
<tr>
<td>JMK1202</td>
<td>5635A0300</td>
<td>Transaction Manager</td>
<td>202</td>
</tr>
<tr>
<td>JMK1203</td>
<td>5635A0300</td>
<td>Extended Terminal Option</td>
<td>203</td>
</tr>
<tr>
<td>JMK1204</td>
<td>5635A0300</td>
<td>Recovery Level Tracking</td>
<td>204</td>
</tr>
<tr>
<td>JMK1205</td>
<td>5635A0300</td>
<td>DB Level Tracking</td>
<td>205</td>
</tr>
<tr>
<td>JMK1206</td>
<td>5635A0300</td>
<td>IMS Java On Demand features</td>
<td>206</td>
</tr>
<tr>
<td>HIR2220</td>
<td>569516401</td>
<td>Internal Resource Lock Manager V02.02.00</td>
<td>220</td>
</tr>
<tr>
<td>HIR2230</td>
<td>569516401</td>
<td>Internal Resource Lock Manager V02.03.00</td>
<td>230</td>
</tr>
</tbody>
</table>

8.2 Fix Category HOLDDATA

SMP/E for z/OS has been extended to help simplify the task of verifying that required software fixes identified in PSP buckets are installed. PSP buckets identify required software fixes for new hardware devices, toleration and coexistence of new software releases, and for enabling new functions. IBM consolidates the lists of required fixes from PSP buckets and produces new Fix Category (FIXCAT) HOLDDATA to identify those fixes.

IMS is now providing fix category data. Examples of IMS 12 categories include IBM.Coexistence.IMS.V12 and IBM.TargetSystem-RequiredService.IMS.V12.

SMP/E uses the new type of ++HOLD statement to identify APARs and their fix categories, and the PTF that resolves the APAR.

The following list is an example of the sequence of events to determine what IMS 12 coexistence service has not been applied:
1. Download the current Enhanced Holddata.
2. Issue the SMP/E RECEIVE command on the current Enhanced Holddata.
3. Run the SMP/E REPORT MISSINGFIX command (Example 8-1).

**Example 8-1 The MISSINGFIX command**

```bash
SET BOUNDARY(GLOBAL) /* IMS Global zone */.
REPORT MISSINGFIX ZONES(targetzone)
FIXCAT(IBM.Coexistence.IMS.V12).
```
The FIXCAT operand can also be used on the APPLY and ACCEPT commands. Example 8-2 shows the APPLY specification. On the APPLY command, we use the fix category as the SOURCEID value. SMP/E then applies any PTFs that are included in that fix category.

Example 8-2  Using FIXCAT as part of an APPLY command

```
SET BOUNDARY(targetzone).
APPLY SOURCEID(IBM.Coexistence.IMS.V12) CHECK.
```

8.2.1 IMS-specific fix categories

To enhance the identification of cross-product dependency fixes needed for IMS, the following fix categories have been added to IMS APARs and PTFs:

- **IBM.Coexistence.IMS.V10**: Fixes that allow IMS 8 and 9 to coexist with, and fall back from, IMS 10.
- **IBM.Coexistence.IMS.V11**: Fixes that allow IMS 9 and 10 to coexist with, and fall back from, IMS 11.
- **IBM.Coexistence.IMS.V12**: Fixes that allow IMS 10 and 11 to coexist with, and fall back from, IMS 12.
- **IBM.TargetSystem-RequiredService.IMS.V10**: Fixes required on other IBM products to allow them to run with IMS 10.
- **IBM.TargetSystem-RequiredService.IMS.V11**: Fixes required on other IBM products so that they can run with IMS 11.
- **IBM.TargetSystem-RequiredService.IMS.V12**: Fixes required on other IBM products so that they can run with IMS 12.

8.2.2 Examples of non-IMS-specific fix categories

Another use of fix categories is to identify APARs/PTFs that have been flagged as HIPER.

- **IBM.ProductInstall-RequiredService**

Other products running on z/OS also include fix categories such as the following examples, which we can use to resolve cross-product dependencies with IMS:

- **IBM.Coexistence.z/OS.V1R11**
- **IBM.Function.SYSPLEXDataSharing**
- **IBM.Device.Server.z9-EC-2094.zAAP**
- **IBM.Device.Disk.DS8000-2107**

Again, we can use the SMP/E REPORT MISSING FIX command used to identify IMS 12 service not installed that has a cross-product dependency with these categories.

For a complete list of FIXCATs, see the IBM Fix category values and descriptions page at: [http://www.ibm.com/systems/z/os/zos/smp/e/fixcategory.html](http://www.ibm.com/systems/z/os/zos/smp/e/fixcategory.html)

8.3 Coexistence with IMS 12

IMS 12 can coexist with previous versions of IMS. In some cases, migration and coexistence APARs and related PTFs need to be applied to the previous version. See Table B-1 on page 455.
8.4 General coexistence considerations

IMS 12 can coexist with previous versions, so that existing applications and data can be used without change.

Keep in mind the following general coexistence considerations:

- A full build of all application controls blocks (ACBs) for all existing program specification blocks (PSBs) and database descriptor (DBDs) must be run before starting IMS.
- A stage1 and stage2 ALL gen must be run. You must restart online systems with the FORMAT ALL operand.
- If you are installing multiple copies of IMS systems at different release levels sharing the same logical partition (LPAR), all systems must use the latest version of the IMS type-2 and Database Recovery Control (DBRC) type 4-SVCs.
- If you are still using IMS 10, the IMS 12 copy of DFSAMFD0 must be installed.
- The Offline Dump Formatter from IMS 12, IMS 11, or IMS 10 works without modification if the appropriate formatter library is used.
- You must restart and complete or otherwise recover any failed applications before migrating to IMS 12.

Restriction: You cannot restart a failed application with extended restart across different releases of IMS.

8.4.1 DBRC coexistence considerations

Both IMS 10 and IMS 11 need small programming enhancement (SPE) fixes applied to DBRC to support migration to IMS 12 and coexistence with an IMS 12 recovery control (RECON):

- IMS 11 to IMS 12
  - IMS 11 DBRC can coexist with IMS 12 if PM05244/UK62971 is applied to IMS 11.
- IMS 10 to IMS 12
  - IMS 10 DBRC can coexist with IMS 12 if PM05243/UK62970 is applied to IMS 10.

In both cases, you need to upgrade the RECONs to IMS 12.

UPGRADE RECON

IMS 10 and IMS11 RECONs can be upgraded to IMS 12 by executing the DBRC utility (DSPURX00) and using the CHANGE.RECON UPGRADE command with an IMS 12 SDFSRESL library.

After the RECON data set has been upgraded, the SPE allows DBRC to convert records to the appropriate release format depending on whether the record is being written or read. This way, IMS 10 and IMS 11 can use the RECONs after they have been upgraded to IMS 12.

However, the SPE does not allow the downlevel DBRC to use the new function.

Restriction: After a RECON data set has been upgraded to the IMS 12 level, it is not accessible to any IMS 10 or IMS 11 systems that does not have the DBRC Coexistence SPE applied.
8.4.2 Exit routine coexistence considerations

The Standard User Exit Parameter List (SXPL) changed with IMS 11. Users who are migrating from IMS 10 to IMS 12 must reassemble any exits that are sensitive to the version of the SXPL.

The user exit enhancements in IMS 11 introduced version 6 of the list (SPXPLVER6). Users migrating from IMS 11 to IMS 12 do not need to make any changes to their exits.

The IMS Connect exit parameter list (HWSEXPRM) was changed in IMS 11. Users migrating to IMS 12 from IMS 10 must reassemble and rebind any IMS Connect user exits that use HWSEXPRM to pick up the changes.

The TM and MSC Message Routing and Control user exit from IMS 10 works without modification with IMS 12, but the routine must be reassembled.

8.4.3 Fast Database Recovery coexistence considerations

A Fast Database Recovery (FDBR) region must be at the same release level as the IMS system it is tracking.

The MODBLKSA, MODBLKSB and MODSTAT data sets are no longer used by FDBR. (The information about MODBLKS is read from the IMS checkpoint log records.) Remove those DD statements from the FDBR job control language (JCL).

8.4.4 Common Queue Server coexistence

The following general coexistence considerations exist for Common Queue Server (CQS):

- A control region at version “n” can register with a CQS at version n, n+1, or n+2.
  - A control region cannot register with an earlier CQS.
- A CQS at version “n” can connect to the same CF structure as a CQS at any of the following versions: n-2, n-1, n, n+1, n+2.
- Any supported version of CQS can run on the same central processor complex (CPC).

8.4.5 IMSplex coexistence considerations

This section lists coexistence considerations for IMSplexes.

Common Service Layer coexistence

You can have some Structured Call Interface (SCI), Operations Manager (OM), Resource Manager (RM) and IMS subsystems on IMS 12 while other subsystems are on IMS 10 or IMS 11. IMS 10 and IMS 11 require some coexistence SPEs to work with IMS 12 RM. The recommendation is to use IMS 12 Common Service Layer (CSL) address spaces if any IMS subsystem is running with IMS 12.

Resource management

APAR/PTF PM32951/UK68883 and PM19025/UK63960 are required on IMS 10 if a CSL RM version 1.3 is being used.

APAR/PTF PM32766/UK68882 and PM19026/UK63964 are required on IMS 11 if a CSL RM version 1.4 is being used.
8.4.6 Program specification blocks changes

IMS 12 includes changes to the database descriptor vector table and the PSB prefix. For an IMS 10 or IMS 11 DLIBATCH or ACBGEN job to use a PSB generated with the IMS 12 macros, you need to apply APAR/PTF PM23840/UK64012 to IMS 10 and APAR/PTF PM23843/UK64013 to IMS 11.

8.4.7 Generalized sequential access method changes

If a PSB that uses a generalized sequential access method (GSAM) database is generated by using IMS 12 macros and ACBGENed is generated by using IMS 11 or IMS 10, you must apply APAR/PTF PM32548/UK65496 to IMS 10 and apply APAR/PTF PM32390/UK65495 to IMS 11 for coexistence.

8.4.8 Log records

If you have any application programs that process IMS log records, determine whether the programs are affected by the changes to the log records.

You can assemble DSECTs for IMS log records by using the ILOGREC macro.

Table 8-3 shows the log records that are new or changed with IMS 12.

<table>
<thead>
<tr>
<th>Type</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>x'22'</td>
<td>Changed</td>
<td>New subcodes x'0D' for recoverable <code>UPDATE POOL</code> command, and x'0E' for recoverable <code>UPDATE IMS</code> command.</td>
</tr>
<tr>
<td>x'4507'</td>
<td>Changed</td>
<td>New logger statistics for OLDS and write-ahead data set (WADS).</td>
</tr>
<tr>
<td>x'4513'</td>
<td>Changed</td>
<td>New MSC statistics for TCP/IP links.</td>
</tr>
<tr>
<td>x'6701'</td>
<td>Changed</td>
<td>Macro DFSL6701 changed to bilingual.</td>
</tr>
<tr>
<td>x'67D0'</td>
<td>Changed</td>
<td>Added subtype x'1B' for long lock timeout, subtype x'15-01' internal processing errors, and DFSBCB section for subtype x'02'.</td>
</tr>
<tr>
<td>x'9904'</td>
<td>Changed</td>
<td>RACF user ID is added when they are produced by batch (DLI or DBB) jobs.</td>
</tr>
</tbody>
</table>

8.4.9 MSC coexistence considerations

IMS 10 and later systems support IMS Multiple Systems Coupling (MSC) and shared-queue networks of mixed IMS releases.

The TM and MSC Message Routing and Control user exit routine (DFSMSCE0) from IMS 10 and IMS 11 works without modification with IMS 12, but the exit must be reassembled.

New function is added to the IMS 12 sample DFSMSCE0 exit for XCF/AOS.

8.4.10 Syntax Checker coexistence considerations

The IMS 12 Syntax Checker supports IMS 10, IMS 11, and IMS 12. Ensure that the version shown is correct when using Syntax Checker to check parameters from earlier versions of IMS.
8.4.11 IMS utilities coexistence considerations

The Batch Backout, Log Archive, Log Recovery, Log Merge, and Log Analysis utilities function properly only when they process data that was created by an IMS subsystem or batch application program that is at the same release level as the utility program.

The IMS 12 Database Recovery (DFSURDB0) utility accepts log, image copy, HISAM unload, and change accumulation (CA) data sets from IMS 10, IMS 11, or IMS 12.

The IMS 12 Database Change Accumulation (DFSUCUM0) utility accepts log and CA data sets from IMS 10, IMS 11, or IMS 12.

Therefore, you must use the following tools:

- Use DBRC with all Database Change Accumulation and Database Recovery jobs, especially during migration and coexistence.
- Use the IMS 12 utilities when the input data contains logs, image copies, or CA data sets created by the IMS 12 system.

8.4.12 IMS Tools migration and coexistence

The IBM DB2 and IMS Tools enhance the performance of IMS and DB2. These tools have been upgraded and enhanced to work with IMS 12.

For complete information about these tools, including the IMS versions that they support, see the DB2 and IMS Tools for System z page at:

http://www.ibm.com/software/data/db2imstools

8.4.13 IBM Information Center

The IBM Information Management Software for z/OS Solutions Information Center has been updated to include information about IMS 12. You can find the information center at:

http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/index.jsp

8.5 Installation Verification Program

The Installation Verification Program (IVP) process provides materials that you can use as a guide to working with your own IMS systems. The IVP process includes:

- Data set allocation
- Post-installation activities on target libraries
- System definition activities
- SVC considerations
- Authorization considerations
- IMS system preparation activities
- IMS system and application execution activities
8.5.1 Enhancements overview

The IVP is enhanced to add support for the IMSRSC Repository.

In IMS 10 and later, the Variable Export utility can be directly accessed as an option from the IVP Phase Selection panel. With this utility, you can build a data set on IMS 10 or IMS 11 to import into the IVP for IMS 12.

To export IVP variables from IMS 11 and import them into IMS 12, complete the following steps starting with the IVP for IMS 11:

1. In the IVP Environment Options panel (Figure 8-1), select option 3.

```
Help

<table>
<thead>
<tr>
<th>IVP</th>
<th>IVP Environment Options</th>
<th>IMS 11.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command ===&gt;</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Select the desired option and press Enter.

Option . . 3

IVP Environments

1. DBB - Database Management (Batch)
2. DBC - Database Management (DBCTL)
3. DBT - Database and Transaction Management (DB/DC)
4. XRF - DB/DC with Extended Recovery Facility (DB/DC with XRF)
5. DCC - Transaction Management (DCCTL)

DFSIX023: DFSIXX01 - Prior session completed successfully for "DBT".
```

Figure 8-1 Starting the IMS 11 IVP dialog
2. Start the IMS 11 IVP dialog the normal way by selecting option 6 of ISPF or from the IMS 11 DFSAPPL panel (Figure 8-2). You can go directly to the variable export function from the IMS 11 DFSAPPL menu. You then run the export process starting from the IVP Variable Export Utility panel (Figure 8-6 on page 291).

Help

IMS Application Menu

Command ===>  

Select an application and press Enter.

1 Single Point of Control (SPOC)
2 Manage resources
3 Knowledge-Based Log Analysis (KBLA)
4 HALDB Partition Definition Utility (PDU)
5 Syntax Checker for IMS parameters (SC)
6 Installation Verification Program (IVP)
7 IVP Export Utility (IVPEX)
8 IPCS with IMS Dump Formatter (IPCS)
9 Abend Search and Notification (ASN)

To exit the application, press F3.

Figure 8-2  IMS 11 DFSAPPL menu
3. After the IVP starts, in the Sub-Option Selection - DBT panel (Figure 8-3), select the suboptions that you normally use for IMS 11. The IMS 11 IVP uses your existing INSTALIB and INSTATBL data sets. The only function that we are accessing from the IVP at this time is the export variables option. Therefore, on the Sub-Option Selection panel, press Enter.

![Figure 8-3 Selecting the suboptions](image)

4. In the Table Merge Request - DBT panel (Figure 8-4), choose option 2, Use existing tables. Press Enter.

![Figure 8-4 Selecting the Use existing tables option](image)
5. In the IMS 11 IVP Phase Selection panel (Figure 8-5), choose option A, and then press Enter.

![Figure 8-5 Choosing option A for variable export](image)

6. In the IVP Variable Export Utility panel (Figure 8-6), complete these steps:
   a. Choose the type of IVP to be exported.
   b. Enter the HLQ value for the INSTATBL data set and the name of the data set that you will use to hold the exported variables.

   The export function writes the current value as XML. The export data set is a simple flat file (or PDS/PDSE member) and can be edited, with care, to globally change values as needed before import. For example, change any occurrences of a high level qualifier that has IMS version as part of its name before running the import. This approach saves time when you run the import and variable gathering phases in the IMS 12 IVP dialog.
Chapter 8. Installation and migration considerations

**Figure 8-6   Entering the INSTATBL and export data set names**

To avoid the need to change JES to add the IMS 12 PROCLIB so that the IMS cataloged procedures can be executed without JCL error, you can update these two variables in the export data set (Example 8-3).

**Example 8-3   XML variables in the IVP Export file**

```
<var>IXUJESC3</var> <val>// JCLLIB ORDER=IMS12Q.PROCLIB</val>
<var>IXUJESC2</var> <val>/*JOBPARM S=*</val>
```

Updating the variables ensures that all JCL created by the file tailoring phase 3 (FT3) process for IMS 12 includes the statements in Example 8-4. This way, all jobs submitted on any z/OS in your sysplex can find cataloged procedures from your IMS 12 PROCLIB data set.

**Example 8-4   Additional JCL generated by the IVP file tailoring process**

```
/*JOBPARM S=* 
// JCLLIB ORDER=IMS12Q.PROCLIB
```
7. Start the IMS 12 IVP dialog for the first time, either from option 6 by directly running the IVP program (DFSIXC01), or from the IMS 12 DFSAPPL menu program. Figure 8-7 shows how to start the IVP dialog directly.

```
Menu List Mode Functions Utilities Help

Enter TSO or Workstation commands below:

###> ex 'ims12q.sdfsclst(dfsixc01)' 'hlq(ims12q)'

Place cursor on choice and press enter to Retrieve command

=> ex 'ims12q.sdfsexec(dfsappl)' 'hlq(ims12q)'
=>
=>
=>
=>
=>
=>
=>
=>
=>

Figure 8-7  Invoke the IVP dialog from ISPF
```

8. In the logo panel (Figure 8-8), which is displayed the first time you run the IMS 12 IVP, press Enter to close it. On subsequent runs, you start at the environment selection panel (Figure 8-10 on page 293).

```
==== ===       ===  ==== ==
==== ====     ==== ========
==   ====   ====   ===  ==
==   ===== =====     ===
==   == ===== ==       ===
==   ==  ===  ==  ==    ===
==== ===  ===  === =========
==== ===   =   === == =====

Information Management System (IMS)

IVP Dialog
for
IMS Version 12.1

ENTER to continue or END to exit

Figure 8-8  The IMS 12 IVP logo panel
9. In the IMS notices panel (Figure 8-9), which is only displayed the first time you access the system, press Enter.

![Figure 8-9  IMS 12 IVP Notices pane (presented the first time only)](image)

10. In the IMS 12 IVP Environment Options panel (Figure 8-10), choose the type of IMS system that will be built by the IMS 12 IVP. Most likely your choice is the same type used to start the IMS 11 IVP dialog (Figure 8-1 on page 287) or to export the IMS 11 IVP variables shown in Figure 8-6 on page 291.

![Figure 8-10  Choosing the IMS system environment](image)
11. In the Sub-Option Selection panel (Figure 8-11), select the suboptions to use for the IMS 12 IVP. Ensure that your selections in this panel include the same options as used on IMS 11 plus any additional options you want to use on IMS 12. For more information, see Figure 8-3 on page 289. Then press Enter.

---

Help

........................................................................................................................................
IVP Sub-Option Selection - IMS 12.1
Command ===>

Select the desired Sub-Options and press ENTER

/ IRLM - Use IRLM in IVP Applications
/ FP - Use Fast Path in IVP Applications
/ ETO - Use Extended Terminal Option
  CQS - Add CQS to CSL Applications
  RACF - Use RACF Security
  JAVA - Use JAVA Applications and Open Database
  PRA - Use Parallel RECON Access
  ICON - Use IMS Connect
  REPO - Use IMSRSC Repository
  COUT - Use Callout Applications

Note: Your Sub-Option selection affects the user variables, jobs, and tasks that will be presented. If you later change your selection, you must redo the IVP Table Merge, Variable Gathering, File Tailoring, and Execution processes. RACF is required when Java sub-option is selected.

**Figure 8-11  Choosing the IVP suboptions**
12. In the Sub-Option Change Verification panel (Figure 8-12), confirm your selections and then press Enter if the list is correct.

```
Help
Sub-Option Change Verification - DBT
Command ===>

The Sub-Options you have just chosen are not the same as the Sub-Options which were last active. If you change Sub-Options, Table Merge and the three Dialog Phases must be re-run from the beginning.

From To
Y Y - IRLM - Use IRLM in IVP Applications (not available for DCCTL)
Y Y - FP - Use Fast Path in IVP Applications (not available for DCCTL)
Y Y - ETO - Use ETO (not available for Batch and DBCTL)
N Y - CQS - Add CQS Applications (not available for Batch and DBCTL)
N Y - RACF - Use RACF Security (not available for Batch)
N Y - JAVA - Use JAVA Applications and Open Database
N N - PRA - Use Parallel RECON Access (not available for Batch)
N Y - ICON - Use IMS Connect
N Y - REPO - Use IMS Repository
N Y - COUT - Use Callout Applications

To confirm your change of Sub-Options: Press ENTER
To return to the Sub-Option Selection menu: Press END
```

Figure 8-12  Confirming your suboption selections

13. In the Table Merge Request panel (Figure 8-13), accept the default selection of option 1, and then press Enter to begin the table merging process.

```
Help
IVP Table Merge Request - DBT IMS 12.1
Command ===>

The IVP Dialog is driven from a set of ISPF tables which contain information about the variables, JOBs, TASKs and sequence of presentation you will need to perform the verifications.

Since the tables will be updated by the dialog, working copies must be made the first time you use the dialog.

If service is applied to your IMS system, or if you decide to use the IVP dialog to build a different environment, then either the existing copies must be updated or new copies created.

Please indicate whether you wish to perform Table Merge/Create:

1  1. YES - Create / Update working tables from master tables.
   2. NO  - Use existing tables.
```

Figure 8-13  Choosing to create or update the working tables
Figure 8-14 shows table merge running. No actions are required on this panel.

<table>
<thead>
<tr>
<th>IVP</th>
<th>VG Table Merge In Progress - DBT</th>
<th>IMS 12.1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Table Merge Progress Indicator</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Variable Gathering Table . . . : DFSIXBV3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Current Row . . . . . . : IXUMBKA</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Percent completed . . . . : 59</td>
<td></td>
</tr>
<tr>
<td></td>
<td>File Tailoring Table . . . . : DFSIXBF3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Current Row . . . . . . : Patience</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Percent completed . . . . : 000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Execution Table . . . . . : DFSIXBE3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Current Row . . . . . . : Patience</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Percent completed . . . . : 000</td>
<td></td>
</tr>
</tbody>
</table>

Please do not interrupt this process.

Figure 8-14 Variable Gathering Table Merge runs as normal

While the table merge process runs, your panel shows an update of the progress. Do not interrupt that variable gathering table merge process.

14. When table merge is complete and the completion panel (Figure 8-15) is displayed, press Enter.

When variable gathering table merge is complete the Phase Complete flags are reset, which forces you to revisit the Variable Gathering phase 1 (VG1) and File Tailoring phase 3 (FT3) phases before moving to the Execution phase 6 (EX6) phase. In Phase VG1, you can import the variables that have been exported from IMS 11.

<table>
<thead>
<tr>
<th>IVP</th>
<th>Table Merge has completed - DBT</th>
<th>IMS 12.1</th>
</tr>
</thead>
</table>
| Command =>

The Table Merge process has completed and the Phase Complete flags have been turned off for all phases.

If Table Merge has just been performed for the first time for this option, then the resetting of Phase Complete flags is of no special interest.

If Table Merge has been performed for some other reason, then the resetting of Phase Complete flags will force you to revisit each of the phases in sequence (Variable Gathering, File Tailoring, and Execution). Make use of this opportunity to examine the tables for changes (the "!" indicator will be set in the action field for items which have been added or changed by service). Your position in each phase has been retained so that you may return to your last position after you have browsed for changes.

Press ENTER to continue.

Figure 8-15 Table Merge process completed
15. In the IVP Phase Selection panel (Figure 8-16), accept the preselected option of VG 1 and then press Enter to start the Variable Gathering phase 1.

![Help](image)

Help

Command ===> 

Select the desired Phase and positioning option and press ENTER

1 A. Variable Export Utility (Export variables to a data set)
   VG s Variable Gathering s (Define user values for variables)
   1. VG1 Start/Restart from the beginning of the phase
   2. VG2 Start/Restart from the last known position within the phase

   FT s File Tailoring s (Create customized INSTALIB members)
   3. FT1 Start/Restart from the beginning of the phase
   4. FT2 Start/Restart from the last known position within the phase
   5. FT3 Start/Restart from the beginning of a selected step

   EX s Execution s (Run the IVP jobs)
   6. EX1 Start/Restart from the beginning of the phase
   7. EX2 Start/Restart from the last known position within the phase
   8. EX3 Start/Restart from the beginning of a selected step

Figure 8-16 Choosing the Variable Gathering phase 1
16. In the Variable Gathering (LST Mode) - DBT panel (Figure 8-17), enter the special “IMP” action code to start the variable gathering import process. Ignore any variable values that you see prefilled in this panel. The import process replaces them.

17. After starting the variable import function, in the IVP Export File Name panel (Figure 8-18), enter the name of the XML data set exported from IMS 11.
18. Enter the name of the data set (or PDS/PDSE member) that contains the variables that you have exported from IMS 11 then press Enter.

While each variable is read and updated from the XML file being imported, a progress panel (Figure 8-19) is displayed to show the status of the process.

```
Figure 8-19  All variable are now updated from the IMS 11 exported data
```

The variable gathering process has a progress indicator to show which variables have been updated using the data from IMS 11. Do not interrupt this variable update process.

Now that the variable import process is complete, we return to variable gathering phase 1 but with the values read from the IMS 11 system (Figure 8-20).

```
Figure 8-20  Changing the variables as needed from the IMS 11 values
```

You have now imported the variables from IMS 11. Review all the imported values and change them as needed for the new IMS 12 system.
8.5.2 File tailoring

After the variable gathering is complete, press End to terminate the process and to move on to the file tailoring phase 3 (FT3) process.

8.5.3 IVP jobs and tasks

The jobs and tasks that are presented in groups by the IVP dialog are determined by your choice of environment option and distribution media.

The last group listed, Steps Zx for index of additional PDS members, does not identify jobs or tasks in the IVP process. It identifies the members of DFSSLIB and DFSISRC libraries that support the IVP process.

Figure 8-21 shows a sample of the initials jobs and tasks presented by the IVP dialog.

Figure 8-21 Execution phase 6 of the IMS 12 IVP

You can print additional documentation for the IVP jobs, tasks, and variables by using the DOC action during the file-tailoring phase or the execution phase of the IVP dialog. Use the IVP dialog to obtain current information regarding IVP jobs and tasks. In these lists, the jobs and tasks are presented in the same sequence that is used by the IVP dialog.
Table 8-4 lists the naming convention used for jobs and tasks presented as IV_ssnnt.

Table 8-4  IVP naming convention for jobs and tasks

<table>
<thead>
<tr>
<th>IV_ssnnt</th>
<th>Meaning</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_</td>
<td>Underscore</td>
<td>The environment that you selected on the first panel:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1: DBB - Batch</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2: DBC - DBCTL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3: DBT - DB/DC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4: XRF - DB/DC with XRF</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5: DCC - DCCTL</td>
</tr>
<tr>
<td>ss</td>
<td>Step</td>
<td>IVP step</td>
</tr>
<tr>
<td>nn</td>
<td>Number</td>
<td>The number of the item within the step</td>
</tr>
<tr>
<td>t</td>
<td>Item type</td>
<td>J (JOB) A PDS member with the same name is placed into INSTALIB during the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>file-tailoring phase. Items of type J are intended to be submitted for</td>
</tr>
<tr>
<td></td>
<td></td>
<td>execution.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>T (Task) Tasks represent items of work that must be prepared by the user.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For some tasks, an example is provided in the INSTALIB data set. These</td>
</tr>
<tr>
<td></td>
<td></td>
<td>examples are not intended for execution.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>N (Supporting materials) The INSTALIB data set can also contains members</td>
</tr>
<tr>
<td></td>
<td></td>
<td>that support other jobs such as CLISTs and control statements.</td>
</tr>
</tbody>
</table>

8.5.4 Executing IVP tailored jobs and tasks

The IVP provides a set of example JCL that you can tailor to your environment in the variable gathering phase. It is normal to run through each of the steps sequentially. This process builds your IVP system and runs each function that you selected on the Sub-Option Selection panel (Figure 8-11 on page 294).

To run the IVP jobs and tasks, complete the following steps:

1. In the IVP Phase Selection panel, select option 6, 7, or 8. Each selection within a phase provides a different positioning option.

2. Open each job or task. To view the instructions for each job and task, use the ENT action command.

   For IVP jobs you can browse, edit, or submit the job. Some items are nonexecutable examples, but the browse and edit actions are available to create an executable version of nonexecutable items.

   For IVP tasks, you are provided a scrollable description to assist you in performing the task.

3. Press End or PF3 when you are done.

4. Press Enter again if you completed the execution of all jobs and tasks, or press End to save your work if you want to complete the execution phase later.
8.5.5 Changes to the IVP for IMS 12

IMS 12 has added an IVP for the IMS Repository.

Prerequisite jobs and tasks for the IMS Repository
Before executing the IMS Repository IVP, complete the following steps:

- Cx system definition (SYSDEF) steps
  The C series of steps includes the jobs needed to build the SDFSRESL with a stage1 and
  stage2 system generation.
- Ex Build IVP Appl/System steps
  The E series of steps includes the jobs needed to compile and bind application programs
  and the steps that add sample control statements to the IMS PROCLIB data set.

IVP IMS Repository jobs and tasks
Figure 8-22 shows the new jobs and tasks that are file tailored when you choose the REPO -
Use IMSRSC Repository option in the Sub-Option Selection panel (Figure 8-11 on
page 294).

Three of these steps are familiar if you have run any of the IVP steps that use the CSL
functions. The RM cannot be started before the repository is running. Other new steps are
listed here:

- IV_U101J scratches and reallocates the data sets needed to perform the Repository
  usage for DRD resources.
- IV_U104J starts the repository server address space.
- IV_U105J adds an IMSRSC repository to the repository server’s catalog and then starts it.
Chapter 8. Installation and migration considerations

- **IV_U202J** lists status information for all repositories.
- **IV_U204J** populates data into the IMS repository:
  a. It converts the data in MODBLKS into the RDDS format.
  b. It copies data from RDDS data set to the IMS repository.
- **IV_U205J** renames the IMS repository in the repository catalog.
- **IV_U206J** lists detailed status information for a single repository. See Example 8-5 for sample output.

**Example 8-5  Example of output from job IV_U206J**

```
LIST REPOSITORY(IMS12XRP)
Repository Name . : IMS12XRP

  Last updated date/time : 2011/07/26 01:44:31 IMSR3
  Status . . . . . . . . : OPEN
  Auto-open . . . . . . : YES
  Security Class . . . . : NOT DEFINED

  Repository Data Set pair 1
   Index (RID) . . : IMS12Q.IMS12X.REPO.IMSPRI.RID
   Member (RMD) . : IMS12Q.IMS12X.REPO.IMSPRI.RMD
   Status . . . . : COPY1

  Repository Data Set pair 2
   Index (RID) . . : IMS12Q.IMS12X.REPO.IMSSEC.RID
   Member (RMD) . : IMS12Q.IMS12X.REPO.IMSSEC.RMD
   Status . . . . : COPY2

  Repository Data Set pair 3
   Index (RID) . . : IMS12Q.IMS12X.REPO.IMSSPR.RID
   Member (RMD) . : IMS12Q.IMS12X.REPO.IMSSPR.RMD
   Status . . . . : SPARE
```

FRP4750I - LIST command processing completed successfully

- **IV_U207J** updates the definitions for a repository.
- **IV_U208J** deletes a repository in the catalog.
- **IV_U209J** tries to start a repository that has already been deleted.
- **Task IV_U401T** and **job IV_U402J** closes everything down and scratches the data sets created for this step of the IVP.

### 8.6 Syntax Checker

Syntax Checker is an ISPF application that helps you define, verify, and validate parameters and their values in the members of the PROCLIB data set. Online Help is available at the parameter level and provides assistance in moving to a new IMS release by identifying new parameters and any obsolete parameters from the previous release.

Syntax Checker saves the parameters to appropriate PROCLIB members in the correct format. You can also use the Syntax Checker to migrate your configuration members to IMS 11.
8.6.1 Starting Syntax Checker

Start IMS Syntax Checker by using either of the following two methods:

- Issue the following command in ISPF option 6, where HLQ is the high-level qualifier of the IMS data sets:
  
  ```
  EXEC 'hlq.SDFSEXEC(DFSSCSRT)' 'HLQ(hlq)
  ```

- In the IMS Application Menu panel (Figure 8-23), select option 5.

```
Help

---------------------------------------------
IMS Application Menu
Command ===> 5

Select an application and press Enter.

1  Single Point of Control (SPOC)
2  Manage resources
3  Reserved for future use
4  HALDB Partition Definition Utility (PDU)
5  Syntax Checker for IMS parameters (SC)
6  Installation Verification Program (IVP)
7  IVP Export Utility (IVPEX)
8  IPCS with IMS Dump Formatter (IPCS)
9  Abend Search and Notification (ASN)

To exit the application, press F3.
```

Figure 8-23  IMS Application Menu
8.6.2 Using Syntax Checker

After you start Syntax Checker, the IMS Parameter Syntax Checker panel (Figure 8-24) is displayed. Here, you can enter the PROCLIB data set name and the name of the member to be processed. If you do not enter the member name, a member list is displayed. Select the member you want to process from the list.

![Syntax Checker entry panel]

**Figure 8-24 Syntax Checker entry panel**

When you press Enter from the main panel, Syntax Checker reads the input file and tries to determine the IMS release and type of control region.

If Syntax Checker cannot determine this information, one of the following entry panels opens:

- IMS Release and Control Region Type entry panel
- IMS Release entry panel

If Syntax Checker can determine the information it requires from comments in the member, the Syntax Checker Keyword Display panel is shown (Figure 8-26 on page 307).
## 8.6.3 IMS Release and Control Region Type entry panel

The “IMS Release and Control Region Type” entry panel shown in Figure 8-25 provides Syntax Checker with IMS release and control region information that is required to process the member correctly.

<table>
<thead>
<tr>
<th>File</th>
<th>Help</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IMS Parameter Syntax Checker</td>
</tr>
<tr>
<td>Command ==&gt;</td>
<td></td>
</tr>
</tbody>
</table>

Enter the following information and press enter.

**IMS Release**
- 1. IMS 12.1
- 2. IMS 11.1
- 3. IMS 10.1

**Type of Control Region**
- 1. DBCTL Control Region
- 2. DCCTL Control Region
- 3. DB/DC Control Region
- 4. FDBR Region
- 5. DLI/DBB Batch

*Figure 8-25  IMS Release and Control Region Type entry panel*

When Syntax Checker saves the member, it adds comment lines to the top of the member saving this information. The next time Syntax Checker processes the member, this panel does not display.

After you type in the data and press Enter, the Syntax Checker Keyword Display panel opens.
8.6.4 Keyword Display panel

The Keyword Display panel (Figure 8-26) of IMS Syntax Checker displays the keywords and their values, and indicates whether any syntax errors exist.

File Edit View Help

---------------------------------------------
IMS 12.1 Parameters for DB/DC
Command ===>

Press enter (without other input) to check for errors.

Data Set Name . . : INSTALIB(DFSPBV11)
IMS Release . . . : 12.1

Sel Codes: C = Comment D = Delete I = Insert P = Process / = Select

<table>
<thead>
<tr>
<th>Sel Keyword</th>
<th>Value</th>
<th>Description</th>
<th>More:</th>
</tr>
</thead>
<tbody>
<tr>
<td>_</td>
<td>ALOT = 60</td>
<td>ETO Auto Logon Off Time</td>
<td>+</td>
</tr>
<tr>
<td>_</td>
<td>AOIP = ______</td>
<td>AOI Pool Upper Limit</td>
<td></td>
</tr>
<tr>
<td>_</td>
<td>AOIS = N</td>
<td>ICMD Security Option</td>
<td></td>
</tr>
<tr>
<td>_</td>
<td>APPC = _</td>
<td>Activate APPC/IMS (Y</td>
<td>N)</td>
</tr>
<tr>
<td>_</td>
<td>APPLID1 = SCSIMSBB</td>
<td>VTAM Applid of Active IMS System</td>
<td></td>
</tr>
<tr>
<td>_</td>
<td>APPLID2 = ______</td>
<td>VTAM Applid of XRF Alternate System</td>
<td></td>
</tr>
<tr>
<td>_</td>
<td>APPLID3 = ______</td>
<td>VTAM Applid of RSR Tracking System</td>
<td></td>
</tr>
<tr>
<td>_</td>
<td>ARC = 01</td>
<td>Automatic Archive: 1-99, 0 - NOT Autom.</td>
<td></td>
</tr>
<tr>
<td>_</td>
<td>ARMRST = N</td>
<td>Allow MVS ARM to Restart (Y</td>
<td>N)</td>
</tr>
<tr>
<td>_</td>
<td>AS .................</td>
<td></td>
<td></td>
</tr>
<tr>
<td>_</td>
<td>AU * Member DFSPBV11 will be processed under IMS 12.1 (DB/DC ) *</td>
<td></td>
<td></td>
</tr>
<tr>
<td>_</td>
<td>BS ..........................</td>
<td></td>
<td></td>
</tr>
<tr>
<td>_</td>
<td>CIOP = ______</td>
<td>CIOP Pool Upper Limit</td>
<td></td>
</tr>
</tbody>
</table>

Figure 8-26  Keyword Display panel

If you select Display new from the View menu, Syntax Checker lists only those parameters that are new in the IMS version you have selected.

To display the default values for parameters, press the F6 function key; press the key again to toggle between displaying and not displaying defaults. The default values are displayed in the description field of the parameter.

After modifying the member, press the Enter key without making any other modifications. Syntax-value checking is performed.

If errors exist, the first keyword with an error moves to the top of the display and an error message is displayed. When all errors are resolved and you have modified the member as required, you can save the member to the originally selected PROCLIB and member or to a different PROCLIB and member.

8.6.5 Exiting Syntax Checker

When you exit Syntax Checker, it prompts you to save any unsaved changes.
8.7 Installation and migration tasks

This section provides an overview of the tasks needed to migrate to IMS 12.

8.7.1 Migration considerations

This section describes practices that are useful for a general migration of current IMS installations in regard to the following tasks:

- Preparation tasks
- Installation tasks
- Validation tasks

Preparation tasks

Perform the following tasks before the migration:

- Contact IBM Software Support for current installation, migration, and problem-resolution information; ask for PSP for IMS.

  Before you install IMS 12, check with your IBM Support Center or use Information/Access or Service Link to determine whether additional PSP information is available that you must be aware of.

  The PSP upgrade name for IMS 12 is IMS1200.

  An alternative to using PSP is to use fix category hold data (see 8.2, “Fix Category HOLDDATA” on page 281).

- Read IMS Version 12 Program Directory, GI10-8843, for the most current hardware requirements, software requirements, prerequisites, and installation information.

- Review the IMS installation overview in IMS Version 12 Installation, GC19-3017.

- Review the service that has been applied to your current system. Determine if critical service has been included in the new IMS release. If not, acquire the appropriate service for the new IMS release. Again the new fix categories process will help you to do that.

- Review the functions and enhancements in IMS Version 12 Release Planning, GC19-3019. In particular, review the changes to the following areas:
  - SMP/E, distribution, and system data sets
  - System definition macros
  - Log records
  - RECON records
  - Exit routines
  - Cataloged procedures
  - Control statement members in the PROCLIB data set
  - Utilities
  - Operator commands
  - Operating procedures
  - Messages and abend codes

- Review the z/OS interface considerations in IMS Version 12 System Administration, SC19-3020, which explains SVCs and the SYS1.PARMLIB updates.

- Install prerequisite software and maintenance.

- Determine the availability of updates to IBM IMS Tools, aids, and related products. You can find the latest information on the Installation page at:
Apply coexistence PTFs to your existing lower-level IMS systems that will operate with IMS 12. For a list of these PTFs, see Appendix B, “Recent maintenance: IMS 12 APARs” on page 455.

Examine Hardware Data Compression (HDC) dictionaries to determine if they incorporate IMS versions that are now out of service.

Although rebinding dictionaries is not required when migrating to a new version of IMS, a gradual refresh of these dictionaries to a current release is a good practice.

Evaluate and update IMS exit routines. If you use a RECON I/O Exit Routine (DSPCEXT0), examine it for required changes because of the change in RECON records.

The DFSMSCE0 exit must be reassembled for use with IMS 11.

Similarly, all locally modified IMS Connect exit routines must be reassembled when migrating IMS Connect to IMS 12.

The IMS Connect HWSIMSO0 and HWSIMSO1 exit routines are not included with IMS 12. The HWSSMPL0 and HWSSMPL1 exit routines provide enhanced functionality and are delivered as source code and object modules with IMS 12. Previous versions only supplied the source code. Use them in place of HWSIMSO0 and HWSIMSO1.

Installation tasks

To install IMS 12, complete these tasks:

1. Install IMS 12 by using the SMP/E installation process.

2. Use CBPDO or ServerPac:
   - The CBPDO product package consists of one logical tape (multiple volumes). A CBPDO package that includes IMS can also include other products in the same System Release (SREL). CBPDO also provides service for the products included with the product order. The service includes all PTFs available within one week of order fulfillment. All PTFs are identified by one or more SOURCEIDs, including PUTyymm, RSUyymm, SMCREC, and SMCCOR.
   - ServerPac is a software delivery package. It consists of products and service for which IBM has performed the SMP/E installation steps and some of the post-SMP/E installation steps. To install the package on your system and complete the installation of the software it includes, use the CustomPac Installation Dialog, which is the same dialog used for all CustomPac offerings, including IBM SystemPac® (dump-by-data-set format), IBM ProductPac®, and RefreshPac.

      For IMS, ServerPac allocates, catalogs, and loads all the data sets; sets up the SMP/E environment; supplies a job to update PARMLIB (IEFSSNxx, PROGxx, IEASVCxx, and SCHEDxx); and directs you to start the IVP.

3. Validate your system definition stage1 source. Consider merging some elements from the IVP source with your source.

   Run the IVP. The IMS IVP is used after the installation of a new IMS system; it is used to verify the installation, and can be used sporadically afterwards.

   The IVP Variable Export utility makes the migration of IVP variables between releases easier.

   Running the IVP is optional, but is a good practice. All required installation tasks are done outside the IVP. The IVP verifies that the installation is correct.

4. Install the system prerequisites and your new IMS system (including the pre-generation service).

   The complete set of IMS 12 modules that are necessary for execution are built by a combination of SMP/E processing and running a stage1 or stage2 ALL system definition
process. Most system definition statements from previous IMS releases are compatible with IMS 12.

If necessary, you can use the SMP/E GENERATE command to generate the JCL for jobs that will build the modules that are not built during the system definition process.

5. Install required service that was not included in the pre-generation service.

6. Install any required updates to IBM IMS and DB2 tools, aids, and related products.

7. Install the IMS 12 Type 2 and Type 4 SVC modules created by system generation.

   A z/OS IPL is not required. The SVCs can be installed by running the DFSUSVC0 utility specifying SVCTYPE=(2,4). The utility must be rerun after every z/OS IPL. Therefore, permanently install them at the next scheduled z/OS IPL.

8. Review the IMS Abend Formatting Module considerations in IMS Version 12 System Administration, SC19-3020.

   IMS 12 installs the abend formatting routine (module DFSAFMX0) dynamically. No user setup is required to install DFSAFMD0 on the host z/OS system for IMS 12 as part of the IMS installation.

9. Upgrade the RECON data set by using the CHANGE.RECON UPGRADE command with the IMS 12 DBRC utility.

10. Build application control blocks with a full ACBGEN.

Restriction: It is not possible to run IMS 12 with an ACB library built using IMS 10 or IMS 11. It is not possible to run IMS 10 or IMS 11 with an IMS 12 ACB library. The IMS system will fail to load DBDs and PSBs because they are not compatible.

Validation tasks
Perform the following validation tasks:

1. Validate users' cataloged procedures.

2. Validate user-created members of the PROCLIB data set.

   Use IMS 12 Syntax Checker to convert members from IMS 10 or IMS 11 where appropriate.

3. Validate, reassemble, and rebind exit routines and user modifications, especially IMS Connect exit routines and code that uses IMS control blocks, such as database randomizers.

   Check your exit routines before reassembling.

4. Validate, reassemble, and rebind user programs that process log records.

   Some log record formats have changed.

5. Validate and update operating procedures.

   This refers to recovery, backup, and restart procedures.

6. Review the various execution parameters in the DFSPBxxx member of the PROCLIB data set that can affect performance and migration.

   Review and set the appropriate values for the AOIP, CMDP, DYNP, EMHB, FPWP, HIOP, LUMC, and LUMP parameters to specify an upper limit on the amount of storage a pool can acquire.

   You can also use Syntax Checker to validate the values for the DFSPBxxx parameters.

7. When using MSC to connect IMS systems with different releases, consider all message types and the prefix sizes that accompany them. Such message types include Intersystem
Communication (ISC), Advanced Program-to-program Communication (APPC), and Open Transaction Manager Access (OTMA).

When message queue data sets are used, make the MSGQ LRECL and block sizes are identical across all IMS MSC systems.

A problem can occur when an IMS system is migrated to a new release that uses messages with larger prefix sizes and new prefix segment types.

Messages that contain these new and larger prefixes are sent to an earlier release of IMS because the new and larger prefixes might not fit the message queues of the earlier release of IMS.

8. Consider other products that can be affected by migration. Any product that depends on the format and contents of the IMS log or the RECON data set is potentially affected. Examples include the following products or utilities:
   - IMS Statistical Analysis utility
   - IMS Fast Path Log Analysis utility
   - IMS Log Transaction Analysis utility
   - IMS MSC Log Merge utility
   - CICS
   - Non-IBM products, including user modifications

9. Stop your pre-version 12 system.

10. Restart your IMS 12.

11. Test your IMS 12 system.

### 8.7.2 Discontinued support in IMS

Support is discontinued for various utilities, exit routines, and functions, which are listed in this section.

IMS 10 was the last release of IMS to support the z/OS-based batch DLIModel utility. It is not supported with IMS 11. If you are using this function, you should migrate to the DLIModel utility Eclipse plug-in which is part of the IBM IMS Enterprise Suite.

The JCA 1.0 resource adapter, one of the Java connectors in the IMS DB distributed resource adapter, is stabilized and is no longer being enhanced. You should switch to using the IMS Universal DB resource adapter that is delivered in IMS 11.

IBM has discontinued support for IBM Enterprise Workload Manager™ (EWLM). Therefore, IMS can no longer offer this support. IBM is providing a transition for EWLM 2.1 clients to an IBM STG Lab Services offering. This new offering provides enhanced capabilities over the EWLM 2.1 product.

IMS 10 was the final version of IMS in which the IMS information is delivered in the IBM BookManager® format.

### 8.7.3 Fallback considerations

IMS does not generally support downward compatibility in any major function between releases.

Consider the following steps when preparing your migration fallback plan. This information is intended as a guide to understanding fallback inhibitors, and should not be considered complete.
For each IMS that you are falling back, complete the following steps:

1. Ensure that the status of all databases updated by IMS 12 is correct.
   Establish a new recovery point for these databases by image copying them before
   allowing updates in the fallback release.

2. Resolve DBRC issues. See 8.7.4, “DBRC fallback considerations” on page 312.
   Make sure you have the correct DBRC Coexistence PTF applied to the older IMS
   (PK61583 and PK61582).


4. Install the old version of IMS.
   You must rebuild your ACB library at the IMS version to which you are falling back.

5. Restart IMS.

   You can use the IBM IMS Queue Control Facility for z/OS (QCF) to requeue IMS 12
   messages to IMS 10 or IMS 11 message queues.

8.7.4 DBRC fallback considerations

Certain steps must be taken to revert DBRC to an IMS 10 or IMS 11 level.

Base Primitive Environment-based DBRC
If you have started to use Base Primitive Environment (BPE)-based DBRC and must fall back
to IMS 10, complete these steps:
1. Shut down the IMS control region that is associated with the BPE-based DBRC address
   space.

2. Modify the DBRC procedure to use JCL appropriate for a non-BPE based DBRC region.

3. Restart IMS with DBRCNM referring to the non-BPE DBRC region startup JCL.

You might have to revert either or both of the following exits to older versions if you have
changed these to not be compatible with the old calling interface:

   ▶ DSPCEXT0
   ▶ DSPDCAX0

Database Change Accumulation Utility (DFSUCUM0)
If you fall back from IMS 12 and you have change-accumulation data sets created by IMS 12,
the logs in these data sets are not recoverable because the older utilities cannot process
them. Invalidate the IMS 12 change-accumulation data sets by running an image copy of the
affected databases at the older level.

Lowering the minimum version value
Do not change MINVERS to '12.1' until you are sure that you do not have to fall back to a lower
IMS version. However, if such a fallback becomes necessary, you can reduce MINVERS by
completing these steps:

1. Shut down all IMS 12 subsystems.
   Confirm that IMS 12 subsystem records have been removed from the RECON data set.
   Use the LIST.SUBSYS command to see the subsystem records in the RECON data set. If
   IMS 12 subsystems remain in the RECON, take the necessary recover actions.

2. Determine the status of your databases.
If the databases need recovery with IMS 12 logs, perform those recoveries with IMS 12. If the databases need recoveries to a time before the IMS 12 updates, perform those recoveries with either the IMS 12 or the older version.

3. Use the following commands to delete all IMS 12 log records and all the allocation records on those logs from the RECON data set:
   
   DELETE.LOG  
   DELETE.ALLOC

4. Reset the MINVERS value by issuing a CHANGE.RECON MINVERS command using IMS 12.

   If you receive message DSP1205E (meaning that the database quiesce flags are active), use the CHANGE.DB or CHANGE.DBDS command to turn off the flags. After the flags are turned off, reissue the CHANGE.RECON MINVERS command.

5. Establish new recovery points for all databases updated by IMS 12 by taking image copies of those databases using the older version of IMS.

For more information about the DBRC commands, see IMS Version 12 Commands, Volume 3: IMS Component and z/OS Commands, SC19-3011.

### 8.8 Review of migration considerations

When migrating to a more recent release of IMS, review the following release planning guides that span your specific migration path:

- IMS Version 12 Release Planning, GC19-3019
- IMS Version 11 Release Planning, GC19-2442

#### 8.8.1 Migrating to IMS 12 Database Manager

Specific migration considerations apply when you are migrating from IMS 10 Database Manager to the IMS 12 Database Manager.

**Database quiesce migration considerations**

To use the database quiesce function, the MINVERS field in the RECON data sets must be set to '12.1'.

DBRC API applications that interrogate the output from Query TYPE=DB, TYPE=DBDS, and TYPE=PART requests do not have to be modified if they do not want to access the fields added with the IMS 11 database quiesce enhancement.

Applications that want to get the new output must map to the new output fields and ensure that the DSPAPQHD block returned by DBRC has a minimum version of 3.0.

**Fast Path 64-bit buffer manager**

If your IMS installation did not use 64-bit storage before IMS 12, you have to make 64-bit storage available to use the Fast Path 64-bit buffer manager.

**Fast Path usability and serviceability enhancements**

If you have automated operations that are triggered when messages DFS2555I or DFS2716I are issued because no main storage databases (MSDBs) are defined, you must modify them because these messages are not issued any more.
Open database enhancements migration considerations
You can migrate existing Open Database Access (ODBA) application servers to use the Open Database Manager (ODBM) delivered with IMS 11, and migrate existing IMS DB resource adapter applications to use the new IMS Universal DB resource adapter, as follows:
1. Apply the appropriate coexistence APAR.
2. Add the IMSPLEX parameter and optionally the ODBMNAME parameter to the DFSPRP macro.
3. Reassemble and rebind the DFSxxxx0 load module (where xxxx is the DRA startup table name specified on the APSB call in the AI溴SNM2 field of the AIB).
4. After performing these tasks, you can simplify the ODBA applications by replacing multiple CIMS INIT commands with a single CIMS CONNECT command.

You can optionally migrate application programs that use the existing IMS DB resource adapter, which supports the JCA 1.0 architecture, to the new type-2 interface IMS Universal DB resource adapter, which supports the JCA 1.5 architecture.

WebSphere Application Server for z/OS applications that want to use the Open Database enhancements must deploy the new type-2 interface IMS Universal DB resource adapter.

After you set up ODBM, you can start to use the type-4 interface of the IMS Universal DB drivers.

8.8.2 Migrating to IMS 12 Transaction Manager

When migrating from IMS 10 Transaction Manager to IMS 12 Transaction Manager, keep in mind the IMS considerations as explained in this section.

APPC enhancements
Migrate all IMS systems that participate in a particular shared-queues environment to IMS 12 before attempting to use the APPC enhancements, even though the APPC local logical unit (LU) functions do not have any explicit migration considerations for IMS 12.

OTMA migration considerations
Migration considerations for OTMA involve compatibility with the message control function, OTMA transaction monitoring, and IMS destination routing descriptors.

OTMA transaction monitoring
IMS 12 can perform only limited OTMA transaction monitoring for OTMA clients that do not specify the transaction expiration time in the OTMA prefix or do not exploit the OTMA resources monitoring (such as IMS Connect in IMS 10 without the expiration PTFs).

In these cases, you can activate transaction expiration (but not message-level expiration) by specifying the EXPRTIME parameter in the TRANSACT macro or by issuing either of the following commands:

CREATE TRAN
UPDATE TRAN SET(EXPRTIME(seconds))

OTMA protocol messages regarding OTMA resource information are ignored by IMS Connect and other OTMA clients that have not taken advantage of OTMA resource monitoring.
**IMS destination routing descriptors**

In IMS 12, type-2 commands can be used to dynamically define destination routing descriptors. The OTMA descriptors can also be defined in the DFSYDTx PROCLIB member and are processed when IMS is restarted. Both types can coexist.

In IMS 10, the definition of the destination routing descriptors in DFSYDTx must be entered from the most specific to the most generic destination routing descriptor name. Any destination with a masked character of asterisk (*) has to occur after the group of names that the asterisk is masking. For example, following the order from most specific to most generic, you must list the contents in DFSYDTx as DEST1234, DEST12*, and DEST*.

In IMS 12, this restriction is lifted and the user creating the descriptor entries no longer must be aware of the order. For example, the entries in DFSYDTx can be in any order, such as DEST*, DESTT1234, and DEST12*. OTMA automatically rearranges this internally from most specific to most generic.

### 8.8.3 Migrating to the IMS 12 system

When migrating from IMS 10 systems to IMS 12 systems, keep in mind the IMS considerations as explained in the following sections.

**Migration tasks for ACB library enhancements**

The migration tasks in the following sections are associated with the ACB library enhancements.

**Caching ACBs into 64-bit storage**

To migrate to ACBLIB members in 64-bit memory, complete the following steps:

1. Specify the **ACBIN64** parameter in the DATABASE section of the DFSDFxxx PROCLIB member.
2. Stop IMS.
3. Restart IMS.

**Migrating ACB libraries to use dynamic allocation**

To migrate ACB libraries to use dynamic allocation, complete the following steps:

1. Create DFSMDA members for the ACBLIBA and ACBLIBB data sets.
   - You can place DFSMDA members in a data set specified in either the IMS STEPLIB concatenation or the IMSDALIB concatenation.
   - Remove the IMSACBA and IMSACBB DD statements from the IMS and DL/I JCL procedures.
2. Stop IMS.
3. Restart IMS with the DFSMDA members.

**CQS migration**

Migrate CQS and its control region (or regions) on the z/OS image at the same time. If this is not possible, CQS must be migrated before any of the control regions are migrated.
DBRC migration considerations
These topics describe the considerations and tasks for migrating DBRC to IMS 12.

Base Primitive Environment-based DBRC migration considerations
To start a BPE-based DBRC address space, create new DBRC JCL and update the IMS EXEC parameter, DBRCNM=, to specify the new member name.

If you are going to use your existing DBRC exit routines, there are no migration considerations.

If you want your DBRC exit routines to use the functionality of BPE, you must create new exits (which are based on current exits) that handle the BPE interface. The new exits must have unique names and must be added to the BPE user exit PROCLIB member.

Changes to the RECON data set
Certain records in the RECON data set are new or changed from the records in IMS 10. The following RECON records have increased in size and have new or changed fields as of IMS 12:

- DSPRCNRC: increased in size by 48 bytes
- DSPCHGRC: increased in size by 16 bytes
- DSPDBHRC: increased by 20 bytes

For information about RECON record types, see IMS Version 12 Diagnosis, GC19-3015.

Parallel RECON Access
If you are migrating to IMS 12 from IMS 10 and parallel RECON access (PRA) is in effect, you must ensure that there is no shunted I/O when the upgrade begins. The upgrade process begins with a quiesce close and a check for shunted I/O. The RECON data sets are closed and reopened in LSR mode. The records are upgraded as they are for non-PRA. This includes upgrading the records in COPY1 and then upgrading the records COPY2. After the upgrade completes, the RECON data sets are reopened in PRA mode and the quiesce is ended.

For information about the RECON I/O exit routine (DSPCEXT0), see IMS Version 12 Exit Routines, SC19-3016.

Unconditional deletion of PRILOG or change accumulation information
If you plan to use this function, modify your DBRC security to include the new CLEANUP.RECON resource.

You might have to modify your procedures for maintaining the RECON data set because of the CLEANUP.RECON command.

Consider modifying your automated programs or tools that issue DBRC commands.

Upgrading the RECON data set
To upgrade an IMS 10 or IMS 11 RECON data set, follow these steps:

1. Apply the IMS 12 coexistence SPEs to all IMS 10 and IMS 11 systems before you attempt to upgrade the RECON data set. For information about recent maintenance, see Appendix B, “Recent maintenance: IMS 12 APARs” on page 455.

Jobs that access the RECON data set and do not create subsystem records, such as the Database Change Accumulation utility (DFSUCUM0) and the DBRC utility (DSPURX00), are not protected from having the RECON data set upgraded while they are running on a version of IMS that does not have the appropriate migration and coexistence SPE applied.
When these types of jobs access the RECON data set after the upgrade, the results might be unpredictable. Make sure that no such jobs are running when you upgrade the RECON data set.

2. Make sure you have two active RECON data sets (COPY1 and COPY2) and a spare data set when you attempt to upgrade the RECON data sets while other jobs are accessing them.

3. Issue the `CHANGE.RECON UPGRADE` command. This command performs the following actions:
   - It upgrades the RECON data set without shutting down all IMS activity.
   - It uses the DBRC I/O recovery algorithms to recover from any failures during upgrade, thereby relieving you of the need to make a backup of the RECON data set before upgrade and restoring it in case of a failure.

You must issue this command by using either the IMS 12 DBRC utility (`DSPURX00`) or the IMS 12 DBRC Command API request. If you use DBRC command authorization, consider setting the RECON qualifier as part of your migration process. You can set the RECON qualifier either at the time of upgrade by adding `CMDAUTH()` parameters to the `CHANGE.RECON UPGRADE` command, or after the RECON has been upgraded by issuing a `CHANGE.RECON CMDAUTH()` command.

Consider using RECON command authorization if you use the `CLEANUP.RECON` command.

When you are sure that a fallback to a previous IMS version is unlikely and all systems that access the RECON data set are IMS 12, update the minimum version value to ‘12.1’.

You are not required to change the `MINVERS` value to ‘12.1’ unless you have to use new function that requires this value. The change.recon upgrade process will set ‘10.1’ as a minimum level.

After you set the `MINVERS` level for an IMS system, system signon fails for earlier versions of IMS for online environments. All other jobs accessing the RECON data set fail DBRC initialization if the version of IMS being used is lower than the `MINVERS` level.

**Exit routine migration considerations**

Any user migrating from IMS 10 to IMS 12 must consider the changes to the SXPL made in IMS 11.

The user exit enhancements in IMS 11 introduce version 6 of the list (SXPLVER6). Exit routines that run in multiple versions of IMS must be sensitive to the version of the SXPL. The version number was SXPLVER5 in IMS 10.

The following exit routines can use the new fields:

- Build Security Environment exit routine (DFSBSEX0)
- Early Initialization exit routine (EINIT)
- Logger exit routine (DFSFLGX0)
- IMS CQS Event exit routine (ICQSEVNT)
- IMS CQS Structure Event exit routine (ICQSSTEV)
- Non-discardable Messages exit routine (DFSNDMX0)
- OTMA Destination Resolution exit routine (DFSYPRX0)
- OTMA Input/Output Edit exit routine (DFSYIOE0)
- OTMA User Data Formatting exit routine (DFSYDRU0)
- Partner Product exit routine (DFSPPUE0)
- Resource Access Security exit routine (DFSRA500)
- Restart exit (user-specified)
- Type 2 Automated Operator exit routine (DFSAOE00)
Exit routines that do not have exit point or interface changes and are supported in the EXITDEF parameter of the USER_EXITs section of the DFSDFxxx member can be enabled for the command functions introduced in IMS 11 by using the new version of the SXPL (SXPLVER6).

The IMS Connect exit parameter list (HWSEXPRM) is changed for IMS 11. You must reassemble and rebind the IMS Connect exit routines that use HWSEXPRM to pick up the changes.

**File system paths changes**

The file system paths changed in IMS 11. The changes affect IMS 10 users who are migrating to IMS 12.

The file system paths must be more efficient during the migration process. Table 8-5 shows how the file system paths differ between IMS 10 and IMS 12.

<table>
<thead>
<tr>
<th>DDNAME</th>
<th>IMS 10 path name</th>
<th>IMS 12 path name</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDFSIC4J</td>
<td>/usr/lpp/ims/ico101/IBM/</td>
<td>/usr/lpp/ims/ims12/ico/IBM/</td>
</tr>
<tr>
<td>SDFSJHFS</td>
<td>/usr/lpp/ims/imsjava10/IBM/</td>
<td>/usr/lpp/ims/ims12/imsjava/IBM/</td>
</tr>
<tr>
<td>SDFSJRAR</td>
<td>/usr/lpp/ims/imsjava10/rar/IBM/</td>
<td>/usr/lpp/ims/ims12/imsjava/rar/IBM/</td>
</tr>
<tr>
<td>SDFSJCIC</td>
<td>/usr/lpp/ims/imsjava10/cics/IBM/</td>
<td>/usr/lpp/ims/ims12/imsjava/classic/cics/IBM/</td>
</tr>
<tr>
<td>SDFSJCPI</td>
<td>/usr/lpp/ims/imsjava10/classic/IBM/</td>
<td>/usr/lpp/ims/ims12/imsjava/classic/IBM/</td>
</tr>
<tr>
<td>SDFSJTOL</td>
<td>/usr/lpp/ims/imsjava10/dlimodel/IBM/</td>
<td>/usr/lpp/ims/ims12/imsjava/classic/dlimodel/IBM/</td>
</tr>
<tr>
<td>SDFSJSAM</td>
<td>/usr/lpp/ims/imsjava10/samples/IBM</td>
<td>/usr/lpp/ims/ims12/imsjava/ivp/IBM/</td>
</tr>
<tr>
<td>SDFSJCP</td>
<td>/usr/lpp/ims/imsjava10/classic/ivp/IBM/</td>
<td>/usr/lpp/ims/ims12/imsjava/ivp/IBM/</td>
</tr>
<tr>
<td>SDFSJXQY</td>
<td>/usr/lpp/ims/imsjava10/IBM/</td>
<td>/usr/lpp/ims/imsjava10/IBM/</td>
</tr>
</tbody>
</table>

The following file system paths, which were created and used by previous releases of IMS, are no longer used in IMS 12. You can delete these obsolete file system paths after you delete the previous release from your system.

- /usr/lpp/imsico/
- /usr/lpp/IMSICO/
- /usr/lpp/ims/imsjava91/IBM/
- /usr/lpp/ims/ico91/IBM/
- /usr/lpp/ims/imsjava91/samples/IBM/
- /usr/lpp/ims/imsjava91/cics/IBM/
- /usr/lpp/ims/imsjava91/lib/IBM/
- /usr/lpp/ims/imsjava91/dlimodel/IBM/

For installation instructions, see *IMS Version 12 Program Directory*, GI10-8843.

**IMSplex migration considerations**

Migrating an IMSplex from one version of IMS to another is a complex process because of the many factors involved and the different configurations that are possible.

Although running a mixed IMSplex is possible, it is desirable to upgrade all your CSL components to IMS 12 if any control region is at IMS 12. Control regions are limited as to
which version of CQS they can connect to. For a listing of the rules, see 8.4.4, “Common Queue Server coexistence” on page 284.

If you are running multiple LPARs, migrate one LPAR at a time. If you are running multiple IMS systems on one LPAR, migrate one IMS at a time.

**IMS Connect migration considerations**

In previous releases, you specified multiple SSL ports even though only one active port at a time was supported. If a second port was opened, unpredictable results, including an abend, occurred, which was clearly undesirable.

In IMS 12, IMS Connect initialization fails if you specify multiple SSL ports. If you are migrating from IMS 10, you must modify the HWSCFGxx member to specify only one SSL port. An alternative option is to use application transparent-transaction layer security (AT-TLS).

Several new specifications are in the HWS, TCPIP, DATASTORE, MSC and RMTIMSCON statements of the HWSCFGxx configuration file. These specifications appear in the display output of commands such as **VIEWHWS**, **VIEWPORT**, and **VIEWDS**. Modify the automation programs and master terminal operator (MTO) documentation to recognize these new fields.

Automation programs that read the output of IMS Connect displays or query the HWSP1410W message must be aware of the new information and fields that have been added by the IMS 12 enhancements. Similarly, MTOs that issue IMS Connect commands should understand that additional information is provided.

Message HWSS0908W is issued if the old exits HWSIMSO0 and HWSIMSO1 continue to be specified in the IMS Connect configuration member. Those exits must be replaced with HWSSMPL0 and HWSSMPL1.

If WARNSOC and WARNINC are specified in the TCPIP HWSCFGxx statement, new messages are issued when the warning level is reached (HWSS0772W) and when the number of sockets falls below the warning level (HWSS0773I).

Because the HWSEXPRM macro has been expanded, IMS Connect exit routines that invoke the macro must be re-assembled. Remember also that the XIBDS (exit interface block data store entry) has also been expanded.

With the TCP/IP automatic reconnect capability, new code in the terminate port thread process of IMS Connect automatically issues an internal OPENPORT command on a timer basis. Operator commands or automation that are issued to ensure that IMS Connect reestablishes connectivity with a TCP/IP network are no longer needed.

To benefit from the Generated Client ID function, the IMS TM resource adapter must be replaced with the new version.

The new BPE trace capability is enabled only when the old function is disabled by removing or commenting out the HWSRCORD DD statement in the IMS Connect startup procedure. After the new function has been enabled, new RCTR entries in the BPE external trace data sets will be introduced as variable length trace entries.

IMS 12 adds the following records for the sending or receiving TCP/IP and XCF:

- ICONTR – TCP/IP Receive
- ICONTS – TCP/IP Send
- ICONIR – IMS OTMA Receive
- ICONIS – IMS OTMA Send
These records are only written to the BPE External trace because the new trace points can generate a significant amount of data. For more information, see 6.3.1, “IMS Connect Recorder Trace” on page 188.

**RACF mixed-case password support**
The default values for `PSWDC` and `PSWDMC` are different in IMS 12 than they were in IMS 10. The new default values are `PSWDC=R` and `PSWDMC=R` (use the Resource Access Control Facility (RACF) specification).

IMS 10 can process mixed-case passwords, but to enable this function, you must specify `PSWDC=M` for IMS and `PSWDMC=Y` for IMS Connect. The default values for IMS 10 are `PSWDC=U` (uppercase) and `PSWDMC=N` (not mixed case).

The `PSWDC` and `PSWDMC` parameters are enhanced in IMS 12 with the “R” specification, which means that IMS and IMS Connect should handle passwords in the same manner as specified in RACF.

**Remote Site Recovery migration**
The migration of systems using Remote Site Recovery (RSR) is similar to migrations for previous releases.

IMS 12 tracking systems can process logs produced by lower releases.

The IMS 12 isolated log sender (ILS) function of the Transport Manager System (TMS) can process logs created by earlier releases, with the following exceptions:
- IMS 11 and IMS 10 tracking systems cannot accept logs produced by IMS 12.
- IMS 11 and IMS 10 ILSs cannot accept logs produced by IMS 12.

Although you can migrate all of the RSR components at the same time, you are more likely to migrate them in stages. The tracking system must be migrated before or at the same time as the ILS at the active site. The ILS at the active site must be migrated before or at the same time as the active IMS system.

The RECONs must be upgraded to IMS 12 before the systems that use them are migrated to IMS 12.

**Migration steps**
Perform the following migration steps:
1. Upgrade the RSR tracking system RECONs to IMS 12.
2. Migrate the RSR tracking system to IMS 12.
3. Upgrade the active system RECONs to IMS 12.
4. Migrate the active TMS running the ILS to IMS 12.
5. Migrate the active IMS to IMS 12.

**Serviceability enhancements**
To benefit from serviceability enhancements when you migrate to IMS 12, you might have to increase storage or modify automated tools or procedures.

Because the number of address spaces included in the system dump is limited by the amount of storage specified by the `MAXSPACE` parameter of the z/OS CHNGDUMP command, you might have to increase the amount of storage to accommodate the additional address spaces. However, IMS does not exceed the `MAXSPACE` value.
You might have to modify your automated operations tools or procedures if they are dependent on the format of the DFS064I or DFS065 messages.

**Syntax Checker migration considerations**

Syntax Checker assists with IMS release-to-release migrations by providing the ability to convert supported PROCLIB members from one release to the other. When using Syntax Checker to check parameters for earlier releases of IMS, you must verify that the correct release number is displayed.

Syntax Checker is a stand-alone, offline component, and it can fall back by installing the previous release of Syntax Checker. Always maintain a copy of your previous-version PROCLIB members to enable fallback.
IMS Enterprise Suite V2.1

The IBM Information Management System (IMS) Enterprise Suite, part of the IMS SOA Integration Suite, is a set of components that support open integration technologies to enable new application development and extend access to IMS transactions and data. The IMS Enterprise Suite provides standard interfaces, simplifies IMS metadata generation, and enables IMS business event data and monitoring. The IMS Enterprise Suite also eases and expands IMS development (including Java and XML), administration, and access. Graphical user interfaces (GUIs) and standards-based programming models are provided through tooling support from the IBM WebSphere and Rational® product families.

For more information, see the IMS Enterprise Suite website at: https://www14.software.ibm.com/webapp/iwm/web/preLogin.do?source=swg-imsentersuite

This chapter explains the functions (components) that are part of the IMS Enterprise Suite V2.1. It highlights the functionality and enhancements of each component from Enterprise Suite V1.1 to V2.1.

This chapter includes the following sections:

- Enterprise Suite V2.1 contents
- Enhancements for Enterprise Suite V2.1
- IMS Enterprise Suite Connect APIs for C
- IMS Enterprise Suite Connect APIs for Java
- IMS Enterprise Suite DLIModel utility plug-in IMS Enterprise Suite V2.1
- IMS Enterprise Suite Explorer for Development
- IMS Enterprise Suite SOAP Gateway 2.1
- IMS Enterprise Suite Java Message Service API
9.1 Enterprise Suite V2.1 contents

The IMS Enterprise Suite components are available to IMS 11 and IMS 12 clients at no additional cost. Some components are available on both z/OS and distributed systems, and some are only available on distributed systems.

The following components are part of the Enterprise Suite V2.1:

- **Base Services** includes sample jobs to use for installation of z/OS-based components and the IBM 31-bit software development kit (SDK) for z/OS, Java 2 Technology Edition, Version 6 (Java Development Kit (JDK) 6.0).

- **IMS Enterprise Suite Connect APIs for C (in IMS 10)**
  
  This component provides simple interfaces for developing custom IMS Connect TCP/IP client applications that are written in C/C++. The application programming interfaces (APIs) can be used to develop custom IMS Connect TCP/IP client applications in Windows environments.

- **IMS Enterprise Suite Connect APIs for Java (enhanced in IMS12)**
  
  This component provides simple interfaces for developing custom IMS Connect TCP/IP client applications that are written in Java. It provides a simple way to describe TCP/IP socket connections, interaction protocols, message headers, and data through the concepts of reusable profiles. More granular lower-level calls are provided for more granular controls. The APIs can be used to develop custom IMS Connect TCP/IP client applications in Windows, and z/OS environments.

- **IMS Enterprise Suite DLIModel utility plug-in (in IMS 11)**
  
  This component translates IMS source files into reliable, application-independent metadata that can be used for Java application development. The utility can be used to generate XML schemas or graphical Unified Modeling Language (UML) models for IMS databases. You can incorporate additional program communication block (PCB), segment, and field information, and generate IMS database web service artifacts with the utility.

- **IMS Enterprise Suite Explorer for Development (new in IMS 12)**
  
  This component simplifies IMS application development tasks by displaying and enabling editing of IMS databases, segments, fields, and more, from an industry-standard integrated development environment (IDE). The IMS Enterprise Suite Explorer for Development is an Eclipse-based graphical tool that enables IMS application developers and database architects and developers to undertake the following activities:

  - Perform common and essential tasks in an end-to-end application development lifecycle
  - Simplify the development and visualization of database description (DBD) and program specification block (PSB) resources definitions
  - Import COBOL and PL/I data structures to an IMS Database
  - Share with IBM Rational Developer for System z to generate PSB source, and to import and export DBD and PSB source from or to a z/OS remote system
  - Use the IMS Universal drivers, offering a relational view of IMS data and offering new function, such as graphical assistance to build SQL statements

- **IMS Enterprise Suite SOAP Gateway V2.1 (enhanced in IMS 12)**
  
  This component enables IMS applications to interoperate outside of the IMS environment through the SOAP protocol to provide and request services that are independent of platform, environment, application language, or programming model. IMS applications can become web service providers or consumers in a service-oriented business environment.
SOAP Gateway also enables IMS applications to emit business events data to business event processing engines such as IBM WebSphere Business Events and IBM WebSphere Business Monitor.

- **IMS Enterprise Suite Java Message Service (JMS) API (in IMS 11)**
  This component issues synchronous callout requests to external services from within a Java message processing (JMP) or Java batch processing (JBP).

- **MFS Web Enablement**
  This component helps you to modernize your existing MFS-based IMS business logic by converting character-based interface-based applications into web-based applications.

### 9.1.1 Acquiring and installing Enterprise Suite V2.1

As shown in Table 9-2, some components are composed of a z/OS part and a distributed part. The distribution for Enterprise Suite V2.1 is different from Enterprise Suite V1.1.

#### Downloading and installing on distributed systems

Download the parts for the distributed systems individually from the IMS Enterprise Suite site at:


Depending on the component, some parts require the IBM Rational Application Installation Manager (at a minimum level of 1.4.4) to be installed in a new, or added to an existing, Rational Workbench package.

#### Downloading and installing on z/OS

For z/OS, install IMS Enterprise Suite components by using the standard SMP/E installation process.

IBM IMS Enterprise Suite for z/OS V2.01.00 consists of the following function modification IDs (FMID):

- **HAHF210 (Base Services)**
- **JAHF201 (SOAP Gateway)**
- **JAHF202 (JMS API)**
- **JAHF203 (Connect API Java)**

Table 9-1 lists the IMS environments that each IBM IMS Enterprise Suite for z/OS V2.01.00 FMID supports.

**Table 9-1  IMS environments**

<table>
<thead>
<tr>
<th>FMID and description</th>
<th>DB Batch</th>
<th>DBCTL</th>
<th>DB/DC</th>
<th>DCCTL</th>
</tr>
</thead>
<tbody>
<tr>
<td>HAHF210 (Base Services)</td>
<td>A&lt;sup&gt;a&lt;/sup&gt;</td>
<td>A</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>JAHF201 (SOAP Gateway)</td>
<td>N&lt;sup&gt;b&lt;/sup&gt;</td>
<td>N</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>JAHF202 (JMS API)</td>
<td>N</td>
<td>A</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>JAHF203 (Connect API Java)</td>
<td>N</td>
<td>N</td>
<td>A</td>
<td>A</td>
</tr>
</tbody>
</table>

<sup>a</sup> The FMID is applicable to this IMS configuration.

<sup>b</sup> The FMID is not applicable to this IMS configuration.
9.2 Enhancements for Enterprise Suite V2.1

Table 9-2 lists the differences between Enterprise Suite V1.1 and V2.1 components for the z/OS and distributed environment.

Table 9-2 Differences between Enterprise Suite V1.1 and V2.1

<table>
<thead>
<tr>
<th>Component name</th>
<th>z/OS</th>
<th>Distributed</th>
<th>Major changes in Enterprise Suite V2.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connect API for C</td>
<td>X</td>
<td></td>
<td>▶ No new feature</td>
</tr>
<tr>
<td>Connect API for Java</td>
<td>X</td>
<td>X</td>
<td>▶ Synchronous callout support</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>▶ Support for type-2 commands</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>▶ Other minor enhancements</td>
</tr>
<tr>
<td>DLIModel utility plug-in</td>
<td>X</td>
<td></td>
<td>▶ No change from Version 1.1</td>
</tr>
<tr>
<td>Enterprise Suite Explorer for Development</td>
<td>X</td>
<td></td>
<td>▶ New with IMS 12</td>
</tr>
<tr>
<td>SOAP Gateway</td>
<td>X</td>
<td>X</td>
<td>▶ Java separation</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>▶ WS-Security enhancements for the web service provider scenario</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>▶ SAML 1.1 signed assertions</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>▶ SAML 2.0 unsigned assertions</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>▶ Rational Developer for System z</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>▶ V8.0.3 enhancements for SOAP Gateway</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>▶ Removal of SOAP Gateway deployment utility</td>
</tr>
<tr>
<td>JMS API</td>
<td></td>
<td></td>
<td>▶ No change from Version 1.1</td>
</tr>
</tbody>
</table>

The following sections describe each of the Enterprise Suite V2.1 components. Because Base Services supports the other components, it is not addressed.

▶ IMS Enterprise Suite Connect APIs for C
▶ IMS Enterprise Suite Connect APIs for Java
▶ IMS Enterprise Suite DLIModel utility plug-in IMS Enterprise Suite V2.1
▶ IMS Enterprise Suite Explorer for Development
▶ IMS Enterprise Suite SOAP Gateway 2.1
▶ IMS Enterprise Suite Java Message Service API

For more information, see IBM Information Management Software for z/OS Solutions Information Center at the following address, and search for JMS API (in “IMS Enterprise Suite Version 1.1”):

http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/index.jsp

9.3 IMS Enterprise Suite Connect APIs for C

The IMS Enterprise Suite Component APIs for C were introduced in IMS Enterprise Suite V1.1, which became available for IMS 11 and IMS 10. The IMS Enterprise Suite Connect API for C provides simple interfaces for developing custom IMS Connect TCP/IP client applications that are written in C/C++. This component is only available for the Windows platform, supported for Windows XP.
9.3.1 Overview of API for C


The Connect API for C simplifies the process of developing C and C++ applications that interact with IMS through IMS Connect in the following ways:

- It manages the communication between the C application and IMS Connect. The client application provides property values that describe the type of connection to establish with IMS Connect. The IMS Enterprise Suite API for C establishes a connection for the applications and maintains that connection for as long as it is needed.
- It helps you to set the interaction specification, which determines the details of the exchange with IMS. The parameter values in the interaction specification determine the fields that are set by the API in the IMS request message (IRM) header of messages that are sent to IMS Connect.
- It provides functions that encapsulate the creation of an input request message to be sent to IMS Connect and the retrieval of the resulting response from IMS Connect. The creation of the request message is based on properties that are specified in the application or loaded from a reusable properties file. The properties determine the type of interaction to be executed.

By completing these steps on behalf of the application, much of the complexity of the IMS Connect headers and protocols can be shielded.

You can use the IMS Enterprise Suite Connect API for C to drive IMS transactions, Open Transaction Manager Access (OTMA)-supported IMS commands, and IMS Connect-supported commands (such as PING and Resource Access Control Facility (RACF) Password Change).

The IMS Enterprise Suite Connect API for C/C++ supports the HWSSMPL0 and HWSSMPL1 IMS Connect user message exits. The default user exit is HWSSMPL1. In addition to the message segment as expected by the IMS program or the command processor, the input message prepared by the API contains an IMS Connect header of TYPE 2 prefixed. This is also called the IRM header.

The IMS Enterprise Suite Connect API for C supports all existing user-supplied IMS Connect client application functions except for the following functions:

- Two-phase commit
- Unicode
- Synchronous callout
- Secure Sockets Layer (SSL) support

9.3.2 API for C reference for data types and functions

The C API specification for the Connect API for C includes reference information about the data types and functions of the Connect API for C:

- Data types of the Connect API for C
  - APIStatus, ConnectionAttribute, messageSegment, outputData, and TMInteractionAttribute
- Connect API for C
  - loadConnection, connectIMS, loadInteraction, is .... examples (isConnected, isReturnMfsModname, isAckNakNeeded..), set ... examples (setHostName,
setPortNumber, setCommitMode ...), get ... examples (getImsConnectReasonCode, getMfsModname, ...), initializeAPI, initializeConnection, initializeInteraction, execute, disconnectIMS, and cleanUp ... examples (cleanUp, cleanUpConnectionMemoryPool ...)

- Tracing with the Connect API for C
  startLogger and stopLogger
- Connect API for C messages

9.3.3 Reference material

For reference information about the IMS Enterprise Suite Connect APIs for C, click the Connect API links on the IMS Connect Function and IMS Enterprise Suite Connect APIs page at:

http://www.ibm.com/software/data/ims/connect/

All materials on this site are downloadable and installable. The downloadable material also contains two C and two C++ sample programs. For compiling and linkediting client programs, a C/C++ workbench is required, such as the Microsoft Visual Studio 2008 Professional Edition.

For more information, go to the IBM Information Management Software for z/OS Solutions Information Center at the following address, and search for Connect API for C:

http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/index.jsp

9.3.4 Prerequisites for Connect API for C

Before you download and install the Connect API for C, see the supported platforms and software requirements. You can find them in the IBM Information Management Software for z/OS Solutions Information Center at the following address, and search for Installing the Connect API for Java:

http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/index.jsp

9.4 IMS Enterprise Suite Connect APIs for Java

This component was introduced in Enterprise Suite V1.1, which became available for IMS 11 and IMS 10, but has been enhanced in IMS Enterprise Suite V2.1. The IMS Enterprise Suite Connect API for Java provides simple interfaces for developing custom IMS Connect TCP/IP client applications that are written in Java. This component is available for both the Windows and the z/OS platform.

9.4.1 Overview of API for Java

The Connect API for Java includes the following key features:

- It opens connections to IMS Connect on behalf of the client application.
- It provides client applications with programmatic control of connections.
- It assembles messages to send to IMS Connect, including the IRM header.
- It manages the IMS Connect message protocol by sending and receiving the appropriate messages for interactions with IMS Connect.
The Connect API for Java manages the communication between the Java application and IMS Connect. The client application provides property values that describe the type of connection to establish with IMS Connect. The Connect API for Java establishes a connection for the application and maintains that connection for as long as it is needed.

You can use the Connect API for Java to drive IMS transactions, OTMA-supported IMS commands, Common Service Layer (CSL) Operations Manager (OM)-supported IMS type-2 commands, and IMS Connect-supported commands (such as PING and RACF Password Change) from your Java client application. The Connect API for Java supports SSL for securing TCP/IP communications between client applications and IMS Connect.

9.4.2 API for Java enhancements in Enterprise Suite V2.1

The IMS Enterprise Suite 2.1 Connect API for Java supports the new items described in the following sections.

**Synchronous callout interaction support**

The Connect API for Java now supports synchronous callout interactions with IMS Connect. The existing TmInteraction interface is enhanced with new methods and property values that you can use to receive a synchronous callout message from IMS, send an ACK or NAK response to acknowledge receipt of the message, construct a response message or response error message, and then send the response to IMS. The Connect API can issue a `resume Tpipe` and respond to the synchronous callout emitted from dependent region programs in IMS.

Callout interactions are initiated from an IMS application program rather than from the Connect API for Java client application. In a synchronous callout interaction, the IMS application program that sends the callout request remains scheduled in the IMS dependent region while the request is processed by the Connect API for Java client application. Figure 9-1 illustrates how it works.
For a complete sample, see the online documentation in the IBM Information Management Software for z/OS Solutions Information Center at:

http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/index.jsp

Prerequisites for Connect API for Java
Check the supported platforms and software requirements before you download and install the Connect API for Java. For more information, see the IBM Information Management Software for z/OS Solutions Information Center at the following address, and search for Installing the Connect API for Java:

http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/index.jsp

See also OTMA destination descriptors in IMS Version 12 Communications and Connections, SC19-3012.

IMS type-2 commands
The Connect API for Java now supports IMS and IMS Connect type-2 command input. The Connect API for Java can return the response data as either a raw XML document or as a parsed response object.

When communicating with IMS 12 and later, client applications can submit IMS type-2 commands to the CSL OM with the Connect API for Java. The Connect API for Java can deliver the response message as either an XML document in a string array or as name and value pairs in a Java Properties array.

The IMS type-2 command interface is provided by the CSL OM component of IMS. Type-2 commands require an active CSL OM in the target IMSplex and return output in XML format.

The Connect API for Java can send IMS type-2 command input in a standard InputMessage object. The InputMessage is parsed and sent to IMS, and the response message is returned to the Connect API for Java client as an XML instance document that conforms to the imsout.dtd XML definition document. Your client application can retrieve the XML directly, or use a set of method calls provided by the Connect API for Java to retrieve the values of individual fields from the response message.

The samples.com.myCompany.myProjects.Type2Command.Type2CommandSample.java program, which is available with the download of the ImsESConnectApiJavaV2R1Samples.jar file, shows how to use this facility.

CM0 ACK NOWAIT transaction support
The Connect API for Java includes CM0 ACK NOWAIT transaction support when communicating with an IMS Connect Version 12 host.

Support for CM0 send-and-receive messages that do not receive a response
In IMS 12, a client application that sends a CM0 send-and-receive message with the TMAMHRSP flag in the state data prefix that does not receive a response from the IMS application program is alerted with a DFS2082 message. This message alerts your Connect API for Java client application when a CM0 send-and-receive request message does not receive a response.

Support for IMS 12 RACF return codes
In IMS 12, the IMS Connect request status message (RSM) contains a 2-byte RACF return code when a RACF security failure occurs during message processing. The RACF return
code indicates what caused the security failure, and it useful for diagnosing security configuration problems. The Connect API for Java provides the `getRacfReturnCode` and `getRacfReturnCodeString` methods to retrieve the error information.

### 9.4.3 API for Java classes and methods

The runtime implementation of Connect API for Java is provided by the `com.ibm.ims.connect` package in the `imsESConnectApiJavaV2R1.jar` file (Figure 9-2). The `com.ibm.ims.connect` package contains classes and interfaces for the following abstractions:

- Connections to IMS Connect
- Constants for connection and interaction property values
- Error messages
- Interactions with IMS Connect and IMS

#### Interface Summary

<table>
<thead>
<tr>
<th>Interface Summary</th>
<th>Modifier and Type</th>
<th>Interface and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApiProperties</td>
<td></td>
<td>Constructs for setting IMS Connect API connection and interaction properties.</td>
</tr>
<tr>
<td>Connection</td>
<td></td>
<td>A TCP/IP socket connection for communicating with IMS Connect.</td>
</tr>
<tr>
<td>InputMessage</td>
<td></td>
<td>Contains the data in the input message to be sent to IMS Connect.</td>
</tr>
<tr>
<td>OutputMessage</td>
<td></td>
<td>Contains the data in the output message received from IMS Connect.</td>
</tr>
<tr>
<td>Transaction</td>
<td></td>
<td>Configures interaction properties and executes an interaction with IMS Connect in an IMS transaction.</td>
</tr>
</tbody>
</table>

#### Class Summary

<table>
<thead>
<tr>
<th>Class Summary</th>
<th>Modifier and Type</th>
<th>Class and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApiLoggingConfiguration</td>
<td></td>
<td>A helper class to configure tracing for IMS Connect API applications.</td>
</tr>
<tr>
<td>ConnectionAttributes</td>
<td></td>
<td>This class represents the configuration of a connection.</td>
</tr>
<tr>
<td>ConnectionFactory</td>
<td></td>
<td>Factory class for creating connections to IMS Connect.</td>
</tr>
<tr>
<td>TransactionAttributes</td>
<td></td>
<td>This class represents the properties that are used to perform an interaction with IMS Connect for running an IMS transaction.</td>
</tr>
</tbody>
</table>

Figure 9-2  The `imsESConnectApiJavaV2R1.jar` file

### 9.4.4 Reference material

For reference information about the IMS Enterprise Suite Connect APIs for Java, click the Connect API links at the IMS Connect Function and IMS Enterprise Suite Connect APIs page at:

http://www.ibm.com/software/data/ims/connect/

To install the Connect API for Java, complete the following tasks:

- Obtain the Connect API for Java component. You can obtain the component by ordering the IMS Enterprise Suite product, or by downloading the component from the IMS Enterprise Suite download website.
- For the z/OS platform, order a Custom-Built Product Delivery Offering (CBPDO) for the IMS Enterprise Suite product through ShopzSeries.
- For other platforms, download the latest service level for the Connect API for Java from the Connect API for Java download site.
- Store the Connect API for Java archive (JAR) file in a directory that is in your class path.
The Connect API for Java is supported on the z/OS and Windows systems. The following software versions are supported:

- z/OS V1.11 or later for IMS 12 and IMS 11
- Microsoft Windows XP (SP2)

The following software is required:

- JDK Version 6.0 (JDK 6.0) or later for the development environment
- Java Runtime Environment (JRE) Version 6.0 or later for the runtime environment

**JRE:** The JRE must be installed with the extended character set library (installed in the `lib\charsets.jar` file) that includes support for Cp037 and Cp1047 EBCDIC encoding. This library is included with the Support for Additional Languages option in the J2SE installer.

For more information, see IBM Information Management Software for z/OS Solutions Information Center at the following address, and search for *Connect API for Java*:

http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/index.jsp

### 9.5 IMS Enterprise Suite DLIModel utility plug-in IMS Enterprise Suite V2.1

The IMS Enterprise Suite DLIModel utility plug-in translates IMS source files into reliable, application-independent metadata that can be used for Java application development in an Eclipse-based environment, eliminating the need for hand-coded control statements.

The DLIModel utility plug-in can be used on both Windows and Linux systems. For a Java application to access an IMS database, this plug-in needs information about that database. This database information is in IMS source files such as program specification blocks (PSBs) and database descriptions (DBDs).

To access this information, you must first convert it into a form that you can use in the Java application, a subclass of the `com.ibm.ims.db.DLIDatabaseView` class that is called the IMS Java metadata class. The DLIModel utility generates this metadata from IMS source files.

In addition to creating metadata, the DLIModel utility plug-in generates a graphical UML model that illustrates the IMS database structure in an interactive tree model:

- It generates annotated XML schemas of IMS databases, which are used to retrieve XML data from or store XML data in IMS databases.
- It incorporates additional field information from COBOL or PL/I copybooks.
- It incorporates additional PCB, segment, and field information, or overrides existing information through a graphical view of the PCB.
- It generates a DLIModel database report, which assists Java application programmers in developing applications based on existing IMS database structures.
- It generates an optional DLIModel trace log.
- It provides a configuration editor as a launching point for generating IMS database web services artifacts.
- It generates XMI descriptions of the PSB and its databases.

The DLIModel utility plug-in can process:
Chapter 9. IMS Enterprise Suite V2.1

- All database organizations except MSDB, HSAM, and SHSAM
- All types and implementations of logical relationships
- Secondary indexes, except for shared secondary indexes (not for data entry database (DEDB))

DLIModel is a helpful tool for visualizing and documenting all details about DBDs and PSBs sources. DLIModel is also required to produce the com.ibm.ims.db.DLIDatabaseView class for accessing IMS databases from Java applications on remote and local locations by using the type 4 and type-2 drivers. Both traditional DLI/SSA and Java Database Connectivity (JDBC) access can be used.

The DLIModel is not changed in this Enterprise Suite V 2.1.

**JRE:** The JRE must be installed with the extended character set library (installed in the lib\charsets.jar file) that includes support for Cp037 and Cp1047 EBCDIC encoding. This library is included with the Support for Additional Languages option in the Java 2 Platform, Standard Edition (J2SE) installer.

For more information, see IBM Information Management Software for z/OS Solutions Information Center at the following address, and search for *IMS Enterprise Suite DLIModel utility plug-in*:

http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/index.jsp

### 9.6 IMS Enterprise Suite Explorer for Development

The IMS Enterprise Suite Explorer (IMS Explorer) is an Eclipse-based graphical tool that simplifies IMS application development tasks such as updating IMS database and program definitions, and using standard SQL to manipulate IMS data. You can use the IMS Explorer graphical editors to import, visualize, and edit IMS database and program definitions. You can also use the IMS Explorer to easily access and manipulate data stored in IMS by using standard SQL.

#### 9.6.1 Building a project with IMS Explorer

After you install the component and its Eclipse environment, you must create a new “IMS Explorer Project”:

1. Select **File → New → IMS Explorer Project** (Figure 9-3). The Project Explorer tab opens (Figure 9-6 on page 335). Alternatively, select **File → New → Project**.

![Figure 9-3 New IMS Explorer Project](image_url)
2. In the New Project window (Figure 9-4), select **IMS → IMS Explorer Project**. Then click **Next**.

![Figure 9-4 New Project Wizard](image)

3. In the New IMS Explorer Project window (Figure 9-5), choose a project name. When you click **Next**, you can import PSBs and DBDs from a local or remote file system, which we do later. The reason for postponing this task is that, in an existing project, the import requires the selection of the adequate “input source”. You must make this selection explicitly. In this case, we click **Finish**.

![Figure 9-5 Name of the project](image)
The project is built now. Figure 9-6 shows how it looks.

![Figure 9-6  ESEExplorer project](image)

### 9.6.2 Importing the resources for DBD and PSB

Many Java classes, all which start with `com.ibm.ims`, have been included in the project. They are packaged in a JAR file, called `imsjava.jar`. The DBD and PSB folders are still empty, because we skipped the resource import.

To import the resources, complete these steps:

1. Right-click **ESEExplorer** and select **Import**.
2. In the Import window, select **IMS Resources**, and then click **Next**.
3. In the Import IMS Resources window, enter the project name, and then click **Next**.
Figure 9-7 illustrates the first three steps in this task.

4. In the Select an Import Source panel, select **IMS Resources**, and then click **Next**.
5. In the next window (Figure 9-8), select whether to input from a local or remote source. We select **Local file system** in this example. Then click **Next**.

![Figure 9-7 Import with Project name selection](image1)

![Figure 9-8 Local or remote import source selection](image2)
6. Because Local file system was selected in the previous step, select the resources from the file system. In this case, we import from a local folder in which we have PSBs and the DBDs, referenced by the PCBs in this PSB (Figure 9-9). We selected the **AUTO dealer application**, with **PSB AUTPSB11**. All required DBDs are in the same folder and preselected. Click Finish.

![Figure 9-9  PSB selected and DBD found in same folder and preselected](image)

7. If you receive a warning that some fields cannot be used with JDBC/SQL access, which can be solved later (Figure 9-10), click **OK**.

![Figure 9-10  Problems to be solved for SQL access](image)
The source code for PSBs and DBDs for the Autodealer program (AUTPSB11) are loaded in the Explorer project. All sources are now available in three copies (Figure 9-11):

- Original imported source code
- Generated or transformed source code
- XML representation of the source (both DBD and PSB)

XML representation of the source is used as meta information. This information can be looked at in two ways. The representation is different for DBDs and PSBs.

**DBDs**

The DBD metacode is available in XML format as shown in Example 9-1, which you can view by using a text editor.

**Example 9-1 Excerpt of the DBD metacode in XML format for DBD “AUTODB”**

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:dbd xmlns:ns2="http://www.ibm.com/ims/DBD" dbdName="AUTODB">
  <access dbType="HDAM">
    <hdam datxexit="N" password="N" osAccess="OSAM">
      <rmName name="DFSHDC40">
        <subOptions bytes="200" maxRBN="5" anchorPoints="1"/>
      </rmName>
      <dataSetContainer>
        <dataSet searchA="0" scan="3" ddname="DFSDLR">
          <block size="-1"/>
          <size size="-1"/>
          <frspc fspf="0" fbff="0"/>
        </dataSet>
      </dataSetContainer>
    </hdam>
  </access>
  <segment imsName="DEALER">
    <field imsDatatype="C" seqType="U" imsName="DLRNO">
      <startPos>1</startPos>
      <bytes>4</bytes>
      <marshaller>
        <typeConverter>CHAR</typeConverter>
      </marshaller>
    </field>
  </segment>
</ns2:dbd>
```
The original DBD source has been interpreted and enriched by the IMS source reader, and represented in XML. This is the base for editing by the Explorer.

The DBD metacode is also available in graphical format, with the IMS DBD editor (Figure 9-12).
Several options are available for using the DBD editor:

- Hovering over a segment field to view the basic characteristics through hover text (Figure 9-13).

- Right-clicking a segment, for example ORDER, and selecting an option from the menu (Figure 9-14).
You can enrich the segment information with a COBOL copybook or a PLI include. Managing the fields gives you a list of the fields in the segments (Figure 9-15). In the Manage Fields window, you can add fields, remove fields, or edit fields.

The Edit Field window (Figure 9-16) shows an example of editing the properties for a segment field.
PSBs

The PSB metacode is available in XML format as shown in Example 9-2, which you view by using a text editor.

Example 9-2  Excerpt of the DBD metacode in XML format for PSB “AUTPSB11”

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns2:psb xmlns:ns2="http://www.ibm.com/ims/PSB" compat="N" lang="JAVA" psbName="AUTPSB11">
  <dbPCB copies="1" procopt="AP" keyLength="100" dbdName="AUTOLDB" name="AUTOLPCB">
    <label>AUTOLPCB</label>
    <senseg name="DEALER">
      <indices/>
      <senseg name="MODEL">
        <indices/>
        <senseg name="ORDER">
          <indices/>
        </senseg>
        <senseg name="SALES">
          <indices/>
        </senseg>
        <senseg name="STOCK">
          <indices/>
          <senseg name="STOCSALE">
            <indices/>
          </senseg>
        </senseg>
      </senseg>
      <senseg name="SALESPER">
        <indices/>
        <senseg name="SALESINF">
          <indices/>
        </senseg>
        <senseg name="EMPLINFO">
          <indices/>
        </senseg>
      </senseg>
    </senseg>
  </dbPCB>
```

......
You can also view the PSB metacode by using the IMS PSB editor. The first part of the PSB editor (Figure 9-17) shows sections related to the IO (alternate) PCB, DB PCBS, and GSAM PCB (not shown).

Figure 9-17  Alternate and DB PCB sections

The last part of the editor shows general characteristics of the PSB (Figure 9-18).

Figure 9-18  General PSB section
You can make several adjustments on the PSB in the editing panel. This panel includes the following IO/DB PCB sections:

- New PCBs can be added.
- Existing PCBs can be removed.
- Details of existing PCBs can be edited.

As an example, we edit one PCB in the Full-function or Fast-Path database section (Figure 9-19).

![Full-function or Fast Path database PCB statement](image1)

**Figure 9-19** Selecting a DBPCB for detailed editing

After selecting the PCB, we have several options, including two edit options (Figure 9-19). We start first with the “Edit” option. Then the Edit PCB Statement window (Figure 9-20) opens. You recognize the options that otherwise are entered with the PCB macro during the PSBGEN. The PCB shown in Figure 9-20 represents access by using a secondary index.

![Edit PCB Statement window](image2)

**Figure 9-20** Edit PCB Statement window
After returning to the previous panel, we can select **Edit data sensitivity**. Figure 9-21 shows the sensitivity by segment and by field. This sensitivity can be changed. By using the symbols in the upper-right corner, you can switch between editing mode and read-only mode.

![Figure 9-21  Sensitivity panel](image)

By hovering over a segment name or even over a field, hover text is displayed, indicating the details of the sensitivity (Figure 9-22).

![Figure 9-22  Field sensitivity](image)
With the PCB macro from the PSBGEN (Example 9-3), you can verify the graphical representation.

**Example 9-3 PCB macro in PSBGEN corresponding to Figure 9-21**

<table>
<thead>
<tr>
<th>PCB</th>
<th>DB NAME</th>
<th>PROCOPT</th>
<th>KEYLEN</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTO1 PCB</td>
<td>AUTOLDB</td>
<td>AP</td>
<td>100</td>
<td>00170000</td>
</tr>
<tr>
<td></td>
<td>DEALER</td>
<td></td>
<td></td>
<td>00180000</td>
</tr>
<tr>
<td></td>
<td>MODEL</td>
<td>DEALER</td>
<td></td>
<td>00190000</td>
</tr>
<tr>
<td></td>
<td>ORDER</td>
<td>MODEL</td>
<td></td>
<td>00200000</td>
</tr>
<tr>
<td></td>
<td>SALES</td>
<td>MODEL</td>
<td></td>
<td>00210000</td>
</tr>
<tr>
<td></td>
<td>STOCK</td>
<td>MODEL</td>
<td></td>
<td>00220000</td>
</tr>
<tr>
<td></td>
<td>STOCSALE</td>
<td>STOCK</td>
<td></td>
<td>00230000</td>
</tr>
<tr>
<td></td>
<td>SALESPER</td>
<td>DEALER</td>
<td></td>
<td>00240000</td>
</tr>
<tr>
<td></td>
<td>SALESINF</td>
<td>SALESPER</td>
<td></td>
<td>00250000</td>
</tr>
<tr>
<td></td>
<td>EMLINFO</td>
<td>SALESPER</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### 9.6.3 Input for IMS Explorer

Chapter 8, “Installation and migration considerations” on page 279, explains how input for IMS Explorer was taken from local PSB and DBD sources. The explorer can also accept input in the following ways:

- Import of a DLIModel Utility project

  An existing DLIModel utility project can be the input for the explorer, by directly referencing its location (Figure 9-23).

![Figure 9-23 Import from DLIModel project](image)
Remote import

The sources of PSBs and DBDs can be imported from a remote location by using FTP (Figure 9-24).

![Figure 9-24 Import from z/OS through FTP](image)

9.7 IMS Enterprise Suite SOAP Gateway 2.1

SOAP Gateway enables IMS applications to interoperate outside of the IMS environment through the SOAP to provide and request services that are independent of platform, environment, application language, or programming model. SOAP Gateway helps you transform your IMS applications to either web service providers or consumers (Figure 9-25).

![Figure 9-25 Web services call-in and call-out](image)
When IMS applications are enabled as web service providers, different types of client applications such as Microsoft .NET, Java, and third-party applications can submit SOAP requests into IMS to drive the business logic of the COBOL applications.

When IMS applications are enabled as web service consumers, they can call out to any external web services and receive the response in the same or a different transaction.

SOAP Gateway can be used on z/OS, Linux for System z, and Windows systems.

This section addresses the following areas:

- SOAP Gateway components
- Web services security
- SOAP Gateway V2.1 security implementation
- Security setup for Enterprise Suite V2.1
- SOAP Gateway Server
- SOAP Gateway management utility
- Implementing a call-in web service for IMS
- SOAP Gateway Administrative Console

### 9.7.1 SOAP Gateway components

SOAP Gateway includes several components. First, it includes a server component that processes web service requests between IMS applications and external applications or web services. It also includes a utility for deploying web services and managing server properties and an administrative console for verifying installation and deployed web services.

- **SOAP Gateway server**
  
  The SOAP Gateway server acts as the gateway between external web services and IMS applications. It serves as a web service server where IMS applications are enabled as web services. It serves a web service consumer when it forwards IMS application callout requests or business event data to external web services or event processing services.

- **SOAP Gateway management utility**
  
  The SOAP Gateway management utility provides a command line or batch interface for configuring server properties, managing the server run time and working with web service artifacts.

- **SOAP Gateway administrative console**
  
  The SOAP Gateway administrative console lists the deployed web services when the server is started. Each item in the list is a link to the Web Services Description Language (WSDL) file for the web service. It also replaces the Deploy utility of IMS Enterprise Suite V1.1.

The following items are new in version 2.1:

- When IMS Enterprise Suite Version 2.1 SOAP Gateway is the service provider, it adds support for Security Assertion Markup Language (SAML) 1.1 sender-vouches signed tokens and SAML 2.0 unsigned tokens for the web service provider scenario.

- SOAP Gateway can be run on a previously installed IBM distribution of the JDK.

- The SOAP Gateway management utility provides a command line or batch interface for configuring server properties, managing the server run time, and working with web service artifacts.
9.7.2 Web services security

To help you understand the WS-security implementation in combination with SAML, this section begins with a general overview of security for web services. You can use two types of security: transport-level security and message-level security (Figure 9-26), as explained here.

Figure 9-26  Message security and transport security

- Transport-level security (for example, HTTPS)
  
  This is a point-to-point security model where the channel is protected between two parties. Transport security starts with a SSL/Transport Layer Security (TLS) handshake between two transport endpoints, and creates a transport channel using symmetric encryption.
  
  Transport security is unaware of the frame protocol used on top of the channel.
  
  Often the service consumer and service provider are separated by intermediaries (for example, an enterprise service bus). In situations like these, each connection between the endpoints must be secured.

- Message security
  
  Message security travels with the message and has been formalized by the WS-Security standard. This outlines support for dynamic authentication on a per message basis for accessing IMS applications as web services. It details message-level security. The message level is part of the message, but it does not exclude the transport over a secure transport channel. This is even required if the authentication information, such as a user name and password, are passed in the clear.
WS-Security incorporates security features in the header of a SOAP message, working in the application layer. Thus it ensures end-to-end security. Figure 9-27 shows SOAP frames without and with message security in the header.

![Figure 9-27 Comparing the SOAP frame, without and with message security](image)

The security elements can be packed in different formats in the header. If the header is filled in directly by the requester of the web service, it can be the following types:

- Usertoken (UserID/password)
- Binary Token (509v3 certificate; not in Enterprise Suite Version 2.1)

If the header is asserted by an identity provider (IDp), it can be passed by a requester as a SAML token. This assumes that before the requester works with the web service, the requester first contacts a security server by, for example, a SOAP interaction, to obtain an asserted identification, which will not be contested by the SOAP gateway contact. This identification will be packed in a SAML token and transported with the SOAP header. The token, which represents the identification, can have the following content:

- SAML with Usertoken (UserId)
  - Format (SAML V1.1 (signed or nonsigned) and SAML V2.0)
- SAML with Binary Token (509v3 certificate, KERBEROS; not in Enterprise Suite Version 2.1 signed, or not signed with attributes)

In 9.7.3, “SOAP Gateway V2.1 security implementation” on page 351, we explain what is implemented in SOAP Gateway Version 2.1. For additional reading about WS-Security, see WS-Security AppNotes at:


**Message- and transport-level security:** Message-level security can provide an end-to-end security solution. It travels with the message. Transport-level security requires that it is implemented between all two-partner nodes in the transport path.
9.7.3 SOAP Gateway V2.1 security implementation

IMS SOAP Gateway can accept web services call-ins and can issue call-outs to a web service server. The security that is available is different.

Figure 9-28 shows what is supported in SOAP Gateway Version 2.1.

**supported security (call-in)**
- Transport Level (SSL/TLS)
- Message Level (WS-security)
  - UserName token Profile 1.0
  - SAML 1.1 unsigned sender-vouches tokens
  - SAML 1.1 signed sender-vouches tokens
  - SAML 2.0 unsigned sender-vouches tokens

**supported security (call-out)**
- Transport Level (SSL/TLS)

IMS 12 supports two basic scenarios for using web services:

- **On the call-out side**, IMS is the web service client. The only security implemented is based on the transport level. No user ID information is carried in the SOAP header of the SOAP messages sent to an external web service, but before using the transport a SSL/TLS session can be established (handshaken) between the two partners.
  
  A user can also use Basic Authorization, which carries user name and password within the HTTP stack.

- **On the call-in side**, the SOAP gateway acts as the server and accepts transport-level security and a subset of the WS-security, eventually in combination with SAML tokens. The security elements which are carried in the SOAP header are provided from the requester. It is the task of the SOAP gateway server to understand the security information, to extract it from the header, and to pass it to IMS by using the extended header of the IMS Connect IRM.

The following sections provide an overview of WS-security (eventually in combination with SAML) supported in Enterprise Suite SOAP gateway V2.1

**Security implementation without SAML**
The configuration that is supported without SAML is based on “UserName token profile 1.0”. The UserName token is carried in the SOAP header. This possibility was already available in Enterprise Suite SOAP Gateway V1.1.
Figure 9-29 shows the flow from requester to IMS. The real SOAP envelope (header, body) is used between the requester and the SOAP gateway.

Figure 9-29  Web service flow from requester to IMS with UserNametoken

The UserName and Password are extracted from the UserName token in the SOAP Gateway and passed to IMS Connect in the IRM request to IMS Connect. IMS Connect verifies the user ID and password with RACF.

The excerpt in Example 9-4 shows the format of the SOAP frame, with Header and Body.

Example 9-4  SOAPheader with UserNametoken

```xml
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header xmlns:wsa="http://www.w3.org/2005/08/addressing">
    <wsse:Security xmlns:wsse="http://docs.oasis-open.org/..">
      <wsse:UsernameToken xmlns:wsu="http://docs.oasis-open.org/..... .">
        <wsse:Username>
          USRT001
        </wsse:Username>
        <wsse:Password Type="http://docs.oasis-open.....Text">MYPASSWORD</wsse:Password>
      </wsse:UsernameToken>
    </wsse:Security>
    (NOT SHOWN)
  </soapenv:Header>
  <soapenv:Body>
    (NOT SHOWN)
  </soapenv:Body>
</soapenv:Envelope>
```

Inside the header, you can recognize the wsse:UsernameToken, with Username and Password. This information in the SOAP frame is in the clear in this case. The only way to protect it is by using transport-level security. Otherwise, the information passes in the clear over TCP/IP.
Security implementation with SAML

In the UserName token case, the identity/authentication username information came directly from the requester.

With SAML, this information must be obtained beforehand from an identity provider (security server). SAML is a specification designed to provide cross-vendor single-sign-on interoperability. SAML was developed by a consortium of vendors (including IBM) under the auspices of OASIS, through the OASIS SSTC (Security Services Technical Council). SAML has two major components:

- SAML assertions are used to transfer information within a single sign-on protocol.
- SAML bindings and profiles are used for a single sign-on protocol.
  - How SAML is employed in any given context is known as a SAML profile.
  - How SAML assertions or protocol messages are conveyed in or over another protocol is known as a SAML binding.

A SAML assertion is an XML-formatted token that is used to transfer user identity (and attribute) information from a user's identity provider to trusted service providers as part of the completion of a single sign-on request. A SAML assertion provides a vendor-neutral means of transferring information between federation business partners. The identity provider asserts the identity of the requester. In practice it happens by the requester contacting the identity server and obtaining this proof of identity in the format of a SAML token.

This token can be used for several purposes. In the case of web services the SAML token, which asserts the identity of the requester, is passed in the SOAP header, replacing the UserName token or other identity tokens.

The SAML token is a richer alternative to standard WS-security identity tokens because in addition to its identity/authentication content, it can also carry several additional attributes, such as access control information, that are related to the requester.

As a protocol, SAML has three versions: SAML 1.0, SAML 1.1, and SAML 2.0, as explained here:

- SAML 1.0 and SAML 1.1 (collectively, SAML 1.x) focus on single sign-on functionality.
- SAML 2.0 represents a major functional improvement over SAML 1.x. Approved in March 2005, SAML 2.0 is based on SAML 1.x with significant input from the Liberty Alliance ID-FF and Shibboleth specifications.

For more information about SAML, see the SAML Technical Overview at:

- Security Assertion Markup Language (SAML) V2.0 Technical Overview (March 2008)

See also Federated Identity Management and Web Services Security with IBM Tivoli Security Solutions, SG24-6394.

SOAP gateway V1.1 already supports SAML V1.1 unsigned. SOAP gateway V2.1 supports two additional versions of SAML.
**SAML 1.1 signed sender-vouches tokens**

With SAML 1.1 signed sender-vouches tokens, the requester contacts the security server (identity server) to obtain an identity assertion (a proof of identity), which is passed to the SOAP gateway. This model can also be called “push/post” because the assertion obtained from the security server is pushed with the “post” request to the web service.

A signed SAML token further protects message integrity by enabling the recipient of the token to validate the authenticity of the token and assert SAML token identity and attributes based on the trust relationship with the token issuer. Figure 9-30 illustrates the flow.

![Figure 9-30 Web service flow from requester to IMS with signed SAML 1.1 token](image)

The signed token has two variants:

- **SAML11SignedTokenTrustOne**
  
  Only one certificate is allowed to verify the digital signature.

- **SAML11SignedTokenTrustAny**
  
  All valid certificates carried together with the signed token are allowed to verify the digital signature.

Example 9-5 shows an excerpt of a SOAP header containing a signed SAML 1.1 token that is emitted to the requester by the security domain server.

**Example 9-5 SOAP header with SAML11SignedToken**

```xml
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-simpl...">
  <soapenv:Header>
    <wsse:Security>
      <saml:Assertion xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion">
        <saml:Conditions NotBefore="..." NotOnOrAfter="2011-08-03T22:...Z">
          <saml:NameIdentifier>Alice</saml:NameIdentifier>
        </saml:Conditions>
      </saml:Assertion>
    </wsse:Security>
  </soapenv:Header>
</soapenv:Envelope>
```
In Example 9-5, you see the SOAP header with WS-security included. The “wsse:..” section contains a “signed SAML11 token”. Within this <saml:Assertion..> token, you can see the following details:

- <saml:Subject>..</saml:Subject>
  - <saml:NameIdentifier>..</saml:NameIdentifier>. No credentials (?password) must be present; the user is asserted by the security server.
  - <saml:Attributes>..</saml:Attributes>

- <ds:Signature../>
  - <ds:SignedInfo../>
    <ds:SignatureMethod> transform, digest algorithms
  - <ds:SignatureValue>
  - <ds:KeyInfo>..</ds:KeyInfo>
    <ds:X509Data../>
    <ds:X509Certificate of the signer (if SAML11SignedTokenTrustAny)

**SAML 2.0 unsigned sender-vouches tokens**

With SAML 2.0 unsigned sender-vouches tokens, the requester contacts the security server (identity server) to obtain an identity assertion (a proof of identity), which is passed to the SOAP gateway. In this case, the token is not signed. Figure 9-31 illustrates the flow.

![Figure 9-31 Web service flow from requester to IMS with SAML 2.0 token](image-url)
SAML 2.0 is a newer standard that introduces features such as session management, attribute profiles, encryption, metadata specifications, and pseudonyms. Example 9-6 shows an excerpt of the SOAPheader with WS-security. It contains a SAML 2.0 token that is followed by attributes. The SOAPheader is then emitted to the requester by the security domain server.

Example 9-6   SOAPheader with SAML 2.0 token

```xml
<?xml version="1.0" encoding="utf-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:soapenv:Header="http://www.w3.org/2005/08/addressing">
  <wsse:Security xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
    <wsu:Timestamp xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-utility-1.0.xsd">
      <wsu:Created>getActualRunDateIso()</wsu:Created>
    </wsu:Timestamp>
    <Assertion..AssertionID=..IssueInstant=..Issuer=..MajorVersion=..MinorV" 
      <Conditions NotBefore=....NotOnOrAfter=....>
    </Conditions>
    <AuthenticationStatement AuthenticationInstant/Method=..
      <Subject>
        <NameIdentifier Format=...:unspecified">
          A202545<
          /NameIdentifier>
        <SubjectConfirmation>
          <ConfirmationMethod>
            urn:oasis:names:tc:SAML:1.0:cm:sender-vouches
          </ConfirmationMethod>
        </SubjectConfirmation>
      </Subject>
    </AuthenticationStatement>
  </wsse:Security>
  <wsa:......
    ....NOT shown
</wsa:Action>
```
9.7.4 Security setup for Enterprise Suite V2.1

The previous section explained aspects of WS-security and the use of SAML. The transport-level security between nodes connected by TCP/IP is also important. This requires the setup of SSL/TLS.

Without the use of SAML tokens, two actors are involved in the Web Services traffic: the requester (client) and the server hosting the web service (SOAP gateway). With SAML, there is a third actor: the security service (identity provider). All three must be able to communicate in a secure way, and eventually other security opportunities will be used (encryption, digital signatures).

SSL and TLS protocol technology protects data exchanges between client and server applications. SSL provides security for your interactions by securing the TCP/IP connection between SOAP Gateway and IMS Connect. TLS is the successor to the SSL protocol. TLS V1.0 was the first version, succeeding SSL V3.0. New TLS versions continue to be defined by the Internet Engineering Task Force (IETF), and the TLS protocol maintains compatibility modes for the earlier SSL protocol.

The SAML security server can also be called the “identity server”. This is the server that is contacted by the requester to obtain a SAML token with user ID and attributes included, which are asserted by this server. This SAML token is passed in the SOAP header between the requester and the SOAP gateway.
If you are using SAML, for example by using the SignedSAML11 token, and you also implement “transport security”, you can have an SSL setup as illustrated in Figure 9-32.

The figure contains the following details:

- A PUB/PRIV key pair (4) has been generated for each of the three actors, signed by a certificate authority (CA) (1) and deposed in the keystore (or RACF, if z/OS), for each actor.

  Signing by a CA normally requests the respect of a procedure. The signing must be done by officially recognized organizations (such as Verisign and Thawthe). However, for test purposes, you can set up a “self-signed CA”. With most of the tools and RACF on z/OS, you can generate a PUB/PRIV key pair for a self-signed CA, which for test purposes can then be used as a signer for all the other PUB/PRIV key pairs.

- Each truststore has to contain the “certificate” of the signers. The certificate of the CA must be present everywhere so that the SSL/TLS handshake between two participants can be certified. The certificate of the CA is used to validate the certificates of the user ID of the actors when they set up the SSL/TLS connection (2). This can be done by importing the CA certificate in the truststore and in the RACF database.

- If a signed SAML 1.1 implementation with option “SAML11SignedTokenTrustOne” was used, then the SAML token in the SOAPHeader was digitally signed by the security server. As a consequence, the “certificate” of the security server must be available in the truststore of the SOAP gateway server (3).

  If a “SAML11SignedTokenTrustAny” was used, then the certificate of the digital signer is carried with SOAP envelope/header.

Tooling is available on Distributed and z/OS platforms to install the keystores or truststores and their required contents. z/OS can use RACF for this.
All tools must have the following functions and capabilities:

- Generate PUB/PRIV key pairs
- Submit the generated certificate to a CA for signing
- Export the key pair in binary and 64-bit encoding for transport with protection by password
- Import key pairs (if password is known)
- Export the certificate in binary and 64-bit encoding for transport
- Import certificates
- View and list installed items

In a test environment, to avoid the cost of signing by official authorities, it is useful to have “self-signed” certificates for test CAs.

The following sections list and describe several of the many keytools that are available for z/OS and distributed environments.

**Security setup on z/OS**

With Fix Pack 2 and later, SOAP Gateway supports the use of the application transparent-TLS (AT-TLS) feature in IBM z/OS Communications Server for SSL protocol handling. Therefore, any SSL or TLS version that the AT-TLS feature supports, including TLS V1.0, TLS V1.1 and SSL V3.0, are supported.

AT-TLS is governed by a policy file, installed on a TCP/IP stack in z/OS. The policies in this file determine whether SSL/TLS will be used or not, depending in the most simple case on the IP/port combinations which will go in session. For example, all sessions connecting to SOAP gateway server port 8443 will have SSL/TLS.

To configure the IBM z/OS Communications Server AT-TLS feature for SOAP Gateway, use the IBM Configuration Assistant for z/OS Communications Server.

The high-level steps to configure the AT-TLS feature for SOAP Gateway are listed here:

1. In the Configuration Assistant, configure your z/OS image.
2. Enable the AT-TLS technology.
3. In the AT-TLS perspective:
   a. Configure the port, IP address, cipher, trace level, and keyring information for server authentication.
   b. Configure other settings such as:
      - Client authentication
      - Certificate selection
      - Certificate revocation list (CRL)
      - Connection settings such as cipher reset timer and SSL session timeouts
      - Additional authentication between SOAP Gateway and IMS Connect
4. Install the master AT-TLS policy configuration file.

After you activate AT-TLS, you must set up RACF as the store.

For information about SSL/TLS and AT/TLS, see *IBM z/OS V1R12 Communications Server TCP/IP Implementation: Volume 4 Security and Policy-Based Networking*, SG24-7899.

Next, we describe the preparation of SSL/TLS on z/OS. Basically you have to set up keydatabases and build the pub(certificates)/priv key pairs.
Using RACF

The information in RACF can be used directly by the AT-TLS implementation. RACF is in itself
the key and truststore.

If the SOAP Gateway requires the keystores or truststores in a hierarchical file system
(HFS/zFS), the preparation work and all generations can still be performed from RACF.

Exports of keys and certificates from RACF and imports in Local and Remote (after FTP)
keystores or truststores can do the ultimate setup.

RACF does not make a distinction between keystore or truststore. Everything is in one store.
The keys are maintained in a key ring. Before you can do anything, the user ID must be
known to RACF security and must have a key ring added. In practice, the name of the key ring
is qualified (username.keyringname). Example 9-7 shows what is required for the user
SOAPSVR1. SOAPSVR1 is signed with a “so-called” self-signed CA, which is generated first.

Example 9-7   RACF definition sample

* we assume that the user SOAPSVR1 does exist on z/OS
* if he has not a keyring yet, first create a keyring
* Creating SSL keyrings for SOAPSVR1
RACDCERT ADDRING(KeyringSV) ID( SOAPSVR1 )
* Generate a SELF SIGNED CA
RACDCERT CERTAUTH GENCERT SUBJECTSDN +
(CN('SGW CertAuth Security Domain') OU('SoapGateway Test setup in RACF'))+
WITHLABEL('SoapGatewayCA.SVL') TRUST NOTAFTER(DATE(2011/12/31))
* Connect the SoapGatewayCA with the keyring of the user SOAPSVR1
RACDCERT ID(SOAPSVR1) CONNECT +
(RING(KeyringSV) CERTAUTH LABEL('SoapGatewayCA.SVL') USAGE(CERTAUTH))
* Generate a PUB/PRIV keypair and sign directly with SoapGatewayCA
RACDCERT ID (SOAPSVR1) GENCERT +
SUBJECTSDN(CN('SOAPSERVER') O('IBM') OU('SVL')) +
WITHLABEL('DefaultCert.SOAPSVR1') +
SIGNWITH(CERTAUTH LABEL('SoapGatewayCA.SVL')) +
NOTAFTER(DATE(2011/12/31))
* Connect the SOAPSVR1 pub/priv with the keyring of user SOAPSVR1
RACDCERT ID(SOAPSVR1) CONNECT +
(RING(KeyringSV) LABEL('DefaultCert.SOAPSVR1') DEFAULT)
* Export the Certificate of the CA to a z/OS dataset, bin option
* also possible with 64bit encoding
RACDCERT CERTAUTH +
EXPORTLABEL('SoapGatewayCA.SVL')) DSN('SOAPGW.CACERT.DER') +
FORMAT(CERTDER)
* Export PRIV/PUB keys of user SOAPSVR1 to a z/OS dataset, bin option
* also possible with 64bit encoding
RACDCERT ID(SOAPSVR1) EXPORT (LABEL('DefaultCert.SOAPSVR1')) +
DSN('SOAPSVR1.USERCERT.P12DER') FORMAT(PKCS12DER) +
PASSWORD( 'MYSOAPGW')
* Transform the datasets to HFS files that can be
* -transported (FTP in this case BIN, mode)
* -imported in the keystores or truststores
OPUT 'SOAPGW.CACERT.DER' '/u/SOAPSVR1/SECURITY/CAGWCERT.der' BIN
OPUT 'SOAPSVR1.TMCERT.P12DER' '/u/SOAPSVR1/SECURITY/USERCERT.p12' BIN

We also show the exports of the certificate of the CA and the public or private information for
the user SOAPSVR1. Everything is done with one RACF command, RACDCERT.
In the **OPUT** command, the **BIN** option must be specified, because the exports were done in binary.

For test purposes, the task must be repeated for the other users’ requesters and security server. The benefits of this solution are that you have a central place where all security elements are preserved, and that the **RACDCERT** command is easy to use and can be executed by a batch job. When you have the file systems with certificate and P12 (public or private keys), you can easily distribute them and import in the users’ keystores and truststores.

**Using the gskkyman command**

The **gskkyman** UNIX command (Figure 9-33) is used to create and maintain digital certificate key databases in a z/OS UNIX file system. This is an alternative to storing digital certificates in the RACF database.

Be aware, though, that if you are using SSL/TLS client authentication to map a digital certificate to a RACF user ID, you must use the RACF **RACDCERT** command to store the client certificate, and not the **gskkyman** command.

```
VANAERS @ SC63:/u/vanaers>gskkyman

Database Menu

  1 - Create new database
  2 - Open database
  3 - Change database password
  4 - Change database record length
  5 - Delete database
  6 - Create key parameter file
  7 - Display certificate file (Binary or Base64 ASN.1 DER)

  11 - Create new token
  12 - Delete token
  13 - Manage token
  14 - Manage token from list of tokens

  0 - Exit program

Enter option number:
```

*Figure 9-33 Options for the gskkyman command*

In RACF, a key database is always present. In a gskkyman environment, the key databases, keystores, trust databases, or truststores must be explicitly created as public/private key, certificate repositories. In turn, those stores can be shared by any user IDs that have access using the HFS.

For more information, see IBM z/OS V1R12 Communications Server TCP/IP Implementation: Volume 4 Security and Policy-Based Networking, SG24-7899.

**Security setup on Windows**

Many tools are available to build stores and to build and import keys. This section highlights two of them. Keys generated under Windows can also be exported and even imported in z/OS RACF.
Using iKeyman

The iKeyman utility is a GUI-based tool that you can use to manage your digital certificates. With iKeyman, you can create a new key database or test a digital certificate, add CA roots to your database, copy certificates from one database to another, request and receive a digital certificate from a CA, set default keys, and change passwords.

The iKeyman utility is a part of the IBM Java Security Socket Extension package. It is shipped with the WebSphere Application Server and with many other Java-related packages.

Using keytool

The command line tool for Windows, keytool, is available in almost all jdk/bin and jre/bin directories. Figure 9-34 shows the options for keytool.

```
C:\Program Files\IBM\SDP_RDZ_8\jdk\bin>keytool
keytool usage:
  -certreq     [-v] [-protected]
  ......       
  -changealias [-v] [-protected] -alias <alias> -destalias <destalias>
  ......       
  -delete     [-v] [-protected] -alias <alias>
  ......       
  -export     [-v] [-rfc] [-protected]
  ......       
  -exportseckey [-v]
  ......       
  -genkeypair [-v] [-protected]
  ......       
  -genseckey  [-v] [-protected]
  ......       
  -help       
  -identitydb [-v] [-protected]
  ......       
  ......       
  -importkeystore [-v]
  ......       
  -importseckey [-v]
  ......       
  -keyclone   [-v] [-protected]
  ......       
  -keypasswd  [-v] [-all | -alias <alias>]
  ......       
  -list       [-v | -rfc] [-protected]
  ......       
  -printcert  [-v] [-file <cert_file>]
  -selfcert   [-v] [-protected]
  ......       
  -storepasswd [-v] [-all] [-new <new_storepass>]
  ......
```

Figure 9-34  Keytool options
9.7.5 SOAP Gateway Server

The SOAP Gateway server acts as the gateway between external web services and IMS applications. It serves as a web service server where IMS applications are enabled as web services. It serves a web service consumer when it forwards IMS application callout requests or business event data to external web services or event processing services.

In all scenarios, the SOAP Gateway server acts as a client to IMS Connect. The IMS Enterprise Suite V2.1 SOAP gateway server can run on both z/OS and Windows (Figure 9-35).

![Figure 9-35 SOAP Gateway as the web service server and the SSL/TLS client](image)

SOAP Gateway provides support for both server authentication and client authentication and Web-services security (WS-Security) for the web service provider scenario regardless of the platform that SOAP Gateway runs on.

SOAP Gateway clients can secure data exchanges with SOAP Gateway through HTTPS requests by using the SSL/TLS security protocol. Similarly, SSL/TLS connections are supported between SOAP Gateway and IMS Connect.

Enabling the SSL on the SOAP gateway server

If you want to use the possibilities of the WS-security, with or without SAML, the implementation of SSL/TLS in the transport between the SOAP gateway and IMS Connect is a solution. You can have HTTPS between the client and SOAP (provider) or SOAP and Service (callout). SSL between SOAP and ICON is optional. Between the client and the SOAP gateway, use HTTPS.

Tip: To configure the server to know where the Java is located, issue the following SOAP command either from UNIX System Services or with sample job AEWIOGBP:

```
iogmgmt -prop -u -java -h /path/to/java
```

Enablement on z/OS with AT-TLS

For the z/OS platform, you can use the IBM z/OS Communications Server AT-TLS feature using System Authorization Facility (SAF; RACF) to secure the connection. You can also benefit from the additional security features in AT-TLS or Quality of Service (QoS), such as security connection refresh settings, maximum connection settings, and revocation of certificates.
The traditional setup with keystores or truststores is available as well for z/OS, but the preferred selection of AT-TLS for the SOAP Gateway does not require the usage of the stores (Figure 9-36).

In addition to the inbound message connection rule for connections from the web service client (service requester) to SOAP Gateway, you can configure an additional set of rules for the connections between SOAP Gateway and IMS Connect.

Figure 9-36 shows the rules that are created to protect traffic in the following ways:

- Inbound traffic in the SOAP Gateway from the requester (assuming not in z/OS)
  
  Use the QoS feature in IBM z/OS Communications Server to specify traffic thresholds and traffic priority to help manage traffic to the SOAP Gateway server. QoS refers to the overall service that a user or application receives from a network, in terms of throughput and delay. To configure the maximum connections, create a traffic descriptor, a traffic shaping level, and a requirement map in the QoS perspective.

- Outbound traffic from the SOAP Gateway to IMS Connect

- Inbound traffic in IMS Connect from the SOAP Gateway

  The “policy rules” in the TCP/IP stacks that are passed in z/OS can all be different. Moreover, for each authentication (client/server), individual SAF key rings in RACF can be used and as a consequence a different set of certificates.

For information about setting up AT-TLS, see the IBM Information Management Software for z/OS Solutions Information Center at:

http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/index.jsp

In this information center, search for the following topics:

- Configuring AT-TLS for SOAP Gateway for server authentication
- Configuring maximum connections for SOAP Gateway
- Configuring additional connectivity rules for SOAP Gateway-to-IMS Connect communication
Enablement on z/OS with keystores or truststores

The tools that are available to build the stores and their contents are described in 9.7.4, “Security setup for Enterprise Suite V2.1” on page 358.

You make the SSL/TLS setup active by pointing to the stores and then activating SSL in the SOAP Gateway configuration for inbound and in the Connection bundles for outbound to IMS Connect.

Perform the following tasks:

- Modify the server.xml file to point to the keystore and truststore with the appropriate passwords (Example 9-8). The configuration for the SOAP gateway server is in the /..../IBM/IMS Enterprise Suite V2.1\SOAP Gateway\server\conf\server.xml file.
  - Set clientAuth to true.
  - Set sslProtocol to SSL.
  - Specify the file location for keystore and truststore.
  - Specify the password for keystore and truststore

- Connection bundle

  **SSL keystore name**
  This specifies the fully qualified path name of the keystore in which trusted certificates and private keys are stored.

  **SSL keystore password**
  This specifies the password for the keystore. The password length must be in the range of 6–20 alphanumeric characters.

  **SSL truststore name**
  This specifies the fully qualified path name of the truststore in which trusted certificates are stored.

  **SSL truststore password**
  This specifies the password of the truststore in which trusted certificates are stored. The password length must be in the range 6–20 alphanumeric characters.

  **SSL encryption level**
  This specifies the encryption type. A value of Strong indicates that a strong cipher suite needs to be used. A value of Weak indicates that a weak cipher suite needs to be used. A value of None indicates that no encryption is used. The None encryption level is used only for authentication.

Attention: Leave these properties blank if you are using IBM z/OS Communications Server AT-TLS feature to secure the connection between the SOAP Gateway and IMS Connect.

Enablement on Windows

For Windows, you can only use keystores or truststores. The build of the stores and their contents and the available tools are described in 9.7.4, “Security setup for Enterprise Suite V2.1” on page 358.

You make the SSL/TLS setup active by pointing to the stores and activating SSL in the SOAP Gateway configuration for inbound, and in the Connection bundles for outbound to IMS Connect.
Perform the following tasks:

- Modify the file server.xml to point to the keystore and truststore with the appropriate passwords. See Example 9-8.

The configuration for the SOAP gateway server is in the C:\Program Files\IBM\IMS Enterprise Suite V2.1\SOAP Gateway\server\conf\server.xml file.

- Set clientAuth to true.
- Set sslProtocol to SSL.
- Specify the file location for keystore and truststore.
- Specify the password for keystore and truststore.

Example 9-8  Changes to the server.xml file for enabling SSL port 8443

```xml
<!-- Define a SSL HTTP/1.1 Connector on your port -->
<Connector port="8443"
    maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
    enableLookups="false" disableUploadTimeout="true"
    acceptCount="100" debug="0" scheme="https" secure="true"
    clientAuth="true" sslProtocol="SSL"
    keystoreFile="C:\Program Files\IBM\IMS Enterprise Suite V2.1\SOAP Gateway\server\conf\SGWkeystore.ks"
    keystorePass="password"
    truststoreFile="C:\Program Files\IBM\IMS Enterprise Suite V2.1\SOAP Gateway\server\conf\SGWtruststore.ks"
    truststorePass="password"
    algorithm="IbmX509"/>
```

You can also make the changes shown in Example 9-8 by using the “SOAP Gateway management utility” command:

```
iog_mgmt -prop..." command option
```

- Connection bundle
  - This is the same as for z/OS.
    - SSL keystore name:
    - SSL keystore password:
    - SSL truststore name:
    - SSL truststore password:
    - SSL encryption level:

Starting the SOAP Gateway

This section explains how to start the SOAP Gateway on z/OS and on Windows, depending on your environment.

Starting the SOAP Gateway on z/OS

Example 9-9 shows how to start the gateway.

Example 9-9  Starting the SOAP Gateway

```
/S es21sgb1
```

ES21SGB1 is simply a customized name for the start procedure distributed by IBM under the name AEWIOGPR.
Example 9-10 shows the results as displayed on the console after a short time.

**Example 9-10  Messages on the console for start of SOAP gateway**

```
IEF695I START ES21SGB1 WITH JOBNAME ES21SGB1 IS ASSIGNED TO USER IMSSOAP, GROUP SYS1
$HASP373 ES21SGB1 STARTED
10G300011I: The SOAP Gateway server is now up and running. Elapsed 233 time is 14,738,455,125 nano seconds. //ES21SGB1 JOB MSGLEVEL=1
//STARTING EXEC ES21SGB1
```

The JOB is associated with user IMSSOAP. This is a result of the RACF association with the started task name.

This JOB is executed as a JZOS job; the required Open MVS environment is read from member JDKSEPB1. The procedure is shown in Example 9-11.

**JZOS Batch Toolkit for z/OS SDKs:** The IBM JZOS Batch Toolkit for z/OS SDKs is a set of tools that improves many of the functional and environmental characteristics of the current Java batch capabilities on z/OS. It includes a native launcher for running Java applications directly as batch jobs or started tasks, and a set of Java methods that make access to traditional z/OS data and key system services directly available from Java applications.

**Example 9-11  START procedure for SOAP Gateway on z/OS**

```
Chapter 9. IMS Enterprise Suite V2.1

Starting the SOAP Gateway on Windows

The SOAP Gateway server can be started (and stopped) from the installation menu (Figure 9-37).

Example 9-12 shows the upcoming messages in the Windows console.

Example 9-12  Start of SOAP gateway server on Windows

IOG30006I: Checking on server status.
INFO: Initializing Coyote HTTP/1.1 on http-8080
INFO: Initialization processed in 2026 ns
[2011-08-05 22:17:35,468] main ParserStartupRoutines
ERROR: IOGS0104E: An I/O exception occurred with the (C:\Program Files\IBM\IMS
Enterprise Suite V2.1\SOAP Gateway\server\webapps\imssoap\WS-SECURITY\SAML20Sign
edTokenTrustAny\server\policy.xml) file. Either the file could not be read or fo
und, or the file encoding is not UTF-8.
Message= SOAP Gateway was unable to load file into memory, defaulting to I/O int
ensive routine..
INFO: Starting service Catalina
INFO: Starting Servlet Engine: Apache Tomcat/6.0.18
INFO: Starting Coyote HTTP/1.1 on http-8080
INFO : IOG30001I: The SOAP Gateway server is now up and running. Elapsed time i
s 18,792,597,662 nano seconds.

Stopping the SOAP Gateway
You can stop the SOAP Gateway on z/OS in the following ways:

- The p(stop) command (Example 9-13), which is the preferred method.
- The c(cance1) command; issue this command when purge does not work.

Example 9-13  Purge command

/P es21sgb1

A p(stop) generates the messages shown in Example 9-14.

Example 9-14  Messages on the console for purge

P ES21SGB1
JZOS - MVS STOP command received
IOG30004I: The SOAP Gateway server is stopped. 208

On Windows, you can stop the SOAP Gateway server from the installation menu (Figure 9-37 on page 369).

9.7.6 SOAP Gateway management utility

The SOAP Gateway management utility, iogmgmt, provides a command line or batch interface for configuring server properties, managing the server run time, and working with web service artifacts. The command-line interface (CLI) facilitates task automation and management flexibility.

With the SOAP Gateway management utility, you can perform the following tasks:

- Start and stop the SOAP Gateway server on Windows systems:
  - For z/OS, use the START and STOP console commands.
  - For Linux on System z, run the iogstart.sh and iogstop.sh scripts.
- Configure SOAP Gateway server properties.
Enable your IMS application as a web service provider or a web service consumer (callout), or to emit business event data.

The SOAP Gateway management utility also contains the deployment function, which previously was performed by the deployment utility.

An `iogmgmt` command consists of the `iogmgmt` statement followed by arguments to specify the command and associated options. The management utility can be used in z/OS and distributed environments.

The universal `iogmgmt` command has many options, which are explained with the specific task. For details about commands, see the IBM Information Management Software for z/OS Solutions Information Center at the following address, and search for SOAP Gateway management utility reference:

http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/index.jsp

You can invoke the `iogmgmt` command in the following way; Example 9-15 lists the available options.

**Example 9-15  Options for the iogmgmt utility**

-`callout -startall/-startone`: Start threads
-`callout -startpool`: Start the thread pool
-`callout -stopall/-stopone`: Stop threads
-`callout -stoppool`: Stop the thread pool
-`callout -updateprop`: Update SOAP Gateway callout properties
-`conn`: Create, update, or delete a connection bundle
-`corr`: Create or update a correlator entry
-`deploy`: Deploy a web service or callout application
-`diagnose`: Diagnose SOAP Gateway problems
-`migrate`: Migrate and upgrade SOAP Gateway
-`prop`: Set/Modify SOAP Gateway server properties
-`start`: Start the SOAP Gateway server
-`stop`: Stop the SOAP Gateway server
-`undeploy`: Undeploy a web service or callout application
-`view -connectionbundle/-connectionbundleentry`: view connection bundle entries.
-`view -correlatorfile`: View correlator information
-`view -calloutproperties`: View callout properties
-`view -calloutthreads`: View the status of callout threads
-`view -soapgatewayproperties`: View SOAP Gateway server properties
-`view -workerthreads`: View status of the callout worker thread pool

**The utility on z/OS**

The SOAP Gateway management utility, `iogmgmt`, is only available from within a UNIX System Services session. UNIX System Services is a required component of z/OS. For SOAP Gateway servers running on z/OS, the SOAP Gateway management utility must run on the same LPAR as the target server.

You can invoke the commands in the following ways:

- Invoke a UNIX System Services session as an OMVS command.
  
  Switch to the SOAP Gateway management utility directory with the following command:
  ```
  cd SOAP_Gateway_home_directory/deploy
  ```
Invoke the SOAP Gateway management utility by entering the following command:

```bash
./iogmgmt ...arguments
```

- Use a UNIX System Services Batch submit (Example 9-16).

**Example 9-16  BPXbatch job control language (JCL)**

```bash
//AEWIOGBP  JOB 'SOAP BATCH',CLASS=A,REGION=0M,MSGLEVEL=(1,1),
 //  MSGCLASS=H,USER=UUUUUUUU
//*
 picsurance **************************************************************
//* LICENSED MATERIALS - PROPERTY OF IBM                              *
//* 5655-T62 (C) COPYRIGHT IBM CORP 2011                             *
//* ALL RIGHTS RESERVED.                                               *
//* US GOVERNMENT USERS RESTRICTED RIGHTS -                          *
//* USE, DUPLICATION OR DISCLOSURE RESTRICTED                        *
//* BY GSA ADP SCHEDULE CONTRACT WITH IBM CORP.                       *
//********************************************************************
//*
// BPXBATCH SAMPLE NAME:        AEWIOGBP                              *
// AEWIOGBP VERSION:            2.1.0                                 *
//*
// DESCRIPTION: THIS WILL EXECUTE IOGMGMT COMMANDS FROM BPXBATCH.     *
// SEVERAL SAMPLE COMMANDS ARE DEMONSTRATED. MODIFY THIS              *
// ACCORDINGLY.                                                      *
//*
// NOTES:                                                            *
// 1) SET VARIABLE IOGPP TO YOUR -PathPrefix- SPECIFIED IN JOB        *
//    AEWJSMKD. IF YOU DID NOT CONFIGURE THE -PathPrefix- THEN        *
//    DO NOT CHANGE IOGPP.                                          *
//    E.G. With a -Pathprefix of "/s/esuite21/"                     *
//    IOGPP="/s/esuite21"                                           *
//    E.G. Without a path -Pathprefix-                             *
//    IOGPP=                                                      *
//    E.G.                                                        *
// 2) BELOW IS AN EXAMPLE OF SEVERAL IOGMGMT COMMANDS CONTAINED IN    *
//    AN ARRAY CMD. WHEN ADDING MORE REPEAT THE SYNTAX AS NEEDED.    *
//    SYNTAX:                                                        *
//    CMD[3]="iogmgmt [-OPTION] [ARG] " ; +                        *
//    SEE IOGMGMT COMMANDS FOR OPTIONS AND ARGS                      *
//    E.G.                                                          *
// 3) IF YOU WANT TO SPECIFY A TCP/IP STACK NAME ADD THE EXPORTED    *
//    VARIABLEBELOW VARIABLE IOGPP.                                  *
//    E.G.                                                          *
//    export _BPXK_SETIBMOPT_TRANSPORT=stack_name; +                *
//    E.G.                                                          *
// 4) MODIFY THE PATH AND FILE NAMES FOR THE OUT LOG AND ERROR        *
//    LOG CURRENTLY SET TO /tmp/iogmgmt.out.txt AND                  *
//    /tmp/iogmgmt.err.txt IF YOU WANT TO HAVE THEM WRITTEN          *
//    ELSEWHERE. IF YOU MODIFY THE LOG PATH YOU MUST ALSO MODIFY     *
//    THE JAVAOUT AND JAVAERR PATH TO MATCH.                        *
//********************************************************************
//*
//STEP1  EXEC PGM=IKJEFT01,DYNAMNBR=250,REGION=OM
//SYSPRINT DD SYSOUT=*                                           
//SYSTSPRT DD SYSOUT=*                                          
```
The utility on Windows

To execute the utility, go to the deploy directory of the SOAP gateway installation as indicated in Example 9-17. From here on, you can run the .bat file with the required parameters.

SOAP Gateway management utility commands are case-sensitive and must be entered in all lowercase. Alternatively, you can enter them as shown in the IBM Information Management Software for z/OS Solutions Information Center at the following address (search for command reference):

http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/index.jsp

Example 9-17   Starting the managed tool without parameters on Windows

C:\Program Files\IBM\IMS Enterprise Suite V2.1\SOAP Gateway\deploy>io MGM
I0GD0201E: The igomgmt command failed because a valid command parameter was not specified.

A SOAP Gateway management utility command requires a valid command parameter.

See the SOAP Gateway management utility command reference information for a list of valid command parameters.
9.7.7 Implementing a call-in web service for IMS

The call-in web service has several changes. However, the call-out web service did not change in IMS 12. The implementation of a web service for existing IMS transactions requires several steps. Some steps are required for all new web services, but other steps are shared later.

Building the web service
The implementation of a new web service requires some activity in an Eclipse Workbench environment. You can use the Enterprise Service Tools option in IBM Rational Developer for System z with Java to help you. You must complete the work in an IMS Enterprise Suite SOAP Gateway Project.

This opportunity already existed in IMS 10 and IMS11. Documentation and examples are available in the following documents:

- IMS 11
- IMS Phonebook sample

Insist on the following setup during the definition process:

- If using WS-security (UserName token or SAML token), in the Generation Options (WSDL and XSD) window, the service location must indicate secure HTTP (HTTPS) and reference the corresponding SSL-enabled port on the SOAP gateway.
- In the IMS Enterprise Suite SOAP Gateway Web Service Provider window, WS-Security must be enabled.

After successful completion of this preparation, Figure 9-38 shows the data sets that are available.

![Figure 9-38 Enterprise Suite Project in Rational Developer for System z Version 8](image)

The following data sets are available:

- **IMSPHBKD.cbl** The COBOL converter and driver file, to be installed on the host for IMS Connect.
- **IMSPHBK.xml** The correlator file, to be installed in the SOAP gateway (distributed or z/OS). This file contains a reference to the “connection bundle”, which describes the connection.
- **IMSPHBK.wsdl** Web Service Description File, which can be used to build a client embryo.
IMSPHBKI.xsd  The inbound schema file (not used by the SOAP Gateway).
IMSPHBKO.xsd  The outbound schema file (not used by the SOAP Gateway).

For usage of the SAML token, the correlator can require a manual update to indicate what token is present in the SOAP header. This information must be set in extendedProperty1. Example 9-18 shows the SAML11SignedTokenTrustAny correlator file.

Example 9-18  Updated correlator file for extendedProperty.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<correlator:correlator xmlns:correlator="http://www.ibm.com/IMS/Correlator" version="1.0">
  <wsdlFile name="IMSPHBK.wsdl"
targetNamespace="file://target.files1312407250521"/>
  <serviceTraceLevel>Error</serviceTraceLevel>
  <correlatorEntry operationName="IMSPHBKOperation" portName="IMSPHBKPort"
serviceName="IMSPHBKService">
    <adapterType>IBM XML Adapter</adapterType>
    <converterName>IMSPHBKD</converterName>
    <connectionBundleName>IMSPHBK</connectionBundleName>
    <socketTimeout>0</socketTimeout>
    <executionTimeout>0</executionTimeout>
    <ltermName></ltermName>
    <inboundTPIPEName></inboundTPIPEName>
    <inboundCCSID>1208</inboundCCSID>
    <hostCCSID>1140</hostCCSID>
    <outboundCCSID>1208</outboundCCSID>
    <trancode>IVTNO</trancode>
    <calloutConnBundleName></calloutConnBundleName>
    <calloutWSTimeout>7500</calloutWSTimeout>
    <enabledWSS>true</enabledWSS>
    <calloutURI></calloutURI>
    <extendedProperty1>SAML11SignedTokenTrustAny</extendedProperty1>
    <extendedProperty2></extendedProperty2>
  </correlatorEntry>
</correlator:correlator>
```

Connection bundle

The connection bundle describes the characteristics of the SOAP Gateway connection to the IMS Connect server. For TCP/IP, the SOAP Gateway is always a client. The name of the connection bundle has been nominated during the build of the web service, and we have to create it by using the command shown in Example 9-19.

Example 9-19  Creation of the connection bundle

```
iogmgmt -conn -c -n IMSPHBK -d I12A -h wtsc63.itso.ibm.com -p 5100
IOGD0113I: The create connection bundle entry (IMSPHBK) command successfully changed the SOAP Gateway master configuration. The parameters submitted with the command were:
-conn
-c
-n IMSPHBK
-d I12A
-h wtsc63.itso.ibm.com
-p 5100.
```
The SOAP Gateway server file system was updated. The changes will be reflected in the runtime configuration of the server after the next time that the SOAP Gateway starts.
No action is required.

Many other parameters are available for the connection bundle. In addition to the options shown in the example, the options listed here are also available. The command is used to create, update, and delete the bundle.

Example 9-20 shows the following information:

- The options -c for create, -d for delete, and -u for update.
- The host_name and port_number referencing the IP address of IMS Connect
- The data_store_name from which we are accepting messages
- The callout_tpipe name, used with the -callout parameter

Example 9-20   Create, delete, update options for the connection bundle

```
iogmgmt
> -conn -c -n--bundle_name
   -d -n--bundle_name
   -u -n--bundle_name

   -h host_name
   -p port_number (default 9999)

   -d--datastore_name -i callout_tpipe_name
```

Most of the other parameters deal with the security references of the SOAP Gateway as a client as shown in Example 9-21, where:

- `saf_user_ID`, `saf_password`, and `saf_group_name` are the default values passed in the extended IRM type-2 header, if no user information is received from ws_security
- `r new_bundle_name` to rename the connection bundle
- `i callout_tpipe_name`
- `...callout_targets...` to call out with a client or server and basic authentication

Example 9-21   Additional options available on create connection bundle

```
-f saf_user_ID -s saf_password
-g saf_group_name
-r new_bundle_name
-i callout_tpipe_name
-l callout_target_keystore_name -y callout_target_keystore_password
-v callout_target_truststore_name -q callout_target_truststore_password
-m callout_target_basic_auth_id -b callout_target_basic_auth_password
```

The options in Example 9-22 describe SSL characteristics for the connection to IMS Connect, when AT-TLS is not used.

Example 9-22   Parameters not used in combination with AT-TLS in TCP/IP

```
-k keystore_name -w keystore_password
-t truststore_name -o truststore_password
-e encryption_type
```
Deploying the web service

The web service prepared by the Eclipse tool must be deployed in the SOAP gateway, which is done with SOAP Gateway management utility. During deployment of the web service, in some cases you need to indicate the kind of identity (authentication) information that is carried in the SOAP header.

Deploying the web service to SOAP Gateway enables the application and allows it to begin processing client requests. Before you deploy the web service, you must have the WSDL file name, correlator XML file name, and connection bundle entry name for the web service.

To deploy the web service, complete the following steps:

1. Issue the following command with the name of the WSDL file and correlator XML file:
   
   ```
   iogmgmt -deploy -w wsdl_file -r correlator_file
   ```
   
   The SOAP Gateway management utility parses the correlator XML file to determine the name of the linked connection bundle. If the connection bundle is not already active in the runtime cache, it is loaded when the service is deployed.

2. Optional: Specify a security token type (for example, `-t SAML11Token`) to enable WS-Security for the service.

   See Figure 9-39.

3. Specify whether the service requires SSL transport. As shown previously for z/OS, you can choose between AT-TLS setup and a setup with keystores or truststores. Windows supports only keystores or truststores. SSL setup will be required in case of an incoming
web service request over SSL and if the SAML11token was signed, we need certificates in the truststore to verify the digital signature.

The deployment must be done with the `iogmgmt` utility. Example 9-23 shows the SAML11SignedTokenTrustAny security token.

**Example 9-23   Deployment of the web service with a SAML11SignedTokenTrustAny token**

```plaintext
iogmgmt -deploy
-w C:\temp\IMSPHBK.wsdl -r c:\temp\IMSPHBK.xml -t SAML11SignedTokenTrustAny

IOGD0104I: The deploy command successfully deployed the
IMSPHBKService web service to the runtime and master configurations:
Web service definition and associated schema XML files:
  C:\temp\IMSPHBK.wsdl
Correlator XML file:
  c:\temp\IMSPHBK.xml.

The -t parameter has the following options:
- UserNameToken
- SAML11Token (unsigned)
- SAML11SignedTokenTrustOne

The SAML11 token in the SOAPheader carries the identity signed by the security server. This signature can only be verified by one element, the certificate of the security server, which must be available in the truststore of the SOAP Gateway server.

The following additional parameters are mandatory for the truststore:
- `-y` JCEKS/JKS/PKCS12 the truststore type
- `-p` truststore password
- `-h` truststore path
- SAML11SignedTokenTrustAny

The SAML11 token in the SOAPheader carries a signed identity. This signature can be verified by the certificate that travels with the SOAP request.

- SAML20Token (unsigned)

4. Optional: Verify that the service is active with the following command:
   ```plaintext
   iogmgmt -view -correlatorfile ALL
   ```

   The correlator file for the newly-deployed service appears in the list of active correlator files in the runtime configuration. If the server is stopped, the correlator file name appears in the list of correlator files in the master configuration instead.

**Message security**

SSL deals with the transport-level security. This section explains what happens to the message security, which is security carried in the SOAP header of the envelope (Figure 9-40 on page 379). The transport-level security should eventually protect the message level security so that it cannot be read or altered.

It is the function of the SOAP Gateway to pass the message security from the SOAP header into the header extensions of the IMS Connect request message (IRM TYPE 2) as RACF recognizable information. By the type parameter, specified during the deployment, the SOAP Gateway server is aware of the type of security in the SOAP header.

If the security is not available from the SOAP header, defaults are taken for USERID/GROUP and PASSWORD specifications in the connection bundle.
Figure 9-40  Passing WS_security into the IRM request to IMS Connect

**COBOL converter**

The COBOL converter function did not change from previous releases. To handle the XML data from the client, you can either modify the IMS application to accept the XML input message and to return an XML output message, or use the IBM Rational Developer for System z XML converter drivers to transform the XML data in IMS Connect. These converter drivers are generated in the COBOL or PL/I language.

The IMS Connect XML adapter function allows the XML adapter to run the converter driver inside IMS Connect to perform the XML transformation in the IMS Connect address space. To handle IMS transaction input and output messages in XML format from the SOAP client, the XML adapter converts the XML-tagged data to the appropriate IMS application format that the IMS application accepts and vice versa for outgoing messages.

If the IMS application is a multi-segment message processing program (MPP), you must use IBM Rational Developer for System z to generate the XML converter SOAP Gateway server administration.

The COBOL XML converter driver that was generated by Rational Developer for System z as part of the WSDL generation must be uploaded to your host IMS machine and compiled and link-edited so it can be accessed by IMS Connect.

To configure IMS Connect to convert XML data from the client into COBOL or PL/I IMS application program data, complete the following basic steps:

1. Specify the HWSSOAP1 user message exit in the EXIT= parameter of the TCP/IP configuration statement (see [1](#) in Example 9-24).
2. Include the ADAPTER configuration statement, ADAPTER=(XML=Y), in the IMS Connect configuration member HWSCFGxx (see [2](#) in Example 9-24).

**Example 9-24  Changes in IMS Connect configuration for SOAP gateway**

```
HWS=(ID=HWSI12A2,PSWDMC=N,RACF=N,RRS=Y,UIDCACHE=Y,XIBAREA=50)  
TCPIP=(EXIT=(HWSSMPL1,HWSOAP1),HOSTNAME=TCPIP,PORT(ID=5100),  
   PORT(ID=5101),PORT(ID=5102))  
DATASTORE=(GROUP=I12XOTMA,ID=I12A,MEMBER=HWBI12A2,TMEMBER=I12AOTMA)  
ADAPTER=(XML=Y)
```

[1](#) [2](#)
3. Define the XML adapter as a BPE exit routine for IMS Connect by coding a BPE exit list in a IMS PROCLIB data set member (Example 9-25).

   **Example 9-25   XML adapter as a BPE exit routine in member HWSEXIT0**

   ```
   EXITDEF(TYPE=XMLADAP,EXITS=(HWSXMLA0),ABLIM=8,COMP=HWS)
   ```

4. Reference it from the BPE IMS PROCLIB member (Figure 9-26).

   **Example 9-26   Excerpt of IMS Connect BPE member**

   ```
   EXITMBR=(HWSEXIT0,HWS)
   # DEFINITIONS FOR SOAP GATEWAY
   ```

5. For COBOL application programs, the XML converter is based on the COBOL copybook of the IMS COBOL application program that processes the message. For PL/I application programs, the XML converter is based on the source of the PL/I application program. Each IMS application that processes messages converted from XML must have its own unique XML converter.

   The XML converters run in an IBM Language Environment for z/OS enclave in the IMS Connect region and use approximately 33 MB of storage. The IMS Connect region size must be increased to accommodate this storage requirement. Compile and link the XML converter into an APF-authorized data set that is concatenated to the STEPLIB in the IMS Connect startup JCL. When linking the XML converter, specify an additional program entry name for an internal service as an ALIAS in the link job.

   **Refreshing an existing converter:** If you are updating an existing converter, refresh it by issuing one of the following commands:

   - The z/OS Modify command `UPDATE CONVERTER`
   - The WTOR command `REFRESH CONVERTER`
   - The type-2 command `UPDATE IMSCON TYPE(CONVERTER)`

   You can find details about this installation in the IBM Information Management Software for z/OS Solutions Information Center at the following address. Search for `configuring XML conversion support for IMS Connect clients for IMS 12`:

   `http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/index.jsp`

   Deploying the web service to SOAP Gateway enables the application and allows it to begin processing client requests. Before you deploy the web service, you must have the WSDL file name, correlator XML file name, and connection bundle entry name for the web service.

   - Configure and remove the configuration for a web service or business event server.
   - Set up the connection and correlation properties for a web service.
   - Configure callout properties, and manage the callout threads and thread pool.
9.7.8 SOAP Gateway Administrative Console

The SOAP Gateway Administrative Console lists the deployed web services when the server is started. Each item in the list is a link to the WSDL file for the web service. The SOAP Gateway server must be started before proceeding. If the SOAP Gateway is stopped, issue the command shown in Example 9-27.

Example 9-27 Command to retrieve information from the file system configuration of a stopped server

```
iogmgmt -view -correlatorfile ALL
```

The web-based Administrative Console is not available if the server is stopped. However, this command retrieves information from the master (file system) configuration of a stopped server instead of the runtime configuration.

To start the SOAP Gateway Administrative Console, choose one of the following options depending on your environment:

- For Windows, from the Start menu, select **Start** → **Programs** → **IBM IMS Enterprise Suite Vx.x** → **SOAP Gateway** → **Administrative Console**.
- For Linux on System z and z/OS, from a web browser, enter the address shown in Example 9-28, where ```hostname``` is the host name and ```port``` is the port number where SOAP Gateway is running. The default port number is 8080.

Example 9-28 Start the SOAP Gateway Administrative Console from browser

```
http://hostname:port/imssoap
```

The Administrative Console opens in a web browser. Click **View Deployed web services**. The list of the currently deployed web services is displayed. Each item in the list is a link to the web service's WSDL file.

For more information, see the IBM Information Management Software for z/OS Solutions Information Center at the following address, and search for **IMS Enterprise Suite SOAP Gateway for IMS 12**:

```
http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/index.jsp
```

9.8 IMS Enterprise Suite Java Message Service API

The IBM Enterprise Suite JMS API component is not changed from IMS Enterprise Suite V1.1. We include it in this chapter for completeness.

JMS is a Java technology to access queue or topic resources. Each vendor offering with the Java language the possibility “to write to/read from queues (point-to-point (PTP) domain)” or “to publish to/subscribe on topics (PUB/SUB domain)” has to implement the core abstract JMS classes. An abstract class contains only the signatures of the methods, but has no logic included.

Implementation classes provide the actual interface to the resources. IBM has its implementation classes for WebSphere MQ.

The dlicall “ICAL” for IMS dependent regions allows a program to perform synchronous callouts when using DLI calls, from traditional languages such as PL/I, COBOL, C, and Java.
With the JMS extension, Java IMS dependent regions (JBP, JMP) can issue the “ICAL” equivalent in a JMS way. The extension is delivered for IMS as a package with classes extending the standard WebSphere MQ JMS implementation package (Example 9-29).

**Example 9-29  WebSphere MQ and IMS extension classes for JMS**

| com.ibm.mq.jms.jar (IBM implementation for MQseries) |
| com.ibm.ims.jms.jar (extension for IMS/ICAL) |

The IMS implementation of JMS is limited to supporting the PTP messaging domain only. In addition, support is provided only for non-transacted QueueSession objects with `Session.AUTO_ACKNOWLEDGE` mode. If the JMP or JBP application attempts to call any JMS method not supported by IMS or pass any unsupported argument to JMS method calls, a JMSException exception is thrown.

The `com.ibm.ims.jms` package provides IMS-specific extensions for performing synchronous call-out from JMP or JBP regions. It contains the `IMSQueueConnectionFactory` class, which is the Factory class used to create IMS point-to-point messaging domain QueueConnection objects.

JMS calls can be issued to three target types:
- JMS/ICAL to StateLess Session Bean (SLSB), located in WebSphere Application Server
- JMS/ICAL to Message Driven Bean (MDB), available in WebSphere Application Server
- JMS/ICAL to web services by using the SOAP Gateway

Figure 9-41 on page 382 illustrates the following flow:
1. JMS ICAL is issued.
2. It nominates an OTMA descriptor, which dictates the target.
3. It passes through IMS Connect, and Web Services call messages are prepared for XML.
4. If the destination is SLSB or MDB, it passes through the IMS/TM adapter for WebSphere Application Server.
5. Destination is Stateless Session Bean.
6. Destination is Message Driven Bean.
7. If the destination is Web Services, it passes through the IMS SOAP Gateway.
8. Destination is Web Services.

![Figure 9-41  JMS ICAL destinations](image-url)
Example 9-30 shows an excerpt of the JMS code.

```java
import javax.jms.JMSException;
import javax.jms.Queue;
import javax.jms.QueueConnection;
import javax.jms.QueueRequestor;
import javax.jms.QueueSession;
import javax.jms.Session;
import javax.jms.TextMessage;
import com.ibm.ims.jms.IMSQueueConnectionFactory;
public class IMS_Sample {
    public static void main(String argv[]) {
        IMSQueueConnectionFactory jmsConnectionFactory = new IMSQueueConnectionFactory();
        QueueConnection jmsConnection = null;
        QueueSession jmsQueueSession = null;
        Queue jmsQueue = null;
        QueueRequestor jmsQueueRequestor = null;
        try {
            jmsConnectionFactory.setTimeout(1000);
            //Set 10 second timeout
            jmsConnectionFactory.setResponseAreaLength(128000);
            //Allocate 128K response area
            jmsConnection = jmsConnectionFactory.createQueueConnection();
            jmsQueueSession = jmsConnection.createQueueSession(false, Session.AUTO_ACKNOWLEDGE);
            jmsQueue = jmsQueueSession.createQueue("OTMDEST1");
            jmsQueueRequestor = new QueueRequestor(jmsQueueSession, jmsQueue);
            TextMessage sendMsg = jmsQueueSession.createTextMessage();
            String msg = "My Message";
            sendMsg.setText(msg);
            TextMessage replyMsg = (TextMessage)jmsQueueRequestor.request(sendMsg);
            System.out.println("Received message: "+replyMsg.getText());
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

To send a message using the JMP and JBP support for synchronous callout and synchronously receive a response:

2. Create a JMS QueueConnection instance by calling the createQueueConnection method on the IMSQueueConnectionFactory object.
3. Create a JMS QueueSession instance by calling the createQueueSession method on the QueueConnection instance. In the method call, you must set the input parameter values to false and Session.AUTO_ACKNOWLEDGE to specify that the generated QueueSession instance is non-transacted and runs in AUTO_ACKNOWLEDGE mode.
4. Create a queue identity by calling the createQueue method on the QueueSession instance. In the method call, you must set the input parameter value to the OTMA descriptor name for the synchronous callout operation.
5. Create a JMS QueueRequestor instance and pass in the QueueSession instance from step 3 and the Queue instance from step 4 as input parameters to the QueueRequestor constructor method.

6. Create a TextMessage instance by calling the createTextMessage method on the QueueSession instance from step 3. Set the string containing the message data.

7. To send the message and retrieve a response, call the request method on the QueueRequestor object from step 5. In the method call, pass in the TextMessage instance from step 6. You need to cast the return value from the request method call to a TextMessage instance. If the call is successful, the return value is the response to the synchronous callout request.

For more information, see the IBM Information Management Software for z/OS Solutions Information Center at the following address, and search for JMS API:

http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/index.jsp
Tools for IMS 12

This chapter provides an overview of the IBM Information Management System (IMS) Tools portfolio and of the major updates to the tools to support IMS 12. It introduces the new IBM Tools strategy with solution packs, including descriptions and useful information about the documentation, maintenance, end of support (EOS), and compatibility with various IMS versions.

The chapter includes the following sections:

- IMS Tools general information
- Examples of specific product updates for IMS 12 support
10.1 IMS Tools general information

IBM continues to increase its investment in tools to deliver comprehensive solutions that respond to customer requirements faster than ever before. The focus of the investment is on common interfaces, cross-tool integration, and usage testing.

To meet the objectives about common interface and cross-tool integration, in 2010, IBM announced the following Solution Packs:

- IBM Tools Base for z/OS (product number 5655-V93) also called IMS Tools Base for z/OS for version 1.1
- IMS Fast Path Solution Pack for z/OS (5655-W14)
- IMS Database Solution Pack for z/OS (5655-S77)
- IMS Recovery Solution Pack for z/OS (5655-V86)
- IMS Performance Solution Pack for z/OS (5655-S42)

You can still order some of the products included in these packs individually, but most of them are now only available through those packs. Other products, which are not included in packs, can be ordered as before.

This section provides an overview of those tools and the major updates made to some of them to support IMS 12.

10.1.1 Useful links

The information that is available on the Internet is constantly being updated. Check the referenced websites in the following sections periodically to ensure that you have latest version of the information.

**Documentation**
For complete IMS Tools documentation see the IMS Tools Product Page, which is available for DB2 and IMS Tools at:

https://www.ibm.com/support/docview.wss?uid=swg27020942

Select the product name at the top of the page to go directly to the information.

**Product life cycle**
For product life cycle information, see the Product Lifecycle for DB2 and IMS Tools page at:

https://www.ibm.com/support/docview.wss?rs=434&uid=swg27008621

This page includes a table with the following information and more:

- The product number
- The General Availability (GA) date with a link to the announcement letter if you click the GA date
- The End of Marketing (EOM) date with a link to the withdrawal letter
- The End of Support (EOS) date with a link on the service discontinuance announcement
- The replacement product name and version if it is no longer supported

This table is constantly being updated.
Important: No letter of discontinuance of service is sent for One Time Charge products. You have to check this information on this site when it is available.

Figure 10-1 shows an example of the information you can find:

IMS Tools Base V1.1 has no End of Marketing yet, nor End of Support date.
IMS Tools Knowledge Base is End of support on 30 Sept 2011, and is replaced by IMS Tools Base V1.1.

<table>
<thead>
<tr>
<th>IMS Tools Base</th>
<th>Ver/Rel</th>
<th>OS/Platform</th>
<th>PID</th>
<th>GA</th>
<th>EOM</th>
<th>EOS</th>
<th>Replacement Product</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1.0</td>
<td>z/OS</td>
<td>5655-V93</td>
<td></td>
<td>19-Feb-10</td>
<td>TBD</td>
<td>TBD</td>
<td>Not Established</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IMS Tools Knowledge Base</th>
<th>Ver/Rel</th>
<th>OS/Platform</th>
<th>PID</th>
<th>GA</th>
<th>EOM</th>
<th>EOS</th>
<th>Replacement Product</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1.0</td>
<td>z/OS</td>
<td>5655-R34</td>
<td></td>
<td>05-June-97</td>
<td>25-July-10</td>
<td>30-Sep-11</td>
<td>IMS Tools Base v1.1</td>
</tr>
</tbody>
</table>

Figure 10-1  Product life cycle information

**Maintenance**

For a list of available fixes (program temporary fixes (PTFs) and a description of the problems they solve (APARs), see the IBM DB2 and IMS Tools PTF Listing page at:


You can also save this information as a comma-separated value (CSV) file to import the list of APARs into a spreadsheet or database for sorting or downloading. Keep a current level of maintenance on the tools, and check this web page as soon as you experience a problem. You might notice that one APAR solves more than one problem, which makes the time to apply them worth it, especially before starting to use new functions or a new version of IMS.

**Compatibility matrix**

For information about the minimum maintenance level required for IMS Tools to support all the current IMS versions, which are Versions 10, 11, and 12, see the IMS Information Management Tools and IMS for z/OS V11.1 Compatibility page at.

https://www-304.ibm.com/support/docview.wss?rs=434&uid=swg21296180

Check this page before you start a new IMS migration to access the latest tools.

**10.1.2 IMS Tools portfolio**

Figure 10-2 on page 388 shows a complete overview of the products and the application area they belong to. A brief description of the solution packs and some of the products is provided in the sections that follow.

For more information about tools, see the official documentation at websites referenced in the 10.1.1, “Useful links” on page 386.
The following sections describe the tools belonging to the Solutions Packs:

- IBM Tools Base Pack for z/OS Version 1 (5655-V93)
- IMS Fast Path Solution Pack for z/OS (5655-W14)
- IMS Database Solution Pack for z/OS (5655-S77)
- IMS Recovery Solution Pack for z/OS (5655-V86)
- IMS Performance Solution Pack 110 (5655-S42)

They also include a short description of the following tools:

- IMS Recovery Expert for z/OS 210 (5655-S98)
- IMS Sysplex Manager 130 (5655-P01)
- IBM Transaction Analysis Workbench 110 (5697-P37)

### 10.1.3 IBM Tools Base Pack for z/OS Version 1 (5655-V93)

Version 1.1 of this pack has been available until now. Version 1.2 was announced on 9 August 2011 and has been available since 9 September 2011. This new release adds a common interface to some DB2 tools.

**Important:** This pack is the foundation of the other packs, and must be installed as a prerequisite to all the IMS solution packs and most of the IMS tools products.

The tools base pack is a *no charge* product, but is not included in other packs. Therefore, you must order it.

IMS Tools Base for z/OS, V1.1 (5655-V93), now called IBM Tools Base for z/OS Version 1.1, combines several IMS products and the entire collection of IMS Tools common infrastructure components into a single, consolidated installation package.
The IMS Tools Base package provides the following features, functions, and common services:

- IMS Tools Knowledge Base
- IMS Tools Common services
- IMS Tools Distributed Access Infrastructure
- IMS Hardware Compression Extended

This pack replaces:

- IMS Tools Knowledge Base (5655-34; end of support on 30 September 2011)
- IMS Tools Online Interface (5697-N50)

**IMS Tools Knowledge Base**

IMS Tools Knowledge Base (ITKB) provides a common information management service that allows the sharing of data generated by multiple tool products within a sysplex from a single, centralized interface.

This information can be:

- Reports from IMS Tools or third parties
- Sensor data: information captured by an IMS Tools product (such as IMS Reorganization Expert 410) at an instance in time that represents the condition, or state, of one or more databases
- Policies: sets of rules that define threshold limits for specific types of database conditions. These threshold limits are then evaluated against your sensor data.

This data is centralized in a common report repository and can be managed with its viewing interface through a complex sysplex environment. Historical report data can be found through a powerful search, and preserved for future decision making. ITKB is the single interface within a sysplex environment for multiple IMS tools products to share report output.

**IMS Tools Common services**

IMS Tools Common services includes the IMS Tools Generic Exits and the IMS Tools Online System Interface (TOSI) functions. IMS Tools Generic Exits is a set of specialized utilities used by IMS Tools product code to manage various interfaces including the managing of IMS exits. TOSI is a general purpose command interface that allows IMS Tools to interface with all supported IMS versions.

**IMS Tools Distributed Access Infrastructure**

IMS Tools Distributed Access Infrastructure serves as the IMS Administrative Gateway Emphasis on flexibility of communication. It acts as a dispatcher for IMS Tools and can construct an execution environment that typically runs under Time Sharing Option (TSO) commands or batch.

Attention: To support IMS 12, the IMS Tools Base Pack, FMID HAHN110, needs PM21167/UK62373.
IMS Hardware Compression Extended

IMS Hardware Compression Extended is a productivity aid for implementing Hardware Assisted Data Compression. It offers the following benefits:

- It extends IMS basic Hardware Data Compression (HDC) support with utilities that ease compression implementation and provide additional flexibility and function. For more information about standard HDC support, see IMS Version 12 Exit Routines, SC19-3016.
- It works on Image Copies, High Performance Unload files, and IMS Unload files.
- It includes sample Compression Dictionaries.
- It enables the effectiveness of existing dictionaries to be assessed.
- It allows monitoring of dictionary effectiveness over time.
- Builds database description (DBD) and Reload job control language (JCL).

Important: IMS Tools Knowledge Base, IMS Tools Common services, IMS Tools Distributed Access Infrastructure, and IMS Hardware Compression Extended cannot be ordered separately and are only available from IMS Tools base pack or IBM Tools base pack.

IBM Tools Base for z/OS Version 1.2

IBM Tools Base for z/OS, V1.2 (5655-V93) consists of the features, functions, and common services listed previously in V1.1. Additionally, it offers IBM Tools Customizer for z/OS, which is used by various IM tools (IMS and DB2) and which assists in customization by providing an enhanced, streamlined customization process. For more information about this topic, see announcement letter 2011_ENU5211-283.

Attention: IBM Tools Base for z/OS Version 1.2 supports IMS 12 without additional maintenance

10.1.4 IMS Fast Path Solution Pack for z/OS (5655-W14)


This pack includes:

- IMS High Performance Fast Path Utilities
  - IMS Fast Path Advanced Tools
  - IMS Fast Path Basic Tools
  - IMS Fast Path Online Tools
- IMS High Performance Image Copy
- IMS Library Integrity Utilities
- IMS Database Repair Facility

Important: IMS HP Fast Path Utilities (5655-W14) and IMS Database Repair Facility (5655-E03) are only available from the IMS Fast Path Solution Pack and cannot be ordered separately. The old versions of those products are not compatible with IMS 12 and are withdrawn from support on 30 September 2011.

The pack also replaces the old version of Fast Path Utilities product, Fast Path Basic Tools (5655-E30), and Fast Path Online Tools (5655-F78), which are at end of support (EOS) since 2006.
For APAR and PTF numbers, see Table 10-1. Other APARs are available. Be current with the maintenance level by checking the maintenance website. When you order this pack, you receive all the products included in it.

Table 10-1  Fast Path Solution Pack maintenance to support IMS 12

<table>
<thead>
<tr>
<th>Product name</th>
<th>Product number</th>
<th>FMID</th>
<th>APAR/PTF</th>
<th>Abstract</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMS Fast Path Solution Pack 110</td>
<td>5655-W14</td>
<td>HAHQ110</td>
<td>PM21939/UK62565 PM31908/UK65466 PM36509/UK69078 PM37894/UK69302</td>
<td>▶ Preconditioning APAR for IMS 12 DEDB secondary index enhancement Compatibility with LIU for IMS12</td>
</tr>
<tr>
<td>IMS High Performance Image Copy 420</td>
<td>5655-N45</td>
<td>H1J0420</td>
<td>PM21942/UK62577 PM34347/UK66329</td>
<td>▶ Preconditioning APAR for IMS 12 ▶ EAV OSAM IMS 12 support for HPIO</td>
</tr>
<tr>
<td>IMS Library Integrity Utilities 210</td>
<td>5655-U08</td>
<td>H27P210</td>
<td>PM21961/UK62602 PM46494/UK71799</td>
<td>▶ Preconditioning APAR for IMS 12 ▶ New function to support indexed DEDB and Fast Path secondary index with IMS 12</td>
</tr>
</tbody>
</table>

IMS High Performance Fast Path Utilities

It is a challenge to increase system online availability when maintaining databases, while reducing total maintenance time. IMS High Performance Fast Path Utilities (IMS HP FP Utilities) provides you with extensive functions to manage your DEDBs in both online and offline environments. It consists of three tools and supplementary utilities as listed in Figure 10-3, and works with IMS 10, 11, and 12, and z/OS 1.9 or later.

▶ Fast Path Advanced Tool (FPA)
▶ Fast Path Basic Tools (FPB)
▶ Fast Path Online Tools (FPO)
▶ Supplementary utilities

IMS HP FP Utilities provides a complete set of high performance utilities such as unload, reload, reorganize, backup, and verify. It reports on DEDB areas and tunes up the libraries. It also reduces CPU and elapsed time, processes in parallel multiple DEDB areas in a single step, and eliminates I/Os for intermediate data sets by enabling unload, reload, analyze and backup tasks to run in a single step.
IMS HP FP Utilities reduces the unavailability time of taking database image copies using advanced technologies. It provides online reorganize, extract, extend, backup, verify and report functions on “online” DEDB areas, and then improves system availability.

Additionally, IMS HP FP Utilities has been enhanced to support the IMS 12 Fast Path Secondary Index (FPSI) new feature. For information about this support, see 10.2.1, “High Performance Fast Path Utilities updated for IMS 12” on page 410.

**High Performance Image Copy**

IBM High Performance Image Copy (HPIC) 420 is included in the pack. It can be ordered separately with product number 5655-N45, but it comes with the pack if the pack is ordered.

This product is a replacement for IMS Image Copy standard solutions for both full function (including high availability large databases (HALDB)) and fast path DEDB databases, whether you are dealing with a batch or concurrent image copy, either standard or DFSMS COPY image copy.

HPIC works with enhanced capabilities such as:

- Allow Parallel IC processing (copy multiple DBDS at the same time)
- Enable hash pointer checking during image copy (not available with concurrent image copy)
- Support compressed image copy output
- Copy image copies with compression or without compression
- Allow stacking of multiple output data sets on tape
- Dynamic allocation of input and output data sets
- Integration with the IMS Tools Base Pack (stop and start database around creation of batch image copy)

HPIC advanced copy services have the following characteristics:

- They are implemented using DFSMSdss API to invoke DUMP and COPY commands.
- They are based on the concept of logical copy and physical copy.
  - Logical copy takes a short time and the database is available afterward.
  - Physical copy (IC) is created from the logical copy.
- They use a combination of software and hardware:
  - Concurrent Copy support for cached subsystems.
  - SnapShot support for RAMAC Virtual Array (RVA) devices.
  - IBM FlashCopy® support for IBM TotalStorage® Enterprise Storage Server® (ESS) and IBM TotalStorage DS8000 devices (there is similar support for OEM devices).
- They are available for both clean and fuzzy image copy processing.

HPIC includes a High Performance I/O engine for read and write processing, and can automatically restart for failed image copies. It also detects automatically improved data recording capability (IDRC) for output tape.

To summarize, High Performance Image Copy offers the following benefits:

- It reduces copy and recovery time.
- It provides a powerful engine and parallelism capabilities to reduce CPU and elapsed time.
- It allows higher database availability by passing commands to IMS itself.
- It tightly integrates with other tools to give better performance and simplify the copy process.
Library Integrity Utility

Library Integrity Utility (LIU) 210 is included in the pack. It can be ordered separately with product number 5655-U08, but it comes with the pack if the pack is ordered.

LIU helps to efficiently maintain the numerous IMS data sets that contain definitions, such as DBDLIB, PSBLIB, ACBLIB, formats MFS. For example, it ensures that a correct definition is in use to prevent database corruption.

To do this, LIU provides the following utilities:

- **Integrity checker**
  This utility prevents DB corruption caused by using a wrong DBD. A wrong DBD is a DBD that is different from the one used to load the database. This situation can arise if you use an old DBD after a DBD change is applied.

- **Consistency checker**
  This batch utility ensures that the necessary definition in IMS has been created for your database, such as ACB in an ACB library, a Database Definition entry in the MODBLKS module (DFSDDIR), a DFSMDA dynamic allocation member for database data sets in a MDA library, and DB and DSG registration in recovery control (RECON).

- **DBD/PSB/ACB Compare**
  This batch utility reports the difference between DBD, program specification block (PSB), and application control block (ACB) members that have the same name but reside in separate object libraries, whatever their type is (same or different).

- **DBD/PSB/ACB Mapper**
  This batch utility produces a printed map (picture of segment hierarchy) from DBDs, PSBs, and ACBs.

- **DBD/PSB/ACB Reversal**
  This batch utility converts a DBD, PSB, or ACB member back into IMS **DBDGEN** or **PSBGEN** utility control statements.

- **Advanced ACB generator**
  This batch utility provides a high-speed generation process for processing large volumes of IMS ACBs. It replaces the IMS **ACBGEN** utility.

- **MFS reversal**
  This batch utility can compare two MFS load module libraries, and convert the format in MSF source format.

LIU has been enhanced to support the IMS 12 Fast Path Secondary Index new feature. For more information about this topic, see 10.2.2, “Library Integrity Utilities updated for IMS 12” on page 423.

Database Repair Facility

The Database Repair Facility product cannot be ordered separately, and it comes with the Fast Path Solution Pack or with the High Performance Pointer Checker 310. The purpose of this function is to repair IMS databases, Full Function or DEDB, that contain pointer or data errors in interactive mode or in batch mode.
IMS Database Solution Pack provides a complete set of high performance utilities to analyze, maintain, and tune IMS Full Function databases including HALDB.

This pack includes:
- IMS High Performance Image Copy 420
- IMS Library Integrity Utilities 210
- IMS High Performance Pointer Checker 310
- IMS Database Reorganization Expert 410
- IMS High Performance Unload 120
- IMS High Performance Load 210
- IMS Index Builder 310
- IMS High Performance Prefix Resolution 310

The maintenance to support IMS 12 must be applied to individual products for the Database Solution Pack (see Table 10-2). Other APARs are available. Be current with their maintenance levels and check the maintenance website.

When you order this pack, you receive all the products included in it. However, in this case, all the products can be ordered on a unitary basis.

<table>
<thead>
<tr>
<th>Product name</th>
<th>Product number</th>
<th>FMID</th>
<th>APAR/PTF</th>
<th>Abstract</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMS High Performance Image copy 420</td>
<td>5655-N45</td>
<td>H1J0420</td>
<td>PM21942/UK62577</td>
<td>Preconditioning APAR for IMS 12</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>PM34347/UK66329</td>
<td>EAV OSAM IMS 12 support for HPIO</td>
</tr>
<tr>
<td>IMS Library Integrity Utilities 210</td>
<td>5655-U08</td>
<td>H27P210</td>
<td>PM21961/UK62602</td>
<td>Preconditioning APAR for IMS 12</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>PM46494/UK71799</td>
<td>New function to support indexed DEDB and Fast Path secondary index with IMS 12</td>
</tr>
<tr>
<td>IMS High Performance Pointer Checker 310</td>
<td>5655-U09</td>
<td>HPC2310</td>
<td>PM21945/UK62559</td>
<td>Preconditioning APAR for IMS 12</td>
</tr>
<tr>
<td></td>
<td></td>
<td>H22K310</td>
<td>PM25552/UK62558</td>
<td>for the HPPC part or the Data Repair Facility part</td>
</tr>
<tr>
<td>IMS Database Reorganization Expert 410</td>
<td>5655-S35</td>
<td>H25N410</td>
<td>PM22116/UK62553</td>
<td>Preconditioning APAR for IMS 12</td>
</tr>
<tr>
<td>IMS High Performance Unload 120</td>
<td>5655-E06</td>
<td>H1IN120</td>
<td>PM22119/UK62576</td>
<td>Preconditioning APAR for IMS 12</td>
</tr>
<tr>
<td>IMS High Performance Load 210</td>
<td>5655-M26</td>
<td>H1IM210</td>
<td>PM22118/UK62579</td>
<td>Preconditioning APAR for IMS 12</td>
</tr>
<tr>
<td>IMS Index Builder 310</td>
<td>5655-R01</td>
<td>H220310</td>
<td>PM22120/UK62546</td>
<td>Preconditioning APAR for IMS 12</td>
</tr>
<tr>
<td>IMS High Performance Prefix Resolution 310</td>
<td>5655-M27</td>
<td>H1IP310</td>
<td>PM22121/UK62343</td>
<td>Preconditioning APAR for IMS 12</td>
</tr>
</tbody>
</table>

**High Performance Image Copy**

HPIC 420 is included in the Database Solution Pack, or it can be ordered separately with product number 5655-N45. For an overview of this product, see “High Performance Image Copy” on page 392.
Library Integrity Utility
LIU 210 is included in the Database Solution Pack, or can be ordered separately with product number 5655-U08. For an overview of this product, see “Library Integrity Utility” on page 393.

IMS High Performance Pointer Checker
High Performance Pointer Checker (HPPC) 310 is included in the Database Solution Pack, or it can be ordered separately with product number 5655-U09.

HPPC can perform a validation of the database pointers, as explained here:
- It can perform standard checking, known as full checking, by checking the pointers one by one and detecting the place of pointer corruption,
- Or it can perform hash checking, which is a quick check used to determine whether a database is valid or has pointer errors. Hash checking cannot detect the location of pointer corruption, but is fast and consumes less CPU time.

IMS databases: HPPC applies to the following IMS databases:
- HDAM
- PHDAM
- HIDAM and primary index databases
- PHIDAM and primary index databases
- HISAM databases
- HDAM/HIDAM/HISAM secondary index databases
- PHDAM/PHIDAM secondary index databases (PSINDEX)

It does not support IMS Partition DB (5697-A06, 5697-D85) or any other products with an equivalent function.

IMS High Performance Pointer Checker provides the following utilities:

HD Pointer Checker (HDPC)
This is the main component. It scans databases and validates them by checking the relation of pointers and segment starting address. It can perform hash checking or standard checking.

DB Repair Facility
You can change database data directly by using the DB Repair Facility TSO ISPF panel or batch job.

Space Monitor (SPMN):
This utility reports space information of the database data set and monitors the threshold for space information.

HD Tuning Aid (HDTA)
This utility evaluates the distribution of root segments. If the root segments are distributed uniformly, good randomizing parameters are given. If not, it is better to consider different parameters. HDTA can simulate the distribution of root segments as though the parameters or database organization were changed.

DB Historical Data Analyzer (DBHDA)
If you do not want to use the IMS Tools Knowledge Base repository to keep the data history, the statistics data can be stored in a history data set by DBHDA. It can print reports and show a data set usage trend graph on TSO.
**DB Segment Restructure (DBSR)**

This utility allows segment data to be changed with control statements. There is no need to code a complicated application program.

**IMS Database Reorg Expert**

IMS Database Reorg Expert 410 is included in the Database Solution Pack. It can also be ordered separately with product number 5655-S35.

IMS Database Reorg Expert 410 is a complete infrastructure to operate IMS Reorganization-related tools in parallel or independently, thereby allowing a significant reduction in reorganization time.

**Online Reorganization Facility tool:** IMS Database Reorg Expert shortens the reorganization process by executing the various steps in parallel, being fully integrated with other tools. However, the databases are unavailable during this short process. It is not an online reorganization. Its power is the conditional reorganization with the sensor data.

Online Reorganization (OLR) Facility is an IMS function available from IMS 9 giving a truly online reorganization, only available for HALDB. The Online Reorganization Facility tool, product number 5655-H97, provides an almost online reorganization for full function databases including HALDB. At the end of the reorganization, it needs to make the database unavailable for a short time to finish applying the updates, renaming the data sets, and updating the RECON data set.

IMS Database Reorg Expert provides the following utilities:

- Smart Reorg Utility, which includes two features:
  - Conditional Reorganization Support Service (CRSS)
  - Parallel Reorganization Service
- IMS Parallel Reorg Unload utility
- IMS Parallel Reorg Load utility
- IMS Parallel Reorg Database Scan utility

The Smart Reorg utility (with CRSS) evaluates the full function databases and collects sensor data. It is then able to tell whether the database needs to be reorganized, according to policies and statistics stored in the IMS Tools Knowledge Base repository. After this evaluation it can start the reorganization automatically and evaluate the reorganized database to check the benefit of the reorganization. Finally, it provides a report on the database status and the changes operated on it.
Figure 10-4 summarizes the Smart Reorg driver operations.

The IMS Parallel Reorganization utilities provide a high performance engine, running unload and load in parallel in only one step and performing pertinent functions like index building, image copy and hash pointer checking, and prefix resolution and update.

The JCL becomes easy to write and you do not need to have deep skills in IMS to write a reorganization batch with this tool.

**IMS High Performance Unload**

HP Unload is included the pack, or it can be ordered separately with product number 5655-E06. It replaces the standard HD Reorganization Unload utility (DFSURGU0). It is serviced by a high performance database retrieval engine called High Speed Sequential Retrieval (HSSR) Engine, and it reduces CPU and elapsed time.

The heart of HP Unload, the HSSR Engine, can be used by two utility programs, FABHURG1 and FABHFSU. These engine provide the following functions among others:

- Unloading of compressed segments without decompression
- User exit for additional selection, editing of unloaded segments, or both
- Production of statistical reports
- Ability to continue after segment sequence errors
- Ability to read corrupted data bases
- Bypass corrupted pointers
- Force access of HIDAM or PHIDAM roots by using an index
- Diagnostic report about pointer errors

HP Unload has an API compatible with HSSR, which enables DL/I application programs to use HSSR to read a database using GN calls. There is no need to change the application to use the engine; the eligibility is done on the PCB.
Similar to other product tools, HP Unload provides a simple JCL, parallelism, and integration with other tools.

**IMS High Performance Load**

HP Load 210 is included in the pack, or it can be ordered separately with product number 5655-M26. It is a high speed alternative of the IMS HD Reorganization Load utility (DFSURGL0).

IMS HP Load provides a set of high performance reorganization reload procedures for the following database organizations:

- HDAM
- HIDAM
- HISAM
- SHISAM
- PHDAM
- PHIDAM

HP Load includes a Physical Sequence Sort for Reload (PSSR) engine that is used before re-Load to sort the unload data set, or which can be used before migrating to HALDB to sort into partition sequence. It can use disk or data space to load data during its process.

HP Load provides features such as:

- Accepts (compressed) input in *HD or *CS formats or in a user format
- Initialization of empty HDAM and HIDAM DBs
- Loads HALDB Partitions without them first being Initialized
- Provides a set of statistical reports
- Support of logical relationship or secondary indexes
- Creates data sets DFSURWF1 and optionally HPSRSIDX (if you want split work files) that can be used by:
  - IMS Index Builder tool
  - IMS High Performance Prefix Resolution tool
  - IMS Prefix Resolution utility (DFSURG10)

**IMS Index Builder**

IMS Index Builder 310 is included in the pack, or it can be ordered separately with product number 5655-R01. It applies to full function databases including HALDB.

With IMS Index Builder, you can build (or rebuild) the following:

- IMS secondary indexes
- Hierarchical Indexed Direct Access Method (HIDAM) primary indexes
- Indirect List Data Sets (ILDS)
- Supports full-function and partitioned HALDB

IMS Index Builder makes it faster to rebuild indexes instead of creating image copy and recover. It streamlines the process by creating multiple indexes in a single job step, and uses both parallel sort and parallel load whenever more than one index is being built, thus reducing the time needed to build multiple indexes of a single physical database.

Index build can be integrated with the DRF recovery process and reorganization process, which is available in the IMS Recovery Solution Pack. See 10.1.6, “IMS Recovery Solution Pack for z/OS (5655-V86)” on page 399.
IMS High Performance Prefix Resolution

HPPR 310 comes with the pack, or it can be ordered separately with product number 5655-M27. With HPPR 310, you can resolve and update prefixes of IMS databases involved in logical relationships in a single job step.

10.1.6 IMS Recovery Solution Pack for z/OS (5655-V86)

With the IMS Recovery Solution Pack for z/OS, clients can easily implement good practices in backup and recovery scenarios and support all recoverable IMS databases types, full function or DEDB.

This pack includes the following products:

- IMS Database Recovery Facility for z/OS
- IMS Database Recovery Facility Extended Functions for z/OS
- IMS High Performance Change Accumulation Utility for z/OS
- IMS High Performance Image Copy for z/OS 420
- IMS Index Builder for z/OS 310

Important: IMS Database Recovery Facility for z/OS 310, IMS Database Recovery Facility Extended Functions for z/OS 110, and IMS High Performance Change Accumulation Utility for z/OS 140 are only available from this pack.

The old versions of these products are not compatible with IMS 12 and are withdrawn from support on 30 September 2011.

The five products come with the IMS Recovery Solution Pack when ordered.

IMS Recovery Expert for z/OS 110 has been renamed to IMS Data Recovery Facility Extended Functions for z/OS 110. It has been merged with the IMS Database Recovery Facility for z/OS 310 into a single library, and they provide a single interface.

IMS Recovery Expert for z/OS Version 2 is another product available outside this pack. For an overview of this product, see “IMS Recovery Expert for z/OS 210” on page 407.

Table 10-3 lists the for APAR and PTF numbers. Other APARs are available. Be sure to stay current with the maintenance levels by checking the maintenance website. When you order this pack, you receive all the products included in it.

<table>
<thead>
<tr>
<th>Product name</th>
<th>PID</th>
<th>FMID</th>
<th>APAR/PTF</th>
<th>Abstract</th>
</tr>
</thead>
</table>
| IMS Recovery Solution Pack 110        | 5655-V86| HAHM110 | PM23052/UK64046 PM31377/UK64652 PM40317/UK68936 PM40723/UK68989 PM36306/UK69609 | ▶ Preconditioning APAR for IMS 12
|                                       |         |         |                                                                          | ▶ DRF functions compatibility with FP Solution Pack |
| IMS High Performance Image copy 420   | 5655-N45| H1J0420 | PM21942/UK62577 PM34347/UK66329                                         | ▶ Preconditioning APAR for IMS 12
|                                       |         |         |                                                                          | ▶ EAV OSAM IMS 12 support for HPIO             |
| IMS Index Builder 310                 | 5655-R01| H220310 | PM22120/UK62546                                                         | Preconditioning APAR for IMS 12               |
**Rebuilding Fast Path Secondary indexes:** To rebuild Fast Path Secondary indexes, DRF calls the INDEXBLD function that is available in Fast Path Solution Pack if PM36306 is applied. The Fast Path Solution Pack must also have PM37894.

If Fast Path Solution Path is not available, or one or both APARs is not applied, the standard IMS utility is used.

**IMS Database Recovery Facility and DRF Extended functions**

IMS Recovery Solution Pack for z/OS provides a high performance IMS database recovery solution with parallel I/O, parallel processing and sorting, a single archived log and change accumulation data set pass, and a single pass DB write. It can recover multiple DBDS or areas in one step, and it can run in parallel with transaction processing, online or batch mode recovery to *any* time stamp, not restricted to allocation boundary.

Figure 10-5 on page 401 provides a general overview of its functions:

- It can be invoked through an IMS `/RECOVER` command (an online region is needed) or through a batch (no online region needed).
- It uses dynamic allocation of data sets for database, log, change accumulation, and image copy.
- It does *not* use IMS resources for recovery.
- It gets all information about recovery data sets from recovery controls (RECONs).
- It performs pre-recovery actions such as the following examples:
  - The `/DBR` command to take DBDS offline.
  - Deleting and redefining the DBDS when necessary. (It can use the `REUSE` option for Virtual Storage Access Method (VSAM) and can always restore-in-place with overflow sequential access method (OSAM).)
  - Defining the Recovery Scope by building the “Recovery List”.
- It performs recovery actions such as the following examples:
  - Using IMS Tools Base to issue commands to IMS and perform automatic delete/define database.
  - Reading log data sets in parallel.
  - Reading Change Accumulation data sets concurrently with log data sets, complete or incomplete (with spill records).
  - Change Accum is not required, even in a data sharing environment.
  - Sorting log records.
  - Updating Image Copy data with log and change accumulation data in a single pass.
Figure 10-5  Data Recovery Facility at a glance

It can perform auxiliary processing as described here:
- During the recovery phase, such as performing image copies or doing validation pointer checking
- During the post-recovery phase, such as index building and running an IMS Index/ILDS Rebuild utility for HALDB

It runs calling internally High Performance Change Accum to read and sort the change accum records.

**IMS High Performance Image Copy**
HPIC 420 comes with the Recovery Solution Pack, or it can be ordered separately with product number 5655-N45. For an overview of this product, see “High Performance Image Copy” on page 392.

**IMS Index Builder**
IMS Index Builder 310 comes with the Recovery Solution Pack, or it can be ordered separately with product number 5655-R01. For an overview of this product, see “IMS Index Builder” on page 398.

**10.1.7 IMS Performance Solution Pack for z/OS (5655-S42)**

The IMS Performance Solution Pack for z/OS offers an end-to-end performance management and problem determination solution set in an IMS and TCP/IP environment.

This pack includes:
- IMS Problem Investigator (5655-R02)
- IMS Performance Analyzer (5655-R03)
- IMS Connect Extension (5655-S56)
The maintenance to support IMS 12 must be applied to individual products for the Performance Solution Pack. Table 10-4 lists the APAR and PTF numbers.

**Table 10-4  Performance Solution Pack maintenance to support IMS12**

<table>
<thead>
<tr>
<th>Product name</th>
<th>PID</th>
<th>FMI</th>
<th>APAR/PTF</th>
<th>Abstract</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMS Problem Investigator</td>
<td>5655-R02</td>
<td>H28T220</td>
<td>PM24662/UK65183</td>
<td>▶ Preconditioning APAR for IMS 12</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>PM34000/UK65462</td>
<td>▶ Compatibility with CEX</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>PM36967/UK68755</td>
<td>▶ Support for new CEX journal records with IMS 12</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>PM42203/OPEN</td>
<td>▶ Compatibility with CEX</td>
</tr>
<tr>
<td>IMS Performance Analyzer</td>
<td>5655-R03</td>
<td>H23K420</td>
<td>PM24585/UK64657</td>
<td>▶ Preconditioning APAR for IMS 12</td>
</tr>
<tr>
<td>IMS Connect Extension</td>
<td>5655-S56</td>
<td>H28S220</td>
<td>PM32394/UK65339</td>
<td>▶ Preconditioning APAR for IMS 12</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>PM24860/UK22288</td>
<td></td>
</tr>
</tbody>
</table>

**IMS Problem Investigator**

IMS Problem Investigator (PI) 220 comes with the pack, or it can be ordered separately with product number 5655-R02.

IMS PI is an easy tool to use, with minimum setup required. It provides an enhanced level of problem determination services for IMS Transaction Manager (IMS TM), IMS DB, Common Queue Server (CQS), IMS Connect (with IMS Connect Extension (CEX) tool) and others. It can include log record information coming from IBM IMS, CEX, DB2, CQS, System Management Facilities (SMF), and IBM OMEGAMON®.

IMS PI can select dynamically the right log data sets from the RECON information or from information from IMS Connect Extension and format them quickly so that you can easily navigate them.
Figure 10-6 shows an example of the instantaneous view you can have of the log records.
Then, you can select records to drill down right to the values of individual flag bits as shown in Figure 10-7.

IMS Problem Investigator is integrated with IMS Performance Analyzer and IMS Connect Extension. It provides the most benefits to IMS Transaction Manager.

With IMS PI, you can perform the following actions:

- Isolate problems in complex interrelated components in hours instead of weeks. This translates to reduced down-time and lowered staffing costs.
- Dissect your resource utilization, allowing you to optimize your hardware usage and identify unnecessary overhead.
- Enable less experienced staff to perform advanced analysis.
- Map the life-cycle of individual transactions, providing you a better understanding of your environment and the ability to preemptively eliminate problems.

Another product, called *IBM Transaction Analysis Workbench*, extends the capabilities of IMS PI to CICS and offers more z/OS logging information formatting. For an overview of this product, see “IBM Transaction Analysis Workbench” on page 408.

### IMS Performance Analyzer

IMS Performance Analyzer (PA) 420 comes with the pack or it can be ordered separately with product number 5655-R03. IMS PA provides performance reports, through an online interface...
able to create the JCL and to select automatically the logs data sets from the RECON or IMS Connect Extension repository.

Figure 10-8 shows an overview of IMS PA functions at a glance.

![IMS PA at a glance diagram]

The reports provided by IMS PA can help to perform the following tasks:

- **Daily monitoring**, using dashboards for a daily system health check, transit response time reports to help identify problem transactions, or management exception reports, which identify when target service levels are not being met.

- **Long-term capacity planning and service levels**, using the transaction history file, which accumulates transaction performance information, or loading into DB2 to build a performance database and to report on a host or workstation using your favorite SQL reporting tool.

- **Analysis of a performance problem and exploration with other reports**, such as transit reports for bad response time, Resource Utilization reports for IMS resource constraint, or specialized traces to track specific events.

**IMS Connect Extension**

IMS Connect Extension for z/OS Version 2.2 comes with the pack or it can be ordered separately with product number 5655-S56. It enhances and augments the services of the IMS component IMS Connect.

IMS Connect Extension provides event collection and instrumentation for IMS Connect, and many features such as transaction routing and workload balancing, security feature such as ACEE caching for IMS Connect security, dynamic addition, reload, deletion, disabling or enabling of user exits, or even setting the transaction expiration time for OTMA transactions.
Figure 10-9 shows a general view of event processing with IMS Connect Extension.

Figure 10-9   Event instrumentation with IMS Connect Extension
IMS Connect Extension provides a console to manage all your IMS Connect instances and data stores in a single view, from an ISPF interface or a GUI as shown in Figure 10-10.

![GUI operation console](image)

You can see the activity on all systems, cancel sessions, stop or start IMS Connect, data stores or ports, or reload exits. Alternatively, you can also have an individual IMS Connect view. It is also easy to issue IMS Connect commands or IMS commands.

IMS Connect Extension has been updated to support new IMS 12 enhancement such as IMS Connect Multiple Systems Coupling link using TCP/IP or IMS Connect to IMS Connect communication. For more information, see 10.2.3, “IMS Connect Extension updated for IMS 12” on page 424.

IMS Connect Extension is integrated with IMS PI, IMS PA, and IBM Transaction Analysis Workbench.

### 10.1.8 Complementary products tools

The complementary tools presented in the following sections help reduce performance costs and simplify procedures:

- IMS Recovery Expert for z/OS 210
- IMS Sysplex Manager (5655-P01)
- IBM Transaction Analysis Workbench

### IMS Recovery Expert for z/OS 210

The IMS Recovery Expert for z/OS 210 product has Version 2. It is not the same product as IMS Recovery Expert V1, which has been integrated to IMS Recovery Solution Pack and renamed to IMS Database Recovery Facility Extended functions.
IMS Recovery Expert for z/OS 210 is a storage-aware backup and recovery solution. It integrates storage processor fast-replication facilities with IMS backup and recovery operations. It allows instantaneous backups with no application downtime, reduces recovery time, and simplifies disaster recovery procedures while using less CPU, I/O, and storage resources.

IMS Recovery Expert for z/OS 210 offers the following major benefits:

- Backup of the entire IMS systems instantaneously with no application downtime.
- Quick and easy recovery using intelligent recovery managers for local and remote recovery support.
- Effortless IMS backup and recovery management through an easy to use ISPF interface.
- Less CPU, I/O, and tape resource than image copy.
- Automatic backup validation to achieve successful recoveries every time.
- Use of one backup for multiple purposes.
- Transformation of disaster recovery into a disaster restart process, reducing recovery time objectives.
- Proof of compliance with internal or federal regulations.

**IMS Sysplex Manager (5655-P01)**
IMS Sysplex Manager is not included in a pack. You must order it separately with product number 5655-P01. The version to order is 130.

IMS Sysplex Manager offers a real-time management of the IMS sysplex environment. It is not a performance monitor.

IMS Sysplex Manager provides a single point of control to get a single system image of your sysplex through a simplified ISPF user interface. You can get through this interface displays of IMS resources or structures like the shared queue or the IRLM lock structure and you can drill down to display and manage for example lock or message information. You can issue global type1 command or OM type2 commands, z/OS commands such as SVC capture, get some z/OS performance basic information and get a dashboard of key system indicators and threshold monitoring.

IMS Sysplex Manager gives management functions to intercept system exceptions and generates console alerts. It can produce real-time IRLM Long Lock Report, can browse, delete and recover messages on Shared Queues, delete RM resource structure entries and assign affinity for transactions in shared queues environment.

**IBM Transaction Analysis Workbench**
The IBM Transaction Analysis Workbench tool is not included in a pack. You must order it separately with product number 5697-P37. The version to order is 110.

IBM Transaction Analysis Workbench tool is a performance tool, adapted to CICS or IMS Transaction Manager environments. This tool can provide an instantaneous view of z/OS logging information for a unit of work, whether it comes from CICS or IMS, including MQ and DB2 data. The tool is transaction-oriented and does not replace a DB2 log analysis tool such as the IBM product DB2 Log Analysis (product number 5655-T56).
Chapter 10. Tools for IMS 12

10.2 Examples of specific product updates for IMS 12 support

Four products are presented in this section to show how they support IMS 12 new functions: IMS High Performance Fast Path Utilities; Library Integrity Utilities for IMS 12 Fast Path Secondary Index new function; IMS Connect Extension for IMS Connect to IMS Connect communication; and Multiple Systems Coupling link using TCP/IP and IMS PA for miscellaneous system enhancements seen through the reports.
10.2.1 High Performance Fast Path Utilities updated for IMS 12

High Performance Fast Path Utilities is included in “IMS Fast Path Solution Pack”. This part has been updated to support the Fast Path Secondary Index function in IMS 12. For information about this pack, see 10.1.4, “IMS Fast Path Solution Pack for z/OS (5655-W14)” on page 390.

Table 10-5 lists the utilities included in this pack and indicates whether they are available online in Fast Path Online (FPO), or offline in Fast Path Basic (FPB) or Fast Path Advanced (FPA) components.

**Important:** PM21939/UK62545 provides support for the following new functions:

- **FPA INDEXBLD** function efficiently builds a secondary index database. It can also concurrently build multiple secondary index databases for one or more areas of a DEDB.
- **FPA ANALYZE** function is enhanced to verify the integrity of the pointer segments in secondary index databases, in addition to verifying the integrity of all IMS pointer values in DEDB Areas.

For more information about maintenance, see Table 10-5.

<table>
<thead>
<tr>
<th>Solution</th>
<th>Function</th>
<th>FPA (offline)</th>
<th>FPB (offline)</th>
<th>FPO (online)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reorganizing</td>
<td>Unload</td>
<td>UNLOAD</td>
<td>DEDB Unload</td>
<td>Online Data Extract (ODE)</td>
</tr>
<tr>
<td></td>
<td>Reload</td>
<td>RELOAD</td>
<td>DEDB Reload</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Change</td>
<td>CHANGE</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Reorganize</td>
<td>REORG</td>
<td></td>
<td>Online Expert Reorganization (OER)</td>
</tr>
<tr>
<td></td>
<td>Extend</td>
<td></td>
<td></td>
<td>Online Area Extender (OAE)</td>
</tr>
<tr>
<td></td>
<td>Build FP Secondary Index</td>
<td>INDEXBLD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Analyzing</td>
<td>Integrity verification and analysis</td>
<td>ANALYZE -DEDB AREAs -Secondary Index DBs</td>
<td>DEDB Pointer Checker -DEDB AREAs</td>
<td>Online Pointer Checker (OPC) -DEDB AREAs</td>
</tr>
<tr>
<td></td>
<td>Print DMAC information</td>
<td>DMACPRT</td>
<td></td>
<td>Online DMAC Print (ODM)</td>
</tr>
<tr>
<td>Extracting</td>
<td>Extract segments</td>
<td>EXTRACT</td>
<td>Use Reload/Reload user exit routines</td>
<td>Online Data Extract (ODE)</td>
</tr>
<tr>
<td>Other tool</td>
<td>Tuning Aid</td>
<td></td>
<td>DEDB Tuning Aid</td>
<td></td>
</tr>
</tbody>
</table>

**Fast Path Secondary Index support**

In FPA, the FPSI function provides build and analyze capabilities for the Fast Path Secondary Index databases that are supported in IMS V12. It supports IMS V12 Fast Path Secondary Index databases with the following features:

- HISAM/SHISAM Secondary Index databases
- Unique Key/Non-Unique Key
- Sparse Indexing
- Symbolic Pointer
- Multiple Secondary Index Segments
- Partition Selection
For information about the Fast Path Secondary Index function in IMS 12, see 3.3.4, “Fast Path DEDB secondary index support” on page 78.

**FPA INDEXBLD function**

The INDEXBLD function of FPA component is a new function brought by PM21939. It efficiently builds one or more secondary index databases. It concurrently scans multiple DEDB areas and builds multiple secondary indexes databases in a single job step.

IMS and the FPA INDEXBLD function support two database structures for Fast Path secondary indexes: HISAM and SHISAM.

Both secondary index database structures offer sequential key access to primary DEDB databases. This function builds the specific secondary index databases in case of system failures, rebuilds secondary index databases faster than initially loading data to a DEDB, and reduces the amount of time that it takes to build multiple secondary index databases using both parallel sorting and parallel loading.

Moreover the FPA INDEXBLD function provides two new reports, a “Secondary Index Definition Report” and a “Secondary Index Processing Report.”

**New INDEXBLD command in HFPSYSIN**

FPA runs the Build Index function as a standard z/OS batch job. You need to specify an EXEC statement and DD statements that define the input and output data sets in your JCL.

Specify the EXEC statement in the following format:

```
//INDEXBLD EXEC PGM=HFPMAIN0,REGION=rrrrM,
// PARM="DBRCGRP=dbrcgrp,GSGNAME=gsgname,IMSPLEX=imsplex"
```

For more information about the parameters, see *IBM Fast Path Solution Pack for z/OS V1R1, IMS High Performance Fast Path Utilities User’s Guide*, SC19-2914.

The INDEXBLD command has been added in sysin HFPSYSIN, as shown in Example 10-1.

**Example 10-1  JCL for the INDEXBLD command**

```
//**************************************
//** BUILDING SECONDARY INDEX DB(S) **
//**************************************
//INDEXBLD EXEC PGM=HFPMAIN0
//*
//STEPLIB DD DISP=SHR,DSN=HPFP.SHFPLMD0
// DD DISP=SHR,DSN=IMSVS.SDFSRESL
//IMSACB DD DISP=SHR,DSN=IMSVS.ACBLIB
//HFPPRINT DD SYSOUT=A
//HFPRPTS DD SYSOUT=A
//*
//HFPSYSIN DD *
GLOBAL DBRC=NO
INDEXBLD
DBD=IVPDB3
/*
//DFSIVD3A DD DISP=OLD,DSN=IMSVS.DFSIVD31
//DFSIVD3B DD DISP=OLD,DSN=IMSVS.DFSIVD33
```
The **INDEXBLD** command has the parameters shown in Figure 10-12.

![INDEXBLD command syntax](image)

**Figure 10-12  INDEXBLD command syntax**

Only the **INDEXDBD** parameter is new. The other parameters were still existing for other functions and have the same meaning:

- **DBD** identifies the DBD that contains the areas that are to be processed.
- **INDEXDBD** specifies the DBD for building or processing secondary index databases.

**Two user scenarios with FPA INDEXBLD function**

You can run the FPA **INDEXBLD** function in several ways. The provided user scenarios show some of the typical ways that you can use. By studying and understanding these scenarios, you can learn the technique to use and to effectively run FPA.

**FPA INDEXBLD user scenario 1**

Scenario 1 builds all FPSI databases of DEDB areas when the secondary indexes are defined against the existing DEDB (Figure 10-13).

![FPA INDEXBLD user scenario 1](image)

**Figure 10-13  FPA INDEXBLD user scenario 1**
To run this scenario, you complete the following steps:

1. Run the **DBDGEN**, **PSBGEN**, or **ACBGEN** utility for the DEDB and FPSI databases.
2. Define VSAM data sets for FPSI databases.
3. Change the access of DEDB to “Read Only” or take DEDB offline.
4. Run the FPA **INDEXBLD** function with a JCL (see Example 10-2).

   **Example 10-2  FPA INDEXBLD user scenario 1 JCL**

   ```
   //HFP      EXEC PGM=HFPMAIN0
   //STEPLIB  DD DISP=SHR,DSN=HPFP.SHFPLMD0
   //         DD DISP=SHR,DSN=IMSVS.SDFSRESL
   //         DD DISP=SHR,DSN=IMSVS.PGMLIB
   //IMSACB   DD DISP=SHR,DSN=IMSVS.ACBLIB
   //IMSDALIB DD DISP=SHR,DSN=IMSVS.MDALIB
   //IMS      DD DISP=SHR,DSN=IMSVS.DBDLIB
   //HFPSYSIN DD *
   GLOBAL DBRC=YES
   INDEXBLD DBD=DEDBJN22,ITASKCTL=4,
               IAREA=ALL,INDEXDBD=ALL
   /*
   
   5. If step 3 is “change the access of DEDB to “Read Only,” take DEDB offline.
6. Make an online change to use the new ACBLIB.
7. Start the DEDB and FPSI databases for IMS online access.

FPSI data set names are identified with the member in the DFSMDA library. You can also specify in the JCL DD statements.

The **NOTIFY.REORG** command is issued for FPSI databases.

**FPA INDEXBLD user scenario 2**

Scenario 2 builds some FPSI databases of DEDB areas when some broken FPSI databases are detected (Figure 10-14).

![Diagram of FPA INDEXBLD user scenario 2](image-url)
To run this scenario, you complete the following steps:

1. Run the **FPA ANALYZE** command to detect some FPSI databases are broken. See “FPA ANALYZE function” on page 416.
2. Stop some broken FPSI databases.
3. Delete and define VSAM data sets for FPSI databases.
4. Change the access of DEDB to “Read Only” or take DEDB offline.
5. Run the **FPA INDEXBLD** function with a JCL (see Example 10-3).

**Example 10-3**  FPA INDEXBLD scenario 2 JCL

```shell
//HFP      EXEC PGM=HFPMAIN0
//STEPLIB  DD DISP=SHR,DSN=HPFP.SHFPLMD0
//       DD DISP=SHR,DSN=IMSVS.SDFSRESL
//       DD DISP=SHR,DSN=IMSVS.PGMLIB
//IMSACB   DD DISP=SHR,DSN=IMSVS.ACBLIB
//IMSDLIB  DD DISP=SHR,DSN=IMSVS.MDALIB
//IMS      DD DISP=SHR,DSN=IMSVS.DBDLIB
//HFPSYSIN DD *
GLOBAL DBRC=YES
INDEXBLD DBD=DEDBJN22,ITASKCTL=4,
          IAREA=ALL,INDEXDBD=(IDXDB1,IDXDB2)
/*
```

6. Start the DEDB and FPSI databases for IMS online access.

FPSI data set names are identified with the member in the DFSMDA library or can be identified by JCL DD statements.

The **NOTIFY.REORG** command is issued for FPSI databases.

If a DEDB area to build the broken FPSI databases is identified, you can specify its name in the **HFPSYSIN IAREA** parameter.
Reports provided by FPA INDEXBLD function

Figure 10-15 and Figure 10-16 shows examples of the two reports provided by the FPA INDEXBLD function. Figure 10-15 shows a secondary index definition report.

IMS HPFP UTILITIES - FPA INDEXBLD
5655-W14 V1R1
- PRIMARY DEDB NAME : DEDB21
- NUMBER OF SEC. INDEXES : 1

DEFINITION OF SEC. INDEX (XDFLD) : XNAME7 (TARGET SEG.: ROOTSEG1, SOURCE SEG.: DD1)

ATTRIBUTES:
- DB ORGANIZATION : HISAM
- POINTER SEG. NAME : DD1X
- NULLVAL : X'E8'
- MULTISEG : NO
- INDEX MAINTENANCE EXIT ROUTINE : INDEXXIT
- PARTITION SELECTION EXIT ROUTINE : DBFPSEYM
- PARTITION SELECTION OPTION : SNGL

SEC. INDEX DBD/DDNAMES:

DEDB DBD

ROOTSEG SEGM NAME=ROOTSEG1, PARENT=0, BYTES=(50,20)
ROOTFLD1 FIELD NAME=(ROOTKEY1,SEQ,U), BYTES=12, START=3, TYPE=C
....
LCHILD NAME=(DD1X,(DEDBGS25,DEDBGS26,DEDBGS27,DEDBGS28)),
PTR=SYMB
XDFLD NAME=XNAME7,SRCH=DD1F1,PSELRTN=DBFPSEYM,
SEGMENT=DD1, PSELOPT=SNGL,NULLVAL=C'Y', EXTRTN=INDEXXIT

Figure 10-15   Secondary index definition report

Figures 10-15 shows a secondary index definition report.

IMS HPFP UTILITIES - FPA INDEXBLD
5655-W14 V1R1
- PRIMARY DEDB NAME : DEDB21
- NUMBER OF SEC. INDEX DATABASE: 6(PARTITION INCLUDED)

SEC. INDEX DATABASE INFORMATION:

DEDB DBD

By NULLVAL or Index maintenance exit

Figure 10-16 shows a secondary index processing report.
FPA ANALYZE function

The FPA ANALYZE function has been enhanced with APAR PM21939 to verify the integrity of the pointer segments in secondary index databases. It verifies DEDB areas and their secondary index databases at the same time.

The following data of pointer segments are verified:

- Delete byte
- Search
- Subsequence
- Duplicate data
- Concatenated key

When errors are detected, the FPA ANALYZE function issues error messages and shows the errors in a report. It provides two new reports:

- A secondary index analysis report

The secondary index analysis report shows the number of verified pointer segments in each secondary index database, as you can see in Figure 10-17.

---

**Figure 10-17  Secondary Index Analysis Report**

- PRIMARY DEDB NAME : DEDBJN21
- NUMBER OF SEC. INDEX DATABASE : 6 (PARTITION INCLUDED)

SEC. INDEX DATABASE INFORMATION:

<table>
<thead>
<tr>
<th>PARTITION</th>
<th>DSNAME</th>
<th>DDNAME</th>
<th>TYPE</th>
<th>DSNAME</th>
<th>ALLOC</th>
<th>POINTER SEGMENTS</th>
<th>SOURCE SEGMENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEDBG21</td>
<td>EASY02.SHINO.IMS12.821KSDS</td>
<td>GS21KSDS</td>
<td>JCL</td>
<td>EASY02.SHINO.IMS12.821KSDS</td>
<td>150,000</td>
<td>240,000</td>
<td>390,000</td>
</tr>
<tr>
<td>GS21ESDS</td>
<td>JCL</td>
<td>EASY02.SHINO.IMS12.821KSDS</td>
<td>390,000</td>
<td>390,000</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Error!

---

No. of segments in sec.index

No. of segments in area

![Error mark](Error_mark.png)
A pointer segment dump report, which is available if you install APAR PM47338

The pointer segment dump report, shown on Figure 10-18, provides the pointer segment information by categorizing the errors based on the following:

- THE FOLLOWING POINTER SEGMENTS ARE NOT FOUND
- THE TARGET SEGMENTS THAT ARE POINTED BY THE FOLLOWING POINTER SEGMENTS ARE NOT FOUND

Figure 10-18  Pointer Segment Dump Report
Figure 10-19 shows the syntax of the **ANALYZE** command. **IAREA**, **IDXTASCTL** and **INDEXDBD** are new parameters. **REPORT** and **THRESHOLD** are new subcommands. For more information about the parameters, see *IBM Fast Path Solution Pack for z/OS V1R1, IMS High Performance Fast Path Utilities User's Guide*, SC19-2914.

You can use the **FPA ANALYZE** function in a lot of scenarios. Three of them are presented in the following sections.
**FPA ANALYZE user scenario 1**

Scenario 1 verifies pointer segments between DEDB areas and all FPSI databases (Figure 10-20).

![Image of database diagram](image)

**Figure 10-20  FPA ANALYZE user scenario 1**

To run this scenario, complete the following steps:

1. Change the access of DEDB to “Read Only” or take DEDB offline.
2. Run the **FPA ANALYZE** function with a JCL similar to Example 10-4.

### Example 10-4  JCL for FPA ANALYZE function user scenario 1

```plaintext
//HFP      EXEC PGM=HFPMAIN0
//STEPLIB  DD DISP=SHR,DSN=HPFP.SHFPLMD0
//         DD DISP=SHR,DSN=IMSVS.SDFSRESL
//         DD DISP=SHR,DSN=IMSVS.PGMLIB
//IMSACB   DD DISP=SHR,DSN=IMSVS.ACBLIB
//IMSDLIB  DD DISP=SHR,DSN=IMSVS.MDALIB
//HFPSYSIN DD *
GLOBAL DBRC=YES
ANALYZE
   DBD=DED8JN22,PTRCHKLVL=FULL,
      IAREA=ALL,INDEXDBD=ALL
REPORT
/*
```
The **FPA ANALYZE** function with a JCL verifies DEDB areas and pointer segments for FPSI as described in Figure 10-21.

3. Start DEDB for IMS online access.

**FPA ANALYZE user scenario 2**

Scenario 2 verifies pointer segments between a specified DEDB area and all FPSI databases as shown in Figure 10-22.
To run this scenario, complete the following steps:

1. Take a specified DEDB area offline.

2. Run the **FPA ANALYZE** function for a specified DEDB area with a JCL similar to Example 10-5 on page 421.

   **Example 10-5  JCL for FPA ANALYZE function user scenario 2**

   ```plaintext
   //HFP      EXEC PGM=HFPMAIN0
   //STEPLIB  DD DISP=SHR,DSN=HPFP.SHFPLMD0
   //         DD DISP=SHR,DSN=IMSVS.SDFSRESL
   //         DD DISP=SHR,DSN=IMSVS.PGMLIB
   //IMSACB    DD DISP=SHR,DSN=IMSVS.PGMLIB
   //IMSDLIB   DD DISP=SHR,DSN=IMSVS.MDALIB
   //HFPSYSIN DD *
   GLOBAL DBRC=YES
   ANALYZE
   DBD=DEDBJN22,PTRCHKLVL=FULL,
   IAREA=AREA1,INDEXDBD=ALL
   */
   
   The **FPA ANALYZE** function with JCL verifies DEDB area and pointer segments for FPSI databases as shown in Figure 10-23.

3. Start a specified DEDB area for IMS online access.

---

Figure 10-23  **FPA ANALYZE** verification in user scenario 2
**FPA ANALYZE user scenario 3**

Scenario 3 verifies all DEDB areas and a specified secondary index (Figure 10-24).

![Figure 10-24  FPA ANALYZE function user scenario 3](image)

To run this scenario, complete the following steps:

1. Process FPA INDEXBLD scenario 2 as explained in “FPA INDEXBLD user scenario 2” on page 413.
2. Change the access of DEDB to “Read Only” or take DEDB offline.
3. Run the FPA ANALYZE function to confirm FPA indexbld result with a JCL similar to Example 10-6.

**Example 10-6  JCL for FPA ANALYZE function user scenario 3**

```plaintext
//HFP      EXEC PGM=HFPMAIN0
//STEPLIB  DD DISP=SHR,DSN=HPFP.SHFPLMD0
//        DD DISP=SHR,DSN=IMSVS.SDFSRESL
//        DD DISP=SHR,DSN=IMSVS.PGMLIB
//IMSACB   DD DISP=SHR,DSN=IMSVS.ACBLIB
//IMSDLALIB DD DISP=SHR,DSN=IMSVS.MDALIB
//HFPSYSIN DD *
GLOBAL DBRC=YES
ANALYZE
   DBD=DEDBJN22,PTRCHKLVL=FULL,
   IAREA=ALL,INDEXDBD=(IDXDB1,IDXDB2)
/```

**Tip:** If areas are identified, you can specify only identified area names in the IAREA parameter.
It verifies DEDB areas and verifies pointer segments for FPSI database as shown in Figure 10-25.

Figure 10-25  FPA ANALYZE function verification in user scenario 3

4. Start DEDB for IMS online access.

10.2.2  Library Integrity Utilities updated for IMS 12

LIU is part of the Fast Path Solution Pack and the Database Solution Pack, or can be ordered separately. For a brief description of the basic functions, see “Library Integrity Utility” on page 393, or see IMS Library Integrity Utilities for z/OS, V2R1, User's Guide, SC19-2479. LIU supports IMS 12 new functions with APARs PM21961 and PM46494. For more information, see Table 10-1 on page 391.

Fast Path Secondary Index support
LIU supports Fast Path Secondary indexes as follows:

- Integrity Checker
  - It verifies that the correct DMB control block is used for indexed DEDB and Fast Path secondary index database definition.
  - It verifies that a correct user partition selection exit, specified by DEDB with a user data partition group of the FP secondary index, is used.
  - LIU creates RDEs when the INDEXBLD function of Fast Path Advanced Tool (FPA) of IMS High Performance Fast Path Utilities succeeds. The RDE or Registered DMB Entry is the information about the DMB that IMS used to load the database.
  - It restores RDEs when IMS Database Recovery Facility (DRF) of IMS Recovery Solution Pack performs a time-stamp recovery of FP secondary index and its primary DEDB area.
- Consistency Checker
  It verifies the consistency of indexed DEDB and FP secondary index database definition and referencing PSBs.

- DBD/PSB/ACB Compare
  It reports differences between two control blocks (DBDs, PSBs, or ACBs) of indexed DEDB and FP secondary index database definition and referencing PSBs.

- DBD/PSB/ACB Mapper
  It produces a map and reports of indexed DEDB and FP secondary index database definition and referencing PSBs.

- DBD/PSB/ACB Reversal
  It converts DBD/PSB/ACB control blocks of indexed DEDB and FP secondary index database definition and referencing PSBs back into IMS DBDGEN and IMS PSBGEN utility control statements.

- Advanced ACBGEN
  It generates ACB members of indexed DEDB and FP secondary index database definition.

- ACBLIB Analyzer
  It analyzes ACB members of indexed DEDB and FP secondary index database definition.

10.2.3 IMS Connect Extension updated for IMS 12

IMS Connect Extension has been updated to support IMS 12. Its new capabilities include:

- Recording of events and real-time statistics related to new IMS Connect Multiple Systems Coupling (MSC) TCP/IP link (see 6.2, “Multiple Systems Coupling using TCP/IP” on page 170) and IMS Connect to IMS Connect communication (see 6.1, “OTMA support for asynchronous IMS to IMS communications” on page 162).

- Graphical user interface enhancements to support IMS V12, including support for new type-2 commands and enhanced monitoring of workloads. See 6.4, “IMS Connect type-2 Single Point of Control commands” on page 195.

This support comes with APAR PM24860 and PM32394.

IMS Connect Extension captures all the IMS Connect log records related to the new workloads supported by IMS 12. This way you can analyze activity in IMS Performance Analyzer and IMS Problem Investigator or IBM Transaction Analysis Workbench.

Examining MSC TCP/IP transactions

Having recorded new MSC events in IMS Connect Extension, you can analyze and isolate problems for these MSC transactions in IMS Problem Investigator. IMS Problem Investigator provides a log browser that you can use to merge logs from multiple IMS systems, incorporating the information collected by IMS Connect Extension.

By merging these log data sources together, you can track and view MSC activity from across multiple systems and identify latencies. For example, are they within the processing of a given IMS, or are these latencies in the transmission of messages between IMS systems?
The IMS Problem Investigator panel (Figure 10-26) shows MSC activity in two IMS systems, and two IMS Connect systems, including the life cycle of the transaction.

Figure 10-26   MSC transaction life cycle under IMS PI

1. The first IMS system begins processing the message (Origin IMS OLDS).
2. It is picked up by the local IMS Connect system (Origin CEX journal).
3. The local IMS Connect sends a message to the remote IMS Connect (Origin CEX journal).
4. The remote IMS Connect begins processing the message (Target CEX journal).
5. The message is sent to the remote IMS system (Target IMS OLDS).

Notice that each step described here is captured in a separate log file, but IMS Problem Investigator provides a time-sequence merged view of this activity.

IMS Problem Investigator tracking can further connect the log records on the local and remote IMS systems. With tracking, you can filter out log records that are not part of a single logical
transaction. By using tracking you can see only those log records for a given transaction, but from both IMS systems. You can then use the merged view to display event latencies, which can be used to identify the source of delays: local or remote IMS, or IMS Connect communication.

When you have identified the locality of the problem, you can drill down to view individual log records. Figure 10-27 shows that IMS Connect Extension journal events have been modified to include new information related to MSC TCP/IP.

```
Figure 10-27  Detail of a log record with MSC connection information
```
1. Exit name, in this case, is the new MSC IMS Connect message exit.
2. This is the partner ID that ties the two systems together.
3. This is the system-generated identifier for the client IMS Connect system.
4. This is the IMS ID of the remote client using the IMS Connect system to make the connection.
5. This is the physical link to which this logical link is assigned.

You can also drill down to individual fields, as shown in Figure 10-28.

```
+0024  IRM_ARCH... 04  Architecture Level
  Off  IRM_ARCH0......... 00  Pre Reroute support
  Off  IRM_ARCH1......... 01  Support - IRM_REROUT_NM/
                                  IRM_RT_ALTCID to be in the
                                  transmitted IRM
  Off  IRM_ARCH2......... 02  Support - IRM_REROUT_NM
                                  - IRM_RT_ALTCID
                                  - IRM_TAG_ADAPT
                                  - IRM_TAG_MAP to be in the
                                  transmitted IRM
  Off  IRM_ARCH3.......... 03  Support for SYNCH CALLOUT CORTKN -
                                  includes all previous fields
  On   IRM_ARCH4.......... 04  ICON to ICON support including MSC -
                                  includes all previous fields
```

*Figure 10-28  Detail of a field showing ICON to ICON support*
View version 12 workloads in the GUI

Figure 10-29 shows some of the statistics that are available through the IMS Connect Extension GUI.

Figure 10-29 shows a merged view of activity for a logical collection of IMS systems. With the statistics provided for each message exit, you can view message rates and volumes for new workload types. In addition to viewing summary statistics, you can view in-flight sessions, including the communication between IMS Connect systems in a single aggregated view.

10.2.4 IMS Performance Analyzer updated for IMS 12

IMS PA has been updated to reflect the write-ahead data set (WADS) information in the Internal Resource Usage Report (IRUR).

The standard IMS PA transit analysis reports that provide response time breakdown and resource usage have not changed for IMS V12, but it is worthwhile recapping other recent enhancements in case you are not aware of them.
Logical logger

WADS processing in IMS V12 has been improved as explained in 2.5, “IMS logger” on page 41. The instrumentation in the type 4507 statistics record has also improved. The Logical Logger report in the Internal Resources Usage Report is enhanced to report WADS I/O times (Figure 10-30).

![Logical Logger Statistics in IRUR report with IMS 12](image)

**List of transactions**

The report in Figure 10-31 shows a list of transactions with their performance characteristics. Notice the database I/O and locking times on the right side of the report. They provide insight into how application database calls are performing.

![List of transactions report](image)

**Transaction summary**

The summary report is based on the summary report form. The form is customized to display the information you need to see in the report. Notice the statistical functions, in particular the new RANGE function that reports the percentage of transactions exceeded the specified threshold.
Figure 10-32 shows the sample report generated by the form.

<table>
<thead>
<tr>
<th>Transcode</th>
<th>Count</th>
<th>Avg CPU Time</th>
<th>Avg InputQ Time</th>
<th>Avg Process Time</th>
<th>Avg DB Call Time</th>
<th>Avg DB IO Time</th>
<th>Avg DB Lock Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEBIT</td>
<td>77</td>
<td>0.071474</td>
<td>0.142351</td>
<td>2.60%</td>
<td>1.926484</td>
<td>0.449301</td>
<td>3.90%</td>
</tr>
<tr>
<td>CREDIT</td>
<td>76</td>
<td>0.064885</td>
<td>0.110116</td>
<td>1.32%</td>
<td>1.336226</td>
<td>0.488499</td>
<td>6.58%</td>
</tr>
<tr>
<td>INQUIRY</td>
<td>61</td>
<td>0.068369</td>
<td>0.055915</td>
<td>0.00%</td>
<td>0.576881</td>
<td>0.416315</td>
<td>1.64%</td>
</tr>
<tr>
<td>ORDER1</td>
<td>75</td>
<td>0.069100</td>
<td>0.290177</td>
<td>0.00%</td>
<td>0.935750</td>
<td>0.439957</td>
<td>2.67%</td>
</tr>
<tr>
<td>ORDER2</td>
<td>81</td>
<td>0.067513</td>
<td>0.158074</td>
<td>3.70%</td>
<td>1.696349</td>
<td>0.521486</td>
<td>8.64%</td>
</tr>
<tr>
<td>ORDER3</td>
<td>77</td>
<td>0.010437</td>
<td>0.003081</td>
<td>0.00%</td>
<td>0.057456</td>
<td>0.063084</td>
<td>0.00%</td>
</tr>
<tr>
<td>ORDER4</td>
<td>73</td>
<td>0.014389</td>
<td>0.003081</td>
<td>0.00%</td>
<td>0.142246</td>
<td>0.066195</td>
<td>0.00%</td>
</tr>
<tr>
<td>STOCK</td>
<td>56</td>
<td>0.013033</td>
<td>0.006077</td>
<td>0.00%</td>
<td>0.352568</td>
<td>0.007178</td>
<td>0.00%</td>
</tr>
</tbody>
</table>

From the report we can determine that 6.5% of CREDIT transactions took longer than one second to process. Also, some of the average and maximum database I/O times seem high, especially because so few DLI DB calls were issued.
Environment description

This appendix provides information about the configuration that we used for IBM Information Management System (IMS) 12, IMS Connect, and the IMS repository during this project.

This appendix includes the following sections:
- Hardware and z/OS configuration
- IMS configuration
- IMS repository
- IMS Connect
- Shared queues
### A.1 Hardware and z/OS configuration

To run our IMS 12 system we allocated two z/OS logical partitions (LPARs) and two integrated coupling facility LPARs on an IBM System z9 2094-S18 EC machine.

Sysplex SC63 had two general use central processors (CPs), two System z9 Integrated Information Processors (IBM zIIPs) and two System z Application Assist Processors (zAAPs). SC64 had four general use CPs, one zIIP and one zAAP. Example A-1 shows the LPAR configuration on z/OS.

**Example A-1  z/OS LPAR configuration**

<table>
<thead>
<tr>
<th>SC63 RESPONSES ---------------------------------------</th>
</tr>
</thead>
<tbody>
<tr>
<td>IEE174I 14.08.46 DISPLAY M 622</td>
</tr>
<tr>
<td>PROCESSOR STATUS</td>
</tr>
<tr>
<td>ID CPU                SERIAL</td>
</tr>
<tr>
<td>00 +                   04991E2094</td>
</tr>
<tr>
<td>01 +                   04991E2094</td>
</tr>
<tr>
<td>02 +A                  04991E2094</td>
</tr>
<tr>
<td>03 +A                  04991E2094</td>
</tr>
<tr>
<td>04 +I                  04991E2094</td>
</tr>
<tr>
<td>05 +I                  04991E2094</td>
</tr>
</tbody>
</table>

CPC ND = 002094.S18.IBM.02.00000002991E  
CPC SI = 2094.710.IBM.02.000000000002991E  
Model: S18  
CPC ID = 00  
CPC NAME = SCZP101  
LP NAME = A04  
LP ID = 4  
CSS ID = 0  
MIF ID = 4

+ ONLINE - OFFLINE . DOES NOT EXIST W WLM-MANAGED  
N NOT AVAILABLE  
A APPLICATION ASSIST PROCESSOR (zAAP)  
I INTEGRATED INFORMATION PROCESSOR (zIIP)  
CPC ND CENTRAL PROCESSING COMPLEX NODE DESCRIPTOR  
CPC SI SYSTEM INFORMATION FROM STSI INSTRUCTION  
CPC ID CENTRAL PROCESSING COMPLEX IDENTIFIER  
CPC NAME CENTRAL PROCESSING COMPLEX NAME  
LP NAME LOGICAL PARTITION NAME  
LP ID LOGICAL PARTITION IDENTIFIER  
CSS ID CHANNEL SUBSYSTEM IDENTIFIER  
MIF ID MULTIPLE IMAGE FACILITY IMAGE IDENTIFIER  

<table>
<thead>
<tr>
<th>SC64 RESPONSES ------------------------------------</th>
</tr>
</thead>
<tbody>
<tr>
<td>IEE174I 14.08.46 DISPLAY M 227</td>
</tr>
<tr>
<td>PROCESSOR STATUS</td>
</tr>
<tr>
<td>ID CPU                SERIAL</td>
</tr>
<tr>
<td>00 +                   06991E2094</td>
</tr>
<tr>
<td>01 +                   06991E2094</td>
</tr>
<tr>
<td>02 +                   06991E2094</td>
</tr>
<tr>
<td>03 +                   06991E2094</td>
</tr>
<tr>
<td>04 +A                  06991E2094</td>
</tr>
<tr>
<td>05 +I                  06991E2094</td>
</tr>
<tr>
<td>06 -                    -</td>
</tr>
<tr>
<td>08 -A                   -</td>
</tr>
<tr>
<td>09 -I                   -</td>
</tr>
</tbody>
</table>
Appendix A. Environment description

We also used two interface control checks (IFCC) coupling facility LPARs on the same machine (Example A-2).

Example A-2  Coupling facility configuration

IXL150I  14.12.51  DISPLAY CF 713
COUPLING FACILITY 002094.IBM.02.00000002991E
PARTITION: 0F  CPCID: 00
CONTROL UNIT ID: FFF5
NAMED CF1

COUPLING FACILITY SPACE UTILIZATION

<table>
<thead>
<tr>
<th>ALLOCATED SPACE</th>
<th>DUMP SPACE UTILIZATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>STRUCTURES:</td>
<td>271360 K</td>
</tr>
<tr>
<td>DUMP SPACE:</td>
<td>2048 K</td>
</tr>
<tr>
<td>FREE SPACE:</td>
<td>1732608 K</td>
</tr>
<tr>
<td>TOTAL SPACE:</td>
<td>2006016 K</td>
</tr>
<tr>
<td>MAX REQUESTED DUMP SPACE:</td>
<td>0 K</td>
</tr>
<tr>
<td>VOLATILE:</td>
<td>YES</td>
</tr>
<tr>
<td>CFLEVEL:</td>
<td>15</td>
</tr>
<tr>
<td>CFCC RELEASE 15.00, SERVICE LEVEL 02.12</td>
<td></td>
</tr>
</tbody>
</table>
BUILT ON 09/14/2010 AT 13:00:00
COUPLING FACILITY HAS 0 SHARED AND 1 DEDICATED PROCESSORS
DYNAMIC CF DISPATCHING: OFF

CF REQUEST TIME ORDERING: REQUIRED AND ENABLED

COUPLING FACILITY SPACE CONFIGURATION

<table>
<thead>
<tr>
<th>IN USE</th>
<th>FREE</th>
<th>TOTAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONTROL SPACE:</td>
<td>273408 K</td>
<td>1732608 K</td>
</tr>
<tr>
<td>NON-CONTROL SPACE:</td>
<td>0 K</td>
<td>0 K</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SENDER PATH</th>
<th>PHYSICAL</th>
<th>LOGICAL</th>
<th>CHANNEL TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>ONLINE</td>
<td>ONLINE</td>
<td>ICP</td>
</tr>
<tr>
<td>D3</td>
<td>ONLINE</td>
<td>ONLINE</td>
<td>ICP</td>
</tr>
</tbody>
</table>

COUPLING FACILITY SUBCHANNEL STATUS

TOTAL: 14 IN USE: 14 NOT USING: 0 NOT USBABLE: 0
OPERATIONAL DEVICES / SUBCHANNELS:
FF1C / 5018  FF1D / 5019  FF1E / 501A  FF1F / 501B
FF20 / 501C  FF21 / 501D  FF22 / 501E  FF2A / 501F
FF2B / 5020  FF2C / 5021  FF2D / 5022  FF2E / 5023
FF2F / 5024  FF30 / 5025

REMOtELY CONNECTED COUPLING FACILITIES
CFNAME        COUPLING FACILITY
-------------- --------------------------
CF2            002094.IBM.02.00000002991E
PARTITION: 1D  CPCID: 00

CHPIDS ON CF1 CONNECTED TO REMOTE FACILITY
RECEIVER:   CHPID    TYPE
D0       ICP
D2       ICP

SENDER:     CHPID    TYPE
D0       ICP
D2       ICP

COUPLING FACILITY 002094.IBM.02.00000002991E
PARTITION: 1D  CPCID: 00
CONTROL UNIT ID: FFF6

NAMED CF2
COUPLING FACILITY SPACE UTILIZATION
ALLOCATED SPACE                  DUMP SPACE UTILIZATION
STRUCTURES:      359936 K        STRUCTURE DUMP TABLES:          0 K
DUMP SPACE:        2048 K                  TABLE COUNT:          0
FREE SPACE:      1644032 K       FREE DUMP SPACE:              2048 K
TOTAL SPACE:      2006016 K      TOTAL DUMP SPACE:              2048 K
MAX REQUESTED DUMP SPACE:          0 K
VOLATILE:           YES        STORAGE INCREMENT SIZE:        512 K
CFLEVEL:            15
CFCC RELEASE 15.00, SERVICE LEVEL 02.12
BUILT ON 09/14/2010 AT 13:00:00
COUPLING FACILITY HAS 0 SHARED AND 1 DEDICATED PROCESSORS
DYNAMIC CF DISPATCHING: OFF

CF REQUEST TIME ORDERING: REQUIRED AND ENABLED

COUPLING FACILITY SPACE CONFIGURATION
IN USE           FREE           TOTAL
CONTROL SPACE:    361984 K       1644032 K       2006016 K
NON-CONTROL SPACE:  0 K           0 K           0 K

SENDER PATH       PHYSICAL     LOGICAL     CHANNEL TYPE
D0               ONLINE       ONLINE       ICP
D2               ONLINE       ONLINE       ICP

COUPLING FACILITY SUBCHANNEL STATUS
TOTAL:   14   IN USE:   14   NOT USING:  0   NOT USABLE:  0

OPERATIONAL DEVICES / SUBCHANNELS:
FF23 / 5026  FF24 / 5027  FF25 / 5028  FF26 / 5029
FFB4 / 502E  FFB5 / 502F  FFB6 / 5030  FFB7 / 5031
FFB8 / 5032  FFB9 / 5033

REMOtELY CONNECTED COUPLING FACILITIES
CFNAME        COUPLING FACILITY
-------------- --------------------------
Both systems were running z/OS 1.12 with JES2 in a multiple access spool (MAS) configuration. Both systems were also accessible from anywhere within the IBM intranet using a TCP/IP connection.

### A.2 IMS configuration

Figure A-1 shows the basic configuration of our IMS 12 IMSplex with four IMS systems that each have an IMS Connect.
We omitted the internal resource lock manager (IRLM), Common Queue Server (CQS), Operations Manager (OM), Resource Manager (RM), Structured Call Interface (SCI) address spaces, and IMS repository server from the diagram for clarity.

Our IMS system is an IMSPlex, composed of four control regions. Two of the systems have shared queues and all four use shared databases and a single shared recovery control (RECON). The systems were remotely located on two LPARs systems named WTSC63 and WTSC64.

Starting with a simple IMS built from the IMS 12 Install or the Installation Verification Program (IVP), we updated the stage1 so that each system had a unique SUF=x and MSVID=nnn value. That change was also reflected in the IMS.PROCLIB(DFSPBxxx) member.

Each IMS had Virtual Telecommunications Access Method (VTAM) Multiple Systems Coupling (MSC) links defined to connect to the other three systems. (MSC is redundant with shared queues but the links were included in the IMS stage1 system generation anyway.)

A single IRLM region is running on each LPAR to reduce complexity. There is an SCI running on each LPAR. Each system has its own local OM. There is one shared RM (running without using a RM coupling facility structure).

### A.2.1 TCP/IP MSC links

The first change to the configuration was for the connectivity enhancements. The change made was to add MSC physical links, logical links and MSNAME using TCP/IP. As part of that testing, a NUCLEUS generate was done for each system to add the new stage1 parameters for TYPE=TCPIP MSC links. You must run a NUCLEUS stage1 or stage2 system definition and restart the system to add the support for MSC using TCP/IP, even if VTAM MSC links already exist in the system.

Example A-3, Example A-4, Example A-5, and Example A-6 show the definitions for each of the four IMS systems. For each link the partner ID and the suffix on the MSPLINK name match one another. As a convention, we defined a VTAM link with PARTNER=local/remote (for example, PARTNER=AB for I12A connected to I12B) and a TCPIP link with PARTNER=remote/local (for example, PARTNER=BA for I12A connected to I12B). The same naming scheme was used for link names, partner IDs, and MSNAMEs.

The MSC system identifier (SYSID) values are defined as shown in the following example. The SYSID values used for TCP/IP simply had 100 added to the value assigned for a VTAM link.

> As shown in Example A-3, IMS I12A uses 1 (VTAM) and 101 (TCP/IP).

```plaintext
Example A-3  Stage 1 macros for MSC in IMS system I12A

*---------------------------------------------------------------*
*   I12A LOCAL  SYSID: 1, 101           *
*   I12B REMOTE SYSID: 2, 102           *
*   I12C REMOTE SYSID: 3, 103           *
*   I12D REMOTE SYSID: 4, 104           *
*---------------------------------------------------------------*
*    VTAM MSC FROM I12A TO I12B          *
*---------------------------------------------------------------*
LINKAB  MSPLINK TYPE=VTAM,NAME=APPL112B,SESSION=2,BUFSIZE=2048
MSLINK PARTNER=AB,MSPLINK=LINKAB
```
As shown in Example A-4, IMS I12B uses 2 (VTAM) and 102 (TCP/IP).

Example A-4 Stage 1 macros for MSC in IMS system I12B
As shown in Example A-5, IMS I12C uses 3 (VTAM) and 103 (TCP/IP).

Example A-5   Stage 1 macros for MSC in IMS system I12C
*----------------------------------*
*    VTAM MSC FROM I12A TO I12C    *
*----------------------------------*
LINKAC MSPLINK TYPE=VTAM,NAME=APPLI12A,SESSION=2,BUFSIZE=2048
MSLINK PARTNER=AC,MSPLINK=LINKAC
MSCAC MSNAME SYSID=(1,3)
*
*----------------------------------*
*   TCP/IP MSC FROM I12C TO I12A  *
*----------------------------------*
LINKCA MSPLINK TYPE=TCP/IP,NAME=I12A,SESSION=2,BUFSIZE=2048,
LCLICON=HWCF12C1,
LCLPLKID=MSC2C2A
MSLINK PARTNER=CA,MSPLINK=LINKCA
MSCCA MSNAME SYSID=(101,103)
*
*----------------------------------*
*    VTAM MSC FROM I12B TO I12C    *
*----------------------------------*
LINKBC MSPLINK TYPE=VTAM,NAME=APPLI12B,SESSION=2,BUFSIZE=2048
MSLINK PARTNER=BC,MSPLINK=LINKBC
MSCCB MSNAME SYSID=(2,3)
*
*----------------------------------*
*   TCP/IP MSC FROM I12C TO I12B  *
*----------------------------------*
LINKCB MSPLINK TYPE=TCP/IP,NAME=I12B,SESSION=2,BUFSIZE=2048,
LCLICON=HWCF12C1,
LCLPLKID=MSC2C2B
MSLINK PARTNER=CB,MSPLINK=LINKCB
MSCBC MSNAME SYSID=(102,103)
*
*----------------------------------*
*    VTAM MSC FROM I12C TO I12D    *
*----------------------------------*
LINKCD MSPLINK TYPE=VTAM,NAME=APPLI12D,SESSION=2,BUFSIZE=2048
MSLINK PARTNER=CD,MSPLINK=LINKCD
MSCCD MSNAME SYSID=(4,3)
*
*----------------------------------*
*   TCP/IP MSC FROM I12D TO I12C  *
*----------------------------------*
LINKDC MSPLINK TYPE=TCP/IP,NAME=I12D,SESSION=2,BUFSIZE=2048,
LCLICON=HWCF12C1,
LCLPLKID=MSC2C2D
MSLINK PARTNER=DC,MSPLINK=LINKDC
MSCDC MSNAME SYSID=(104,103)
As shown in Example A-6, IMS I12D uses 4 (VTAM) and 104 (TCP/IP).

Example A-6  Stage1 macros for MSC in IMS system I12D

*----------------------------------*
* I12A REMOTE SYSID: 1, 101     *
* I12B REMOTE SYSID: 2, 102     *
* I12C REMOTE SYSID: 3, 103     *
* I12D LOCAL  SYSID: 4, 104     *
*----------------------------------*

*------------------------------*
*    VTAM MSC FROM I12A TO I12D *
*------------------------------*
LINKAD  MSPLINK TYPE=VTAM,NAME=APPLI12A,SESSION=2,BUFSIZE=2048
MSLINK  PARTNER=AD,MSPLINK=LINKAD
MSCAD   MSNAME SYSID=(1,4)

*------------------------------*
*    TCP/IP MSC FROM I12D TO I12A *
*------------------------------*
LINKDA  MSPLINK TYPE=TCPIP,NAME=I12A,SESSION=2,BUFSIZE=2048,          X
       LCLICON=HWCF12D1,                                       X
       LCLPLKID=MSC2D2A
MSLKD   MSPLINK PARTNER=DA,MSPLINK=LINKDA
MSCDA   MSNAME SYSID=(101,104)

*------------------------------*
*    VTAM MSC FROM I12B TO I12D *
*------------------------------*
LINKBD  MSPLINK TYPE=VTAM,NAME=APPLI12B,SESSION=2,BUFSIZE=2048
MSLINK  PARTNER=BD,MSPLINK=LINKBD
MSCBD   MSNAME SYSID=(2,4)

*------------------------------*
*    TCP/IP MSC FROM I12D TO I12B *
*------------------------------*
LINKDB  MSPLINK TYPE=TCPIP,NAME=I12B,SESSION=2,BUFSIZE=2048,          X
       LCLICON=HWCF12D1,                                       X
       LCLPLKID=MSC2D2B
MSLKD   MSPLINK PARTNER=DB,MSPLINK=LINKDB
MSCDB   MSNAME SYSID=(102,104)

*------------------------------*
*    VTAM MSC FROM I12C TO I12D *
*------------------------------*
LINKCD  MSPLINK TYPE=VTAM,NAME=APPLI12C,SESSION=2,BUFSIZE=2048
MSLINK  PARTNER=CD,MSPLINK=LINKCD
MSCCD   MSNAME SYSID=(3,4)

*------------------------------*
*    TCP/IP MSC FROM I12D TO I12C *
*------------------------------*
Example A-7 shows the complete MSC link network using the IMS MSC Verification Utility.

Example A-7  MSC links (output from the MSC Verification Utility)

<table>
<thead>
<tr>
<th>MS001</th>
<th>MS002</th>
<th>MS003</th>
<th>MS004</th>
</tr>
</thead>
<tbody>
<tr>
<td>0001 LOCAL 001 AB 001 AC 001 AD</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0002 002 AB LOCAL 002 BC 002 BD</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0003 003 AC 003 BC LOCAL 003 CD</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0004 004 AD 004 BD 004 CD LOCAL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0101 LOCAL 001 BA 001 CA 001 DA</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0102 002 BA LOCAL 002 CB 002 DB</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0103 003 CA 003 CB LOCAL 003 DC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0104 004 DA 004 DB 004 DC LOCAL</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| VTAM AB 001--------AB 001 |
| VTAM AC 003-----------------AC 001 |
| VTAM AD 005------------------AD 001 |
| TCP BA 002--------BA 002 |
| TCP CA 004-----------------CA 002 |
| TCP DA 006------------------DA 002 |
| VTAM BC 003--------BC 003 |
| VTAM BD 005-----------------BD 003 |
| TCP CB 004--------CB 004 |
| TCP DB 006------------------DB 004 |
| VTAM CD 005--------CD 005 |
| TCP DC 006--------DC 006 |

We also added a set of transactions (Example A-8) that were defined as local in one system and as remote in each of the other three systems. The xMSC transaction was assigned to the VTAM MSC links. The xMSC1 transaction was assigned to the TCP/IP MSC link.

The COBOL program that the transaction runs simply makes some IMS INQY calls and builds a six-line reply with the data returned from those calls.

Example A-8  Application and transaction definitions (only I12A shown)

*----------------------------------*
* I12A LOCAL SYSID: 1, 101 *
* I12B REMOTE SYSID: 2, 102 *
* I12C REMOTE SYSID: 3, 103 *
* I12D REMOTE SYSID: 4, 104 *
*----------------------------------*
*                              *
* Local                        *
*                              *
* APPLCTN GPSB=MSC1I12A,LANG=ASSEM *
* TRANSACT CORANSACT CODE=AMSCI1,PRTY=(7,10,2),INQUIRY=NO,MODE=SNGL, *
* MSGTYPE=(SNGLSEG,NONRESPONSE),EDIT=ULC *
*                              *
* Remote in I12B                *
*                              *
A.2.2 Base Primitive Environment Database Recovery Control

The Database Recovery Control (DBRC) procedures for all systems were updated to use Base Primitive Environment (BPE)-based DBRC (Example A-9).

Example A-9  DBRC job control language (JCL) using BPE

```plaintext
//DBRC    PROC RGN=0M,SOUT=A,
//         SYS1='IMS12A.',
//         BPECFG=BPECONFG,
//         DBRCINIT=12X,
```
The DBRC initialization member is shown in Example A-10. The DBRC exit DSPSCIX0 was installed so that the IMSPLEX and GROUPID parameters were set automatically. Using that exit also meant that Automatic RECON Loss Notification (ARLN) was active. The RECON was updated to have MINVERS('12.1').

Example A-10  IMS.PROCLIB member DSPBI12X

A.3  IMS repository

We created an IMS repository using phase U of the IMS Installation and Verification Program (IVP). See 8.5.5, “Changes to the IVP for IMS 12” on page 302 for more information about this topic. One system I12D was updated to use the repository rather than MODBLKS or RDDS data sets. Example A-11 shows the repository JCL.

Example A-11  Repository JCL

A.3  IMS repository
A.4 IMS Connect

The JCL used to run IMS Connect was identical for each system. Only the parameters defining the configuration members were different in each case. See Example A-12.

The BPE configuration had minor differences for the recorder trace data set. IMS Connect configuration members had definitions that were unique to each system.

Example A-12 IMS Connect JCL

```plaintext
//*****************************************************************************
//*     IMS 12.1
//* HWS PROCEDURE IM12xHW1
//* FUNCTION: START IMS CONNECT REGION WITH IMS Connect Extensions 2.2
//*    PARAMETERS:
//*         BPECFG          BPE TRACE, SOAP PARAMETERS
//*         HWSCFG         IMS CONNECT CONFIGURATION
//*****************************************************************************

IM12BHW1 PROC RGN=0M,TME=1440,SOUT=*,
              BPECFG=BPEHWS2x,HWSCFG=HWCF12x1

WHSREGN EXEC PGM=HWSHWS500,REGION=&RGN,TIME=&TME,
              PARM='BPECFG=&BPECFG,HWSCFG=&HWSCFG'
STEPLIB DD   DSN=CEX.V2R2M0.SCEXLINK,DISP=SHR
             DSN=FUN.V2R2M0.SFUNLINK,DISP=SHR
             DSN=IMS12Q.SDFSRESL,DISP=SHR
PROCLIB DD   DSN=IMS12Q.PROCLIB,DISP=SHR
SYSPRINT DD   SYSOUT=&SOUT
SYSDUMP DD   SYSOUT=&SOUT
HWSRCORD DD   DSN=IMS12Q.HWSRCDR,DISP=SHR /* Not used */
CEXREPOS DD   DSN=CEX.V2R2M0.IMS12Q.REPOS,DISP=SHR
CEXPRINT DD   SYSPRINT DD   SYSOUT=&SOUT

The BPE configuration in Example A-13 has the new recorder trace. The HWSRCORD was removed from JCL.

Example A-13 IMS Connect BPE configuration

*****************************************************************************
* CONFIGURATION FILE FOR BPE WITH SOAP GATEWAY
*****************************************************************************

LANG=ENU /* LANGUAGE FOR MESSAGES */
        /* (ENU = U.S. ENGLISH) */

# DEFINITIONS FOR BPE SYSTEM TRACES
# TRCLEV=(*,HIGH,BPE,PAGES=20) /* DEFAULT TRACES TO HIGH */
Example A-14, Example A-15, Example A-16, and Example A-17 show the HWSCFG members for each copy of IMS Connect. Example A-14 is shown with comments to document each parameter. SSLPORT is not used.

Example A-14  IMS Connect IM12AHW1 configuration

```plaintext
* --------------HWS MEMBER FOR IMS12A   HWCF12A1-------------
* ------------------HWS-----------------------
* ID=
* PSWDMC=N  MIXED-CASE PASSWORDS DISABLED
* CMOATQ=  NAME OTMA CMO ACK TIMEOUT QUEUES(DEF DFS££TOQ)
* CAN BE OVERRIDED BY DATASTORE
* RACF=N  ENABLES SECURITY SUPPORT BY RACF
* RRS=Y  RESOURCE RECOVERY SERVICES ENABLED
* SMEMBER  SUPERMEMBER
* #UIDAGE=  REFRESH INTERVAL FOR CACHED RACF USER IDS IN SECS
* #UIDCACHE=Y IMS CONNECT CACHES VERIFIED RACF USER IDS FOR REUSE
* XIBAREA=  NUMBER OF FULLWORDS ALLOCATED FOR THE XIB USER AREA
HWS=(ID=HWS12A1,PSWDMC=R,RACF=N,RRS=Y,
    SMEMBER=I12S,
    UIDCACHE=Y,UIDAGE=500,XIBAREA=50)
* -------------------TCP/IP-------------------
```
* EXIT NAMES OF ONE OR MORE IMS MESSAGE EXIT ROUTINES
* HWSJAVA0 IS AUTOMATICALLY LOADED
* IMS CONTROL CENTER REQUIRES HWSCSLO0 AND HWSCSLO1
* HOSTNAME TCP/IP JOBNAME
* PORTID TCP/IP PORT ENABLES LOCAL OPTION CONNECTIONS
* -- OR --
* PORT TCP/IP PORT
* SSLPORT TCP/IP PORT FOR SSL SUPPORT
* SSLENVAR NAME OF THE SSL INITIALIZATION FILE
TCPIP=(EXIT=(HWSMPL1,HWSSOAP1),HOSTNAME=TCP/IP,PORT(ID=7100), PORT(ID=7101),PORT(ID=7102))
* ------------------------DATASTORE --------------------------
* --DATASTORE ADDRESSES IMS CONTROL REGION VIA OTMA
* GROUP XCF(OTMA) FROM GRNAME IN DFSBPXXX
* ID DATA STORE NAME (INIRM REQUEST)
* MEMBER XCF MEMBER NAME THAT IDENTIFIES IMS CONNECT
* IF SAME GROUP, NAME MUST BE UNIQUE WITHIN DATASTORE
* TMEMBER THE XCF MEMBER NAME FOR IMS
* OAAV= (ACEE) AGING VALUE, IN SECONDS (DEFAULT 999999)
* MAXI= INPUT MESSAGE FLOOD CONTROL VALUE
* CMOAT0= OTMA CMO ACK TIMEOUT QUEUE (DEF DFS££TOQ)
* APPL= TCP/IP APPL NAME DEFINED TO RACF IN PTKTDATA
* DRI= OWN OTMA DESTINATION RESOLUTION USER EXIT NAME
* RRNAME= NAME OF AN ALTERNATE DESTINATION SPECIFIED IN A CLIENT
* #ACKTO= TIMEOUT INTERVAL FOR OUTPUT ACKS.CMO,CM1..IMS TO IMS
* (DEFAULT 120)
* REROUTE FOR IMS TO IMS TO THE DEFAULT TIMEOUT QUEUE, DFS££TOQ
* SMEMBER SUPERMEMBER
* -TO I12A
* DATASTORE=(GROUP=I12XOTMA,ID=I12A,MEMBER=HWBI12A1,TMEMBER=I12AOTMA, OAAV=300,SMEMBER=I12S)
* -------------------------ADAPTER----------------------------
* XML=Y XML ADAPTER SUPPORT SHOULD BE ENABLED (SOAP GATEWAY)
* ADAPTER=(XML=Y)
* -------------------------IMSPLEX----------------------------
* -- REGISTERS IMS CONNECT AS A MEMBER OF AN IMPLEX
* -- ENABLES BOTH IMS TYPE-2 COMMAND SUPPORT FOR IMS CONNECT
* -- COMM BETWEEN IMS CONNECT AND OTHER MEMBERS OF THE IMSPLEX.
* MEMBER NAME THAT IDENTIFIES IMS CONNECT IN THE IMSPLEX
* TMEMBER IMSPLEX THAT IMSCON IS JOINING, IMSPLEX(NAME= )
* IMSPLEX=(MEMBER=HWSI12A1,TMEMBER=IM12X)
* -------------------------ODACCESS---------------------------
* DRDAPORT PORT NUMBER, TCP/IP KEEP ALIVE VALUE,
* (ID PORTNR
* KEEPAV TCP/IP LAYER SENDS A PACKET ON IDLE CONS
* PORTMTIM AMOUNT TIME IMS CONNECT WAITS FOR NEXT INPUT)
* ODBMAUTOCONN SPECIFIES WHETHER IMS CONNECT AUTOMATICALLY
* NEW AND EXISTING INSTANCES OF ODBM WITHIN AN IMSPLEX.
* ODACCESS=(DRDAPORT=(ID=6100,KEEPAV=0,PORTMTIM=18000), DRDAPORT=(ID=6101,KEEPAV=0,PORTMTIM=18000), ODBMAUTOCONN=Y)
* ####-------------------------MSC----NEW IN IMS 12----------
* GENIMSID= MSC TCP/IP GENERIC IMS ID (OPTIONAL)
* ALSO SPECIFIED ON THE GENIMSID IN DFSDCxxx
* IMSPLEX (MEMBER, TMEMBER)
* THIS NAME MUST MATCH THE NAME SPECIFIED ON THE LCLICON
* PARM OF THE MSPLINK MACRO DEFINITION OF THE LOCAL IMS SYSTEM
* LCLIMS= IMS ID OF THE LOCAL IMS SYSTEM AS REGISTERED WITH SCI
* =>QUERY IMSPLEX ?? C
Appendix A. Environment description

Example A-15 shows the IMS Connect IM12BHW1 configuration.

Example A-15  IMS Connect IM12BHW1 configuration

* LCLPKLID= LOCAL NAME OF THE MSC PHYSICAL LINK
* RMTIMS= IMS ID OF THE REMOTE IMS SYSTEM AS REGISTERED WITH SCI
  **query IMSplex ?? C**
* RMTIMSCON= REMOTE IMS CONNECT CONNECTION TO USE FOR MSC MESSAGES
  * MUST MATCH ID PARAMETER OF ONE OF THE
  * RMTIMSCON STATEMENTS SPECIFIED IN THE LOCAL IMS CONNECT
* RMTPLKID= NAME MSC PHYSICAL LINK IN REMOTE LCLPKLID
*MSC=(LCLPKLID=MSC2A2B,RMTPLKID=MSC2B2A,LCLIMS=I12A,RMTIMS=I12B,
  GENIMSID=ACIM,
  IMSPLEX=(MEMBER=HWSI12A1,TMEMBER=IM12X),RMTIMSCON=HWSI12B1)
MSC=(LCLPKLID=MSC2C2B,RMTPLKID=MSC2B2C,LCLIMS=I12C,RMTIMS=I12B,
  GENIMSID=ACIM,
  IMSPLEX=(MEMBER=HWSI12A1,TMEMBER=IM12X),RMTIMSCON=HWSI12B1)

*####----------------------RMTIMSCON---NEW IN IMS/VS 12--------*
* -CONNECTION DEFINED BY RMTIMSCON CAN BE USED FOR EITHER OTMA
* - OR MSC MESSAGES, BUT NOT BOTH
* :OTMA REQUIRES IMSCON STATEMENT BY SENDING IMSCONN
* :MSC REQUIRES CORRESPONDING RMTIMSCON IN OTHER HWSCFGXX
* APPL= APPLICATION NAME TO USE IN A RACF PASSTICKET
* ID IDENTIFIES THIS DEFINITION OF A CONNECTION
* AUTOCONN= CONNECTS TO REMOTE IMSCON DURING STARTUP (DEF=N)
* IDLETO= OPEN SOCKET CONNECTIONS IDLE TIME
* *EITHER THE IPADDR OR THE HOSTNAME PARAMETER
* IPADDR= IP_ADDRESS  OF OTHER IMS CONNECT
* HOSTNAME   THE HOST NAME OF THE REMOTE IMS CONNECT
* PERSISTENT=DEFINES SOCKETS USED FOR CONNECTION AS PERSIST
* PORT= PORTNR OF LISTENING PORT
* RESVSOC= NUMBER OF SEND SOCKETS THAT IMS CONNECT RESERVES
* TO I12A VIA TCP/IP
* *-USED BY MSC
RMTIMSCON=(ID=HWSI12B1,HOSTNAME=WTSC64.ITSO.IBM.COM,
  PORT=7202,AUTOCONN=Y,RESVSOC=10)
*---**************** END   HWCF12A1   DEFINITION---------------

Example A-15 shows the IMS Connect IM12BHW1 configuration.

Example A-15  IMS Connect IM12BHW1 configuration

* -------------------HWS MEMBER FOR IMS12B     HWCF12B1---------
HWS=(ID=HWSI12B1,PSWDMC=R,RACF=N,RSS=Y,
  SMEMBER=I12S,
  UIDCACHE=Y,UIDAGE=500,XIBAREA=50)

* TCPIP=(EXIT=(HWSSMPL1,HWSSOAP1),HOSTNAME=TCPIP,PORT(ID=7200),
  PORT(ID=7201),PORT(ID=7202))

* DATASTORE=(GROUP=I12XOTMA,ID=I12A,MEMBER=HWBI12A2,TMEMBER=I12AOTMA,
  OAAV=0,SMEMBER=I12S)
DATASTORE=(GROUP=I12XOTMA,ID=I12B,MEMBER=HWBI12B2,TMEMBER=I12BOTMA)

* ADAPTER=(XML=Y)

* IMSPLEX=(MEMBER=HWSI12B1,TMEMBER=IM12X)

* ODACCESS=(DRDAPORT=(ID=6200,KEEPAV=0,PORTTMOT=18000),
  DRDAPORT=(ID=6201,KEEPAV=0,PORTTMOT=18000),ODBMAUTOCONN=Y)
  *
Example A-16 shows the IMS Connect IM2CHW1 configuration.

Example A-16  IMS Connect IM12CHW1 configuration

*  ------------HWS  MEMBER FOR IMS12C     HWCF12C1-------------
HWS=(ID=HWSI12C1,PSWDMC=R,RACF=N,RRS=Y,
UIDCACHE=Y,UIDAGE=500,XIBAREA=50)
  *
TCPIP=(EXIT=(HWSSMPL1,HWSSOAP1),HOSTNAME=TCPIP,PORT(ID=7300),
PORT(ID=7301),PORT(ID=7302))
  *
DATASTORE=(GROUP=I12XOTMA,ID=I12A,MEMBER=HWBI12A3,TMEMBER=I12AOTMA,
OAAV=600)
DATASTORE=(GROUP=I12XOTMA,ID=I12C,MEMBER=HWBI12C3,TMEMBER=I12COTMA)
  *
ADAPTER=(XML=Y)
  *
IMSPLEX=(MEMBER=HWSI12C1,TMEMBER=IM12X)
  *
ODACCESS=(DRDAPORT=(ID=6300,KEEPAV=0,PORTTMOT=18000),
DRDAPORT=(ID=6301,KEEPAV=0,PORTTMOT=18000),ODBMAUTOCONN=Y)
  *
RMTIMSCON=(ID=HWSI12D1,HOSTNAME=WTSC64.ITSO.IBM.COM,
PORT=7401,AUTOCONN=Y,PERSISTENT=Y,IDLETO=60000,RESVSOC=10)

Example A-17 shows the IMS Connect IM2DHW1 configuration.

Example A-17  IMS Connect IM12DHW1 configuration

*  ------------HWS  MEMBER FOR IMS12D     HWCF12D1-------------
HWS=(ID=HWSI12D1,PSWDMC=R,RACF=N,RRS=Y,
SMEMBER=I12S,
UIDCACHE=Y,UIDAGE=500,XIBAREA=50)
  *
TCPIP=(EXIT=(HWSSMPL1,HWSSOAP1),HOSTNAME=TCPIP,PORT(ID=7400),
PORT(ID=7401),PORT(ID=7402))
  *
DATASTORE=(GROUP=I12XOTMA,ID=I12D,MEMBER=HWBI12D4,TMEMBER=I12DOTMA,
SMEMBER=I12S)
  *
ADAPTER=(XML=Y)
  *
IMSPLEX=(MEMBER=HWSI12D1,TMEMBER=IM12X)
  *
ODACCESS=(DRDAPORT=(ID=6400,KEEPAV=0,PORTTMOT=18000),
DRDAPORT=(ID=6401,KEEPAV=0,PORTTMOT=18000),ODBMAUTOCONN=Y)
  *
RMTIMSCON=(ID=HWSI12C1,HOSTNAME=WTSC63.ITSO.IBM.COM,
PORT=7301,AUTOCONN=Y,PERSISTENT=Y,IDLETO=60000,RESVSOC=10)
For the SOAP gateway, we added an extra instance of IMS Connect using application transparent-transaction layer security (AT-TLS) to provide a Secure Sockets Layer (SSL). See Example A-18.

Example A-18  IMS Connect configuration for IMS SOAP Gateway

HWS=(ID=HWSI12A2,PSWDMC=N,RACF=N,RRS=Y,UIDCACHE=Y,XIBAREA=50)
TCP/IP=(EXIT=(HWSMPL1,HWSOAPI1),HOSTNAME=TCP/IP,PORT(ID=5100),
PORT(ID=5101),PORT(ID=5102))
DATASTORE=(GROUP=I12XOTMA,ID=I12A,MEMBER=HWBI12A2,TMEMBER=I12AOTMA)
ADAPTER=(XML=Y)
IMSPLEX=(MEMBER=HWSI12A2,TMEMBER=IM12X)
ODACCESS=(DRDAPORT=(ID=4100,KEEPAV=0,PORTTMT=18000),
DRDAPORT=(ID=4101,KEEPAV=0,PORTTMT=18000),
ODBAUTOCONN=Y)

A.5 Shared queues

Shared queues were set up between the two IMS systems running on WTSC63. Example A-19 shows the coupling facility structures we defined.

Example A-19  IMS shared queues coupling facility structures

STRUCTURE NAME(I12X_IRLM) SIZE(32768)
  PREFLIST(CF2,CF1)
  REBUILDPERCENT(1)

STRUCTURE NAME(I12X_MSGQ)
  SIZE(16000)
  INITSIZE(8000)
  MINSIZE(8000)
  PREFLIST(CF2,CF1)
  REBUILDPERCENT(1)
  ALLOWAUTOALT(YES)
  FULLTHRESHOLD(60)

STRUCTURE NAME(I12X_MSGQOFLW)
  SIZE(8000)
  MINSIZE(8000)
  PREFLIST(CF2,CF1)
  REBUILDPERCENT(1)
  ALLOWAUTOALT(YES)
  FULLTHRESHOLD(60)

STRUCTURE NAME(I12X_EMHQ)
  SIZE(16000)
  INITSIZE(10000)
  MINSIZE(10000)
  PREFLIST(CF2,CF1)
  REBUILDPERCENT(1)
  ALLOWAUTOALT(YES)
  FULLTHRESHOLD(60)

STRUCTURE NAME(I12X_EMHQOFLW)
  SIZE(8000)
  MINSIZE(8000)
  PREFLIST(CF2,CF1)
  REBUILDPERCENT(1)
ALLOWAUTOALT(YES)
FULLTHRESHOLD(60)

STRUCTURE NAME(I12X_LOGRMSGQ)
  SIZE(16000)
  INITSIZE(11000)
  PREFLIST(CF2,CF1)

STRUCTURE NAME(I12X_LOGREMHQ)
  SIZE(4000)
  PREFLIST(CF2,CF1)
  REBUILDPERCENT(1)

STRUCTURE NAME(I12X_RSC)
  SIZE(16000)
  INITSIZE(8000)
  MINSIZE(8000)
  ALLOWAUTOALT(YES)
  FULLTHRESHOLD(60)
  DUPLEX(ALLOWED)
  PREFLIST(CF2,CF1)

A z/OS log stream is needed for IMS CQS as shown in Example A-20.

Example A-20  IMS shared queues z/OS log streams

DEFINE STRUCTURE NAME(I12X_LOGRMSGQ)
  LOGSNUM(1)
  AVGBUFSIZE(4096)
  MAXBUFSIZE(65272)
DEFINE LOGSTREAM NAME (SYSLOG.MSGQL.LOGOFFLD)
  STRUCTNAME(I12X_LOGRMSGQ)
  LS_MGMTCLAS(NO_LS_MGMTCLAS)
  LS_SIZE(200)
  HLQ(LOGR)

DEFINE STRUCTURE NAME(I12X_LOGREMHQ)
  LOGSNUM(1)
  AVGBUFSIZE(4096)
  MAXBUFSIZE(65272)
DEFINE LOGSTREAM NAME (SYSLOG.EMHQL.LOGOFFLD)
  STRUCTNAME(I12X_LOGREMHQ)
  LS_MGMTCLAS(NO_LS_MGMTCLAS)
  LS_SIZE(200)
  HLQ(LOGR)

Example A-21 shows the JCL used for IMS CQS. The CQSINIT and SSN parameters were suffixed with the IMS ID letter (for example CQSINIT=12A and SSN=C12A).

Example A-21  IMS CQS JCL

```c
//*------------------------------------------------------------------*
//*     PARAMETERS:                                                  *
//*     BPECFG  - NAME OF BPE MEMBER                                 *
//*     CQSINIT - Suffix for your CQSIxxx member                     *
//*     PARM1   - other override parameters                          *
//*               SSN     - CQS subsystem name                       *
//*               STRDEFG - Suffix for CQSSGxxx member               *
//*               STRDEFL - Suffix for CQSSLxxx member               *
```
The following examples show the parameters for IMS CQS proclib members:

- **Example A-22** shows the parameters for CQSSG12X.

  Example A-22  CQSSG12X global shared queues configuration

  ******************************************************************************
  * DEFINITION FOR IMS MESSAGE QUEUE STRUCTURES
  ******************************************************************************

  STRUCTURE (STRNAME=I12X_MSGQ,OVFLWSTR=I12X_MSGQOFLW,
  SRDSDSN1=IMS12Q.IMS12X.MSGQ.SRDS1,
  SRDSDSN2=IMS12Q.IMS12X.MSGQ.SRDS2,
  LOGNAME=SYSLOG.MSGQL.LOGOFFLD,
  OBJAVGSZ=1024)

  --------------------------------------------------------------------

- **Example A-23** shows the parameters for CQSSL12A.

  Example A-23  CQSSL12A local shared queues configuration for I12A

  *--------------------------------------------------------------------*
  * LOCAL STRUCTURE DEFINITION PROCLIB MEMBER
  *--------------------------------------------------------------------*

  *--------------------------------------------------------------------*
  * DEFINITION FOR IMS MESSAGE QUEUE STRUCTURE
  *--------------------------------------------------------------------*
Example A-24 shows the parameters for CQSSL12C.

Example A-24  CQSSL12C local shared queues configuration for I12C

Example A-25 shows the parameters for CQSIP12A.

Example A-25  CQSIP12A CQS initialization parameters for I12A
Example A-26 shows the parameters for CQSIP12C.

<table>
<thead>
<tr>
<th>Example A-26</th>
<th>CQSIP12C CQS initialization parameters for I12C</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><em>--------------------------------------------------</em></td>
</tr>
<tr>
<td></td>
<td>* CQS INITIALIZATION PROCLIB MEMBER.</td>
</tr>
<tr>
<td></td>
<td><em>--------------------------------------------------</em></td>
</tr>
<tr>
<td>ARMREST=N</td>
<td>/* ARM SHOULD RESTART CQS ON FAILURE */</td>
</tr>
<tr>
<td>CQSGROUP=CQSGX</td>
<td>/* GROUP NAME (XCF GROUP = GRUP1CQS) */</td>
</tr>
<tr>
<td>SSN=C12C</td>
<td>/* CQS SUBSYSTEM NAME (CQSID = CQS1CQS) */</td>
</tr>
<tr>
<td>STRDEFG=12X</td>
<td>/* GLOBAL STR DEFINITION MEMBER = CQSSG000 */</td>
</tr>
<tr>
<td>STRDEFL=12C</td>
<td>/* LOCAL STR DEFINITION MEMBER = CQSSL000 */</td>
</tr>
<tr>
<td>IMSPLEX(NAME=IM12X)</td>
<td>/* IMSPLEX NAME (CSLPLEX1) */</td>
</tr>
<tr>
<td></td>
<td><em>--------------------------------------------------</em></td>
</tr>
<tr>
<td></td>
<td>* END OF MEMBER CQSIP12C</td>
</tr>
<tr>
<td></td>
<td><em>--------------------------------------------------</em></td>
</tr>
</tbody>
</table>

The IMS startup member DFSPBxxx was also updated to point to the DFSSQxxx members using the `SHAREDQ=xxx` parameter. The DFSSQxxx member identifies the local CQS and the queue structures that IMS should use. Example A-27 shows the PROCLIB members for I12A.

Example A-27  IMS.PROCLIB member DFSSQ12A

<table>
<thead>
<tr>
<th>Example A-27</th>
<th>IMS.PROCLIB member DFSSQ12A</th>
</tr>
</thead>
<tbody>
<tr>
<td>CQS=IM12ACQS</td>
<td></td>
</tr>
<tr>
<td>CQSSSN=C12A,</td>
<td></td>
</tr>
<tr>
<td>EMHQ=I12X_EMHQ,</td>
<td></td>
</tr>
<tr>
<td>MSGQ=I12X_MSGQ,</td>
<td></td>
</tr>
<tr>
<td>SQGROUP=CQSGX</td>
<td></td>
</tr>
</tbody>
</table>

Example A-28 shows the PROCLIB members for I12C.

Example A-28  IMS.PROCLIB member DFSSQ12C

<table>
<thead>
<tr>
<th>Example A-28</th>
<th>IMS.PROCLIB member DFSSQ12C</th>
</tr>
</thead>
<tbody>
<tr>
<td>CQS=IM12CCQS,</td>
<td></td>
</tr>
<tr>
<td>CQSSSN=C12C,</td>
<td></td>
</tr>
<tr>
<td>EMHQ=I12X_EMHQ,</td>
<td></td>
</tr>
<tr>
<td>MSGQ=I12X_MSGQ,</td>
<td></td>
</tr>
<tr>
<td>SQGROUP=CQSGX</td>
<td></td>
</tr>
</tbody>
</table>

The checkpoint and structure recovery data sets were created by modifying the JCL from phase O of the IMS 12 Install/IVP to use our data set names.
Recent maintenance: IMS 12 APARs

With a new version of IBM Information Management System (IMS) reaching general availability, the maintenance stream becomes extremely important. Feedback from early users and development of additional functions cause a flux of APARs which enrich and improve the product code.

This appendix examines recent maintenance for IMS 12 that relates in a general way to new functionality and critical corrections.

Keep in mind that these lists of APARs represent a snapshot of current maintenance at the time of writing. As such, they may be incomplete or even incorrect by the time you read this publication; they are presented here simply to help identify areas of functional improvements. At the time of your installation, be sure to contact your IBM Service Representative for the most current maintenance. Also check RETAIN to determine the applicability of these APARs to your environment and to verify prerequisites and postrequisites.

Use the Consolidated Service Test (CST) as the base for service.

Table B-1 lists various APARs that provide functional enhancements to IMS 12. The list is not exhaustive, so check RETAIN and the IMS website for current information.

<table>
<thead>
<tr>
<th>APAR #</th>
<th>Area</th>
<th>Text</th>
<th>PTF and notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>PK71135</td>
<td>OTMA</td>
<td>Activates the IMS synchronous callout function</td>
<td>UK50878</td>
</tr>
<tr>
<td>PM05243</td>
<td>DBRC</td>
<td>Coexistence IMS 10 to IMS 12</td>
<td>UK62970</td>
</tr>
<tr>
<td>PM05244</td>
<td>DBRC</td>
<td>Coexistence IMS 11 to IMS 12</td>
<td>UK62971</td>
</tr>
<tr>
<td>PM19025</td>
<td>Coexistence</td>
<td>Required for Common Service Layer (CSL) Resource Manager (RM) 1.3 on IMS 10</td>
<td>UK63960</td>
</tr>
<tr>
<td>PM19026</td>
<td>Coexistence</td>
<td>Required for CSL RM 1.4 on IMS 11</td>
<td>UK63964</td>
</tr>
<tr>
<td>PM23840</td>
<td>PSB changes</td>
<td>Apply to IMS 10 for coexistence with 12</td>
<td>UK64012</td>
</tr>
<tr>
<td>APAR #</td>
<td>Area</td>
<td>Text</td>
<td>PTF and notes</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td>----------------------------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>PM23843</td>
<td>PSB changes</td>
<td>Apply to IMS 12 for coexistence with 12</td>
<td>UK64013</td>
</tr>
<tr>
<td>PM24860</td>
<td>IMS Connect</td>
<td>Run under support</td>
<td>UK68052</td>
</tr>
<tr>
<td>PM26518</td>
<td>Internal APARs</td>
<td></td>
<td>UK62105</td>
</tr>
<tr>
<td>PM29809</td>
<td>Repository</td>
<td>Correct output of a \texttt{QUERY PGM NAME(*) SHOW(DEFN LOCAL)} command from RDDS</td>
<td>UK64166</td>
</tr>
<tr>
<td>PM31420</td>
<td>CICS</td>
<td>With the CCTL DRA Open Thread TCB enhancement, users of CCTL DRA (including CICS TS 4.2) are allowed DL/I processing on the application task control block (TCB)</td>
<td>UK70991</td>
</tr>
<tr>
<td>PM31779</td>
<td>Repository</td>
<td>Enablement of new functions for IMSRSC repository</td>
<td>UK68464</td>
</tr>
<tr>
<td>PM32390</td>
<td>GSAM</td>
<td>Apply to IMS 11 for coexistence with 12</td>
<td>UK65495</td>
</tr>
<tr>
<td>PM32394</td>
<td>IMS Connect</td>
<td>IMS Connect Extension (CEX) trace correction</td>
<td>UK65339</td>
</tr>
<tr>
<td>PM32548</td>
<td>GSAM</td>
<td>Apply to IMS 10 for coexistence with 12</td>
<td>UK65496</td>
</tr>
<tr>
<td>PM32766</td>
<td>Coexistence</td>
<td>Required for CSL RM 1.4 on IMS 11</td>
<td>UK68882</td>
</tr>
<tr>
<td>PM32805</td>
<td>Repository</td>
<td>Validate tran, routing code, and program attributes before writing to repository</td>
<td>OPEN</td>
</tr>
<tr>
<td>PM32951</td>
<td>Coexistence</td>
<td>Required for CSL RM 1.3 on IMS 10</td>
<td>UK68883</td>
</tr>
<tr>
<td>PM35503</td>
<td>Repository</td>
<td>\texttt{/CHECKPOINT FREEZE</td>
<td>DUMPQ</td>
</tr>
<tr>
<td>PM37257</td>
<td>Repository</td>
<td>Add RM and \texttt{IMPORT} command preconditioning support for the Repository change list</td>
<td>UK70001</td>
</tr>
<tr>
<td>PM37894</td>
<td>DBRC</td>
<td>Enables IMS Database Recovery Facility (DRF) to build Fast Path Secondary Index databases from a data entry database (DEDB)</td>
<td>UK69302</td>
</tr>
<tr>
<td>PM41761</td>
<td>Repository</td>
<td>The \texttt{QUERY SHOW(DEFN)} command is showing extra incorrect information</td>
<td>UK70007</td>
</tr>
<tr>
<td>PM41952</td>
<td>Repository</td>
<td>Do not allow an RM enabled with Repository to come up if the Repository Server is not available</td>
<td>UK72427</td>
</tr>
<tr>
<td>PM42130</td>
<td>Repository</td>
<td>Enable change list support</td>
<td>OPEN</td>
</tr>
<tr>
<td>PM47327</td>
<td>CICS</td>
<td>Post requirement for PM31420</td>
<td>OPEN</td>
</tr>
</tbody>
</table>
## Abbreviations and acronyms

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACB</td>
<td>application control block</td>
</tr>
<tr>
<td>ACBLIB</td>
<td>ACB library</td>
</tr>
<tr>
<td>ACEE</td>
<td>access control environment element</td>
</tr>
<tr>
<td>ACK</td>
<td>acknowledgement</td>
</tr>
<tr>
<td>AGN</td>
<td>Application Group Name</td>
</tr>
<tr>
<td>AIB</td>
<td>application interface block</td>
</tr>
<tr>
<td>AOI</td>
<td>Automated Operator Interface</td>
</tr>
<tr>
<td>AOP</td>
<td>automated operator program</td>
</tr>
<tr>
<td>AOS</td>
<td>APPC/OTMA SMQ</td>
</tr>
<tr>
<td>APF</td>
<td>authorized program facility</td>
</tr>
<tr>
<td>API</td>
<td>application programming interface</td>
</tr>
<tr>
<td>APPC</td>
<td>Advanced Program-to-Program Communication</td>
</tr>
<tr>
<td>APSB</td>
<td>allocate PSB</td>
</tr>
<tr>
<td>ARLN</td>
<td>Automatic RECON Loss Notification</td>
</tr>
<tr>
<td>ARM</td>
<td>Automatic Restart Manager</td>
</tr>
<tr>
<td>ASN</td>
<td>Abend Search and Notification</td>
</tr>
<tr>
<td>BMP</td>
<td>batch messaging program</td>
</tr>
<tr>
<td>BPE</td>
<td>Base Primitive Environment</td>
</tr>
<tr>
<td>BSAM</td>
<td>Basic Sequential Access Method</td>
</tr>
<tr>
<td>CA</td>
<td>certificate authority</td>
</tr>
<tr>
<td>CA</td>
<td>change accumulation</td>
</tr>
<tr>
<td>CA</td>
<td>control area</td>
</tr>
<tr>
<td>CBM</td>
<td>Component Business Modeling</td>
</tr>
<tr>
<td>CBPDO</td>
<td>Custom-built Product Delivery Option</td>
</tr>
<tr>
<td>CCF</td>
<td>Common Connector Framework</td>
</tr>
<tr>
<td>CCI</td>
<td>Common Client Interface</td>
</tr>
<tr>
<td>CCTL</td>
<td>coordinator controller</td>
</tr>
<tr>
<td>CDE</td>
<td>contents directory entry</td>
</tr>
<tr>
<td>CDSP</td>
<td>control block</td>
</tr>
<tr>
<td>CEX</td>
<td>Connect Extension</td>
</tr>
<tr>
<td>CGI</td>
<td>Common Gateway Interface</td>
</tr>
<tr>
<td>CI</td>
<td>Control Interval</td>
</tr>
<tr>
<td>CICS</td>
<td>Customer Information Control System</td>
</tr>
<tr>
<td>CLI</td>
<td>command-line interface</td>
</tr>
<tr>
<td>CLP</td>
<td>command-line processor</td>
</tr>
<tr>
<td>COMPID</td>
<td>component IDs</td>
</tr>
<tr>
<td>CPC</td>
<td>central processor complex</td>
</tr>
<tr>
<td>CP</td>
<td>central processor</td>
</tr>
<tr>
<td>CQS</td>
<td>Common Queue Server</td>
</tr>
<tr>
<td>CRL</td>
<td>certificate revocation list</td>
</tr>
<tr>
<td>CRSS</td>
<td>Conditional Reorganization Support Service</td>
</tr>
<tr>
<td>CSA</td>
<td>common storage area</td>
</tr>
<tr>
<td>CSL</td>
<td>Common Service Layer</td>
</tr>
<tr>
<td>CSM</td>
<td>Complete Status Message</td>
</tr>
<tr>
<td>CST</td>
<td>Consolidated Service Test</td>
</tr>
<tr>
<td>CSV</td>
<td>comma-separated value</td>
</tr>
<tr>
<td>CTG</td>
<td>CICS Transaction Gateway</td>
</tr>
<tr>
<td>DB2</td>
<td>Database 2</td>
</tr>
<tr>
<td>DBCTL</td>
<td>Database Control</td>
</tr>
<tr>
<td>DBD</td>
<td>database description</td>
</tr>
<tr>
<td>DBDS</td>
<td>database data sets</td>
</tr>
<tr>
<td>DBHDA</td>
<td>DB Historical Data Analyzer</td>
</tr>
<tr>
<td>DBMS</td>
<td>database management system</td>
</tr>
<tr>
<td>DBPCB</td>
<td>database PCB</td>
</tr>
<tr>
<td>DBRA</td>
<td>Database Resource Adapter</td>
</tr>
<tr>
<td>DBRC</td>
<td>Database Recovery Control</td>
</tr>
<tr>
<td>DBSR</td>
<td>Database Segment Restructure</td>
</tr>
<tr>
<td>DCC</td>
<td>Data Communications Control</td>
</tr>
<tr>
<td>DMB</td>
<td>data management block</td>
</tr>
<tr>
<td>DRD</td>
<td>dynamic resource definition</td>
</tr>
<tr>
<td>DRDA</td>
<td>Distributed Relational Database Architecture</td>
</tr>
<tr>
<td>DRF</td>
<td>Database Recovery Facility</td>
</tr>
<tr>
<td>DSN</td>
<td>Data Source Name</td>
</tr>
<tr>
<td>DTP</td>
<td>distributed transaction processing</td>
</tr>
<tr>
<td>EAB</td>
<td>Enterprise Access Builder</td>
</tr>
<tr>
<td>EAI</td>
<td>enterprise application integration</td>
</tr>
<tr>
<td>EAS</td>
<td>extended addressing space</td>
</tr>
<tr>
<td>EAV</td>
<td>extended address volume</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Full Form</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>ECB</td>
<td>event control block</td>
</tr>
<tr>
<td>ECNT</td>
<td>extended communication name table</td>
</tr>
<tr>
<td>ECSA</td>
<td>extended common service area</td>
</tr>
<tr>
<td>EINIT</td>
<td>Early Initialization exit routine</td>
</tr>
<tr>
<td>EIS</td>
<td>enterprise information system</td>
</tr>
<tr>
<td>EJB</td>
<td>Enterprise JavaBeans</td>
</tr>
<tr>
<td>EMAC</td>
<td>extended area control block</td>
</tr>
<tr>
<td>EMCS</td>
<td>extended multiple console support</td>
</tr>
<tr>
<td>EMF</td>
<td>Eclipse Modeling Framework</td>
</tr>
<tr>
<td>EOM</td>
<td>End of Marketing</td>
</tr>
<tr>
<td>EOM</td>
<td>end-of-memory</td>
</tr>
<tr>
<td>EOS</td>
<td>end of support</td>
</tr>
<tr>
<td>EOT</td>
<td>end-of-task</td>
</tr>
<tr>
<td>EPS</td>
<td>extended pointer set</td>
</tr>
<tr>
<td>EPVT</td>
<td>extended private</td>
</tr>
<tr>
<td>ESS</td>
<td>Enterprise Storage Server</td>
</tr>
<tr>
<td>ETO</td>
<td>Extended Terminal Option</td>
</tr>
<tr>
<td>EWLM</td>
<td>Enterprise Workload Manager</td>
</tr>
<tr>
<td>EX6</td>
<td>execution phase 6</td>
</tr>
<tr>
<td>EXE</td>
<td>execution</td>
</tr>
<tr>
<td>FDBR</td>
<td>Fast Database Recovery</td>
</tr>
<tr>
<td>FIXCAT</td>
<td>Fix Category</td>
</tr>
<tr>
<td>FMID</td>
<td>function modification ID</td>
</tr>
<tr>
<td>FPA</td>
<td>Fast Path Advanced</td>
</tr>
<tr>
<td>FPB</td>
<td>Fast Path Basic Tools</td>
</tr>
<tr>
<td>FPO</td>
<td>Fast Path Online</td>
</tr>
<tr>
<td>FPSI</td>
<td>Fast Path Secondary Index</td>
</tr>
<tr>
<td>FT3</td>
<td>file tailoring phase 3</td>
</tr>
<tr>
<td>GA</td>
<td>General Availability</td>
</tr>
<tr>
<td>GDG</td>
<td>generation data set group</td>
</tr>
<tr>
<td>GEF</td>
<td>Graphical Editor Framework</td>
</tr>
<tr>
<td>GN</td>
<td>Get Next</td>
</tr>
<tr>
<td>GNP</td>
<td>Get Next within Parent</td>
</tr>
<tr>
<td>GSAM</td>
<td>generalized sequential access method</td>
</tr>
<tr>
<td>GU</td>
<td>Get Unique</td>
</tr>
<tr>
<td>GUI</td>
<td>graphical user interface</td>
</tr>
<tr>
<td>HALDB</td>
<td>high availability large databases</td>
</tr>
<tr>
<td>HDAM</td>
<td>hierarchical direct access method</td>
</tr>
<tr>
<td>HDC</td>
<td>Hardware Data Compression</td>
</tr>
<tr>
<td>HDPC</td>
<td>HD Pointer Checker</td>
</tr>
<tr>
<td>HDTA</td>
<td>HD Tuning Aid</td>
</tr>
<tr>
<td>HIDAM</td>
<td>hierarchical indexed direct access method</td>
</tr>
<tr>
<td>HPIC</td>
<td>High Performance Image Copy</td>
</tr>
<tr>
<td>HPPC</td>
<td>High Performance Pointer Checker</td>
</tr>
<tr>
<td>HSSR</td>
<td>High Speed Sequential Retrieval</td>
</tr>
<tr>
<td>HTML</td>
<td>Hypertext Markup Language</td>
</tr>
<tr>
<td>HTTP</td>
<td>Hypertext Transfer Protocol</td>
</tr>
<tr>
<td>HTTPS</td>
<td>HTTP Secure</td>
</tr>
<tr>
<td>IBM</td>
<td>International Business Machines Corporation</td>
</tr>
<tr>
<td>IDE</td>
<td>integrated development environment</td>
</tr>
<tr>
<td>IDRC</td>
<td>improved data recording capability</td>
</tr>
<tr>
<td>IDp</td>
<td>identity provider</td>
</tr>
<tr>
<td>IETF</td>
<td>Internet Engineering Task Force</td>
</tr>
<tr>
<td>IFCC</td>
<td>interface control checks</td>
</tr>
<tr>
<td>ILDS</td>
<td>indirect list data set</td>
</tr>
<tr>
<td>ILK</td>
<td>indirect list key</td>
</tr>
<tr>
<td>ILS</td>
<td>isolated log sender</td>
</tr>
<tr>
<td>IMS</td>
<td>Information Management System</td>
</tr>
<tr>
<td>IPCS</td>
<td>Interactive Problem Control System</td>
</tr>
<tr>
<td>IPL</td>
<td>initial program load</td>
</tr>
<tr>
<td>IRLM</td>
<td>internal resource lock manager</td>
</tr>
<tr>
<td>IRM</td>
<td>IMS request message</td>
</tr>
<tr>
<td>IRUR</td>
<td>Internal Resource Usage Report</td>
</tr>
<tr>
<td>ISC</td>
<td>intersystem communication</td>
</tr>
<tr>
<td>ISPF</td>
<td>Interactive System Productivity Facility</td>
</tr>
<tr>
<td>IISRT</td>
<td>Insert</td>
</tr>
<tr>
<td>ISV</td>
<td>independent software vendor</td>
</tr>
<tr>
<td>ITKB</td>
<td>IMS Tools Knowledge Base</td>
</tr>
<tr>
<td>ITSO</td>
<td>International Technical Support Organization</td>
</tr>
<tr>
<td>IVP</td>
<td>Installation Verification Program</td>
</tr>
<tr>
<td>IVP鑫</td>
<td>IVP Export Utility</td>
</tr>
<tr>
<td>J2C</td>
<td>Java EE Connector Architecture</td>
</tr>
<tr>
<td>J2SE</td>
<td>Java 2 Platform, Standard Edition</td>
</tr>
<tr>
<td>JAR</td>
<td>Java archive</td>
</tr>
<tr>
<td>JBP</td>
<td>Java batch processing</td>
</tr>
<tr>
<td>JCA</td>
<td>Java Connector Architecture</td>
</tr>
<tr>
<td>JCC</td>
<td>Java Common Client</td>
</tr>
<tr>
<td>JCL</td>
<td>job control language</td>
</tr>
<tr>
<td>JDBC</td>
<td>Java Database Connectivity</td>
</tr>
<tr>
<td>JDK</td>
<td>Java Development Kit</td>
</tr>
<tr>
<td>JEE</td>
<td>Java Platform, Enterprise Edition</td>
</tr>
<tr>
<td>JMP</td>
<td>Java Message Program</td>
</tr>
<tr>
<td>JMS</td>
<td>Java Message Service</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>JNDI</td>
<td>Java Naming and Directory Interface</td>
</tr>
<tr>
<td>JRE</td>
<td>Java Runtime Environment</td>
</tr>
<tr>
<td>JSP</td>
<td>JavaServer Pages</td>
</tr>
<tr>
<td>JTA</td>
<td>Java Transaction API</td>
</tr>
<tr>
<td>JTS</td>
<td>Java Transaction Service</td>
</tr>
<tr>
<td>JVM</td>
<td>Java Virtual Machine</td>
</tr>
<tr>
<td>Java EE</td>
<td>Java Platform Enterprise Edition</td>
</tr>
<tr>
<td>KBLA</td>
<td>Knowledge-Based Log Analysis</td>
</tr>
<tr>
<td>KSDS</td>
<td>key-sequenced data set</td>
</tr>
<tr>
<td>LAN</td>
<td>local area network</td>
</tr>
<tr>
<td>LDAP</td>
<td>Lightweight Directory Access Protocol</td>
</tr>
<tr>
<td>LDS</td>
<td>linear data set</td>
</tr>
<tr>
<td>LIU</td>
<td>Library Integrity Utility</td>
</tr>
<tr>
<td>LLLL</td>
<td>length value</td>
</tr>
<tr>
<td>LPAR</td>
<td>logical partition</td>
</tr>
<tr>
<td>LSQA</td>
<td>local system queue area</td>
</tr>
<tr>
<td>LU</td>
<td>logical unit</td>
</tr>
<tr>
<td>LU2</td>
<td>logical unit 2</td>
</tr>
<tr>
<td>LUNAME</td>
<td>logic unit name</td>
</tr>
<tr>
<td>MAS</td>
<td>multiple access spool</td>
</tr>
<tr>
<td>MCI</td>
<td>Message Control Information</td>
</tr>
<tr>
<td>MDB</td>
<td>message-driven bean</td>
</tr>
<tr>
<td>MFBP</td>
<td>message format buffer pool</td>
</tr>
<tr>
<td>MFS</td>
<td>Message Format Service</td>
</tr>
<tr>
<td>MOD</td>
<td>message output descriptor</td>
</tr>
<tr>
<td>MPP</td>
<td>message processing program</td>
</tr>
<tr>
<td>MR</td>
<td>Manage Resources</td>
</tr>
<tr>
<td>MSC</td>
<td>Multiple Systems Coupling</td>
</tr>
<tr>
<td>MSDB</td>
<td>main storage database</td>
</tr>
<tr>
<td>MTO</td>
<td>master terminal operator</td>
</tr>
<tr>
<td>MVS</td>
<td>Multiple Virtual System</td>
</tr>
<tr>
<td>NAK</td>
<td>negative acknowledgment</td>
</tr>
<tr>
<td>ODBA</td>
<td>Open Database Access</td>
</tr>
<tr>
<td>ODBC</td>
<td>Open Database Connectivity</td>
</tr>
<tr>
<td>ODBM</td>
<td>Open Database Manager</td>
</tr>
<tr>
<td>OLC</td>
<td>online change</td>
</tr>
<tr>
<td>OLDS</td>
<td>online data sets</td>
</tr>
<tr>
<td>OLDS</td>
<td>online log data set</td>
</tr>
<tr>
<td>OLR</td>
<td>Online Reorganization</td>
</tr>
<tr>
<td>OM</td>
<td>Operations Manager</td>
</tr>
<tr>
<td>OO</td>
<td>object-oriented</td>
</tr>
<tr>
<td>OSAM</td>
<td>overflow sequential access method</td>
</tr>
<tr>
<td>OTE</td>
<td>Open Transaction Environment</td>
</tr>
<tr>
<td>OTMA C/I</td>
<td>OTMA Callable Interface</td>
</tr>
<tr>
<td>OTMA</td>
<td>Open Transaction Manager Access</td>
</tr>
<tr>
<td>OTT</td>
<td>Open Thread TCB</td>
</tr>
<tr>
<td>PA</td>
<td>Performance Analyzer</td>
</tr>
<tr>
<td>PC</td>
<td>personal computer</td>
</tr>
<tr>
<td>PCB</td>
<td>program communication block</td>
</tr>
<tr>
<td>PCBs</td>
<td>program control blocks</td>
</tr>
<tr>
<td>PDIR</td>
<td>program directory</td>
</tr>
<tr>
<td>PDS</td>
<td>partitioned data set</td>
</tr>
<tr>
<td>PDU</td>
<td>Partition Definition Utility</td>
</tr>
<tr>
<td>PHIDAM</td>
<td>partitioned HIDAM</td>
</tr>
<tr>
<td>PI</td>
<td>problem investigator</td>
</tr>
<tr>
<td>PPT</td>
<td>program properties table</td>
</tr>
<tr>
<td>PRA</td>
<td>parallel RECON access</td>
</tr>
<tr>
<td>PSB</td>
<td>program specification block</td>
</tr>
<tr>
<td>PSP</td>
<td>Preventive Service Planning</td>
</tr>
<tr>
<td>PSSR</td>
<td>Physical Sequence Sort for Reload</td>
</tr>
<tr>
<td>PTF</td>
<td>program temporary fix</td>
</tr>
<tr>
<td>QoS</td>
<td>Quality of Service</td>
</tr>
<tr>
<td>RACF</td>
<td>Resource Access Control Facility</td>
</tr>
<tr>
<td>RAD</td>
<td>Rational Application Developer</td>
</tr>
<tr>
<td>RAR</td>
<td>resource adapter archive</td>
</tr>
<tr>
<td>RAS</td>
<td>Resource Access Security</td>
</tr>
<tr>
<td>RBA</td>
<td>relative byte address</td>
</tr>
<tr>
<td>RDDS</td>
<td>resource definition data set</td>
</tr>
<tr>
<td>RDS</td>
<td>repository data set</td>
</tr>
<tr>
<td>RDS</td>
<td>restart data set</td>
</tr>
<tr>
<td>RECON</td>
<td>recovery control</td>
</tr>
<tr>
<td>RLDS</td>
<td>recovery log data set</td>
</tr>
<tr>
<td>RM</td>
<td>Resource Manager</td>
</tr>
<tr>
<td>RMM</td>
<td>Request MOD Message</td>
</tr>
<tr>
<td>RRDS</td>
<td>relative record data set</td>
</tr>
<tr>
<td>RRS</td>
<td>Resource Recovery Services</td>
</tr>
<tr>
<td>RRS/MVS</td>
<td>Resource Recovery Services/MVS</td>
</tr>
<tr>
<td>RS</td>
<td>Repository Server</td>
</tr>
<tr>
<td>RSM</td>
<td>request status message</td>
</tr>
<tr>
<td>RSR</td>
<td>Remote Site Recovery</td>
</tr>
<tr>
<td>RVA</td>
<td>RAMAC Virtual Array</td>
</tr>
<tr>
<td>RYO</td>
<td>Roll-Your-Own</td>
</tr>
<tr>
<td>SAF</td>
<td>System Authorization Facility</td>
</tr>
<tr>
<td>SAML</td>
<td>Security Assertion Markup Language</td>
</tr>
<tr>
<td>SAS</td>
<td>secondary address space</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
</tr>
<tr>
<td>SCI</td>
<td>Structured Call Interface</td>
</tr>
<tr>
<td>SDEP</td>
<td>sequential dependent segment</td>
</tr>
<tr>
<td>SDK</td>
<td>software development kit</td>
</tr>
<tr>
<td>SEVT</td>
<td>structure event trace table</td>
</tr>
<tr>
<td>SGML</td>
<td>Standard Generalized Markup Language</td>
</tr>
<tr>
<td>SHISAM</td>
<td>simple hierarchical sequential access method</td>
</tr>
<tr>
<td>SLA</td>
<td>service level agreement</td>
</tr>
<tr>
<td>SLDS</td>
<td>system log data set</td>
</tr>
<tr>
<td>SLSB</td>
<td>stateless session bean</td>
</tr>
<tr>
<td>SMB</td>
<td>scheduler message block</td>
</tr>
<tr>
<td>SMF</td>
<td>System Management Facilities</td>
</tr>
<tr>
<td>SMP/E</td>
<td>System Modification Program/Extended</td>
</tr>
<tr>
<td>SMQ</td>
<td>shared message queue</td>
</tr>
<tr>
<td>SMU</td>
<td>Security Maintenance Utility</td>
</tr>
<tr>
<td>SNA</td>
<td>Systems Network Architecture</td>
</tr>
<tr>
<td>SOA</td>
<td>service-oriented architecture</td>
</tr>
<tr>
<td>SOMA</td>
<td>Service Oriented Modeling and Architecture</td>
</tr>
<tr>
<td>SPE</td>
<td>small programming enhancement</td>
</tr>
<tr>
<td>SPI</td>
<td>system programming interface</td>
</tr>
<tr>
<td>SPMN</td>
<td>Space Monitor</td>
</tr>
<tr>
<td>SPOC</td>
<td>single point of control</td>
</tr>
<tr>
<td>SQ</td>
<td>shared queues</td>
</tr>
<tr>
<td>SQLJ</td>
<td>SQL for Java</td>
</tr>
<tr>
<td>SRB</td>
<td>service request block</td>
</tr>
<tr>
<td>SREL</td>
<td>same System Release</td>
</tr>
<tr>
<td>SSA</td>
<td>segment search argument</td>
</tr>
<tr>
<td>SSL</td>
<td>Secure Sockets Layer</td>
</tr>
<tr>
<td>SSPM</td>
<td>sysplex serialized program management</td>
</tr>
<tr>
<td>STE</td>
<td>storage tracking element</td>
</tr>
<tr>
<td>STR</td>
<td>structure trace table</td>
</tr>
<tr>
<td>STSN</td>
<td>set and test sequence numbers</td>
</tr>
<tr>
<td>SVL</td>
<td>Silicon Valley Laboratories</td>
</tr>
<tr>
<td>SXPL</td>
<td>Standard User Exit Parameter List</td>
</tr>
<tr>
<td>SYSDEF</td>
<td>system definition</td>
</tr>
<tr>
<td>SYSID</td>
<td>system identifier</td>
</tr>
<tr>
<td>TCB</td>
<td>task control block</td>
</tr>
<tr>
<td>TCB</td>
<td>thread task control block</td>
</tr>
<tr>
<td>TCO</td>
<td>Time-Controlled Operations</td>
</tr>
<tr>
<td>TCP/IP</td>
<td>Transmission Control Protocol/Internet Protocol</td>
</tr>
<tr>
<td>TM</td>
<td>Transaction Manager</td>
</tr>
<tr>
<td>TMRA</td>
<td>Transaction Manager Resource Adapter</td>
</tr>
<tr>
<td>TMS</td>
<td>Transport Manager System</td>
</tr>
<tr>
<td>TOSI</td>
<td>Tools Online System Interface</td>
</tr>
<tr>
<td>TPIPE</td>
<td>transaction pipe</td>
</tr>
<tr>
<td>TSO</td>
<td>Time Sharing Option</td>
</tr>
<tr>
<td>TTL</td>
<td>time to live</td>
</tr>
<tr>
<td>UI</td>
<td>user interface</td>
</tr>
<tr>
<td>UML</td>
<td>Unified Modeling Language</td>
</tr>
<tr>
<td>UOR</td>
<td>Unit of Recovery</td>
</tr>
<tr>
<td>UOW</td>
<td>unit of work</td>
</tr>
<tr>
<td>VIPA</td>
<td>Virtual IP Addressing</td>
</tr>
<tr>
<td>VSAM</td>
<td>Virtual Storage Access Method</td>
</tr>
<tr>
<td>VTAM</td>
<td>Virtual Telecommunications Access Method</td>
</tr>
<tr>
<td>W3C</td>
<td>World Wide Web Consortium</td>
</tr>
<tr>
<td>WADS</td>
<td>write-ahead data set</td>
</tr>
<tr>
<td>WAN</td>
<td>wide area network</td>
</tr>
<tr>
<td>WFI</td>
<td>wait for input</td>
</tr>
<tr>
<td>WLM</td>
<td>workload manager</td>
</tr>
<tr>
<td>WSDL</td>
<td>Web Services Description Language</td>
</tr>
<tr>
<td>WTOR</td>
<td>write to operator with reply</td>
</tr>
<tr>
<td>WWW</td>
<td>World Wide Web</td>
</tr>
<tr>
<td>XCF</td>
<td>cross-system coupling facility</td>
</tr>
<tr>
<td>XML</td>
<td>XML Metadata Interchange</td>
</tr>
<tr>
<td>XMI</td>
<td>Extensible Markup Language</td>
</tr>
<tr>
<td>zAAP</td>
<td>z Application Assist Processor</td>
</tr>
<tr>
<td>zIIP</td>
<td>z9 Integrated Information Processors</td>
</tr>
<tr>
<td>zIIP</td>
<td>z9 Integrated Information Processors</td>
</tr>
</tbody>
</table>
Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

IBM Redbooks

The following IBM Redbooks publications provide additional information about the topic in this document. Some publications referenced in this list might be available in softcopy only.

- IBM CICS Scalability: New Features in V4.2, REDP-4787
- Federated Identity Management and Web Services Security with IBM Tivoli Security Solutions, SG24-6394
- IMS 11 Open Database, SG24-7856
- IMS Version 11 Technical Overview, SG24-7807
- IBM z/OS V1R12 Communications Server TCP/IP Implementation: Volume 4 Security and Policy-Based Networking, SG24-7899

You can search for, view, download or order these documents and other Redbooks, Redpapers, Web Docs, draft and additional materials, at the following website:

ibm.com/redbooks

Other publications

These publications are also relevant as further information sources:

- IMS and SOA Executive Overview, GC19-2516
- IMS High Performance Fast Path Utilities for z/OS, SC18-9869-04
- IMS Version 12 Application Programming, SC19-3007
- IMS Version 12 Application Programming APIs, SC19-3008
- IMS Version 12 Commands, Volume 1: IMS Commands A-M, SC19-3009
- IMS Version 12 Commands, Volume 2: IMS Commands N-V, SC19-3010
- IMS Version 12 Commands, Volume 3: IMS Component and z/OS Commands, SC19-3011
- IMS Version 12 Communications and Connections, SC19-3012
- IMS Version 12 Database Administration, SC19-3013
- IMS Version 12 Database Utilities, SC19-3014
- IMS Version 12 Diagnosis, GC19-3015
- IMS Version 12 Exit Routines, SC19-3016
- IMS Version 12 Installation, GC19-3017
IMS Version 12 License Programming Specifications, GC19-3024
IMS Version 12 Messages and Codes, Volume 1: DFS Messages, GC18-9712
IMS Version 12 Messages and Codes, Volume 2: Non-DFS Messages, GC18-9713
IMS Version 12 Messages and Codes, Volume 3: IMS Abend Codes, GC19-9714
IMS Version 12 Messages and Codes, Volume 4: IMS Component Codes, GC19-9715
IMS Version 12 Operations and Automation, SC19-3018
IMS Version 12 Program Directory, GI10-8843
IMS Version 12 Release Planning, GC19-3019
IMS Version 12 System Administration, SC19-3020
IMS Version 12 System Definition, GC19-3021
IMS Version 12 System Programming APIs, SC19-3022
IMS Version 12 System Utilities, SC19-3023

Online resources

These web sites are also relevant as further information sources:

- IBM Information Management Software for z/OS Solutions Information Center
- Whitepaper WebSphere MQ for z/OS and IMS Transaction Expiration
- IMS Tools Product Page
- Product Lifecycle for DB2 and IMS Tools

Help from IBM

IBM Support and downloads
ibm.com/support

IBM Global Services
ibm.com/services
Index

Symbols
/DIAG SNAP command 30
/DIAG SNAP LINE() command 39
/DIAGNOSE command
    APPC client 31
/DIAGNOSE SET AOSLOG command 38, 123

Numerics
64-bit storage 45–46, 97, 315

A
abend elimination 103
ACB (application control block) 94, 107, 310
caching 10, 315
    library 6, 25, 27, 94, 108, 315
active 27
    IMS 12 310
    individual members 27
member 28, 107
    online change 27
    online change function 27
online change 25, 27–28
    considerations 30
ACBGEN utility 60
ACBLIB 25, 27, 108
    IMS 12 310
    individual members 27
library built with IMS 10 or IMS 11 310
member migration in 64-bit memory 315
members 27
migration 108
on-line change 94
usability enhancements 6
access control environment element (ACEE) 5, 126
catched 128
ACEE (access control environment element) 5, 126
catched 128
ACK 120, 329
active IMS
    register 49
    system 320
ACTIVE-RRS 112
address space 30–31, 37, 68, 201–202, 284, 302, 312, 316, 320, 379
    control block 37
    secondary block 37
    severe storage conditions 64
startup JCL 50
startup procedure JCL 209
Advanced Program-to-Program Communication (APPC) 156
    client to issue the /DIAGNOSE command 31
    message type 311
    OTMA shared queues 5
    OTMA shared queues enhancement 110
    AIB (application interface block) 314
    AOI (Automated Operator Interface) program 31
    AOIP parameter 310
    AOP API 12
    AOS (APPC/OTMA SMQ) 110
        enablement 114
        local status 111
        ACT-REXCF 112
        FORCE-RRS 112
    SCD Extension control block 31
API
    AOP 12
    DBRC 142, 313
    DBRC Command request 317
    OM 5, 14, 240
    REXX SPOC 14
    APPC (Advanced Program-to-Program Communication)
        client to issue the /DIAGNOSE command 31
        message type 311
        OTMA shared queues 5
        enhancement 110
        synchronous conversations 156
    APPC/OTMA SMQ (AOS) 110
        enablement 114
        local status 111
        ACT-REXCF 112
        FORCE-RRS 112
    SCD Extension control block 31
    application control block (ACB) 94, 107, 310
        caching 10, 315
        library 6, 25, 27, 94, 108, 315
        active 27
        IMS 12 310
        individual members 27
        member 28, 107
        online change 27
        online change function 27
        online change 25, 27–28
        considerations 30
    application interface block (AIB) 314
    application program 19–20, 58, 84, 89–90, 107, 124, 133, 162, 252, 285–286, 302
        IMS resource in DRD-enabled system 15
    application programming interface (API) 12
    application transparent-transport layer security (AT-TLS) 319, 360, 449
    ARLN 443
    AT-TLS (application transparent-transport layer security) 319, 360, 449
    audit level 231
    Automated Operator Interface (AOI) program 31
    automatic export 21, 233
back-end system 110–111
background information 12, 17, 137
Base Primitive Environment (BPE) 9, 47–48, 189, 201, 312, 442
configuration 51, 189, 207
parameter member 51
exit list 50
PROCLIB member 52
Basic Sequential Access Method (BSAM) 43
batch job 64, 76, 103, 138, 229, 362, 368
JOB statement 104
BPE (Base Primitive Environment) 9, 47–48, 189, 201, 312, 442
configuration 51, 189, 207
parameter member 51
PROCLIB member 52
exit list 50
BPECFG 50
BSAM (Basic Sequential Access Method) 43
buffer pool 57–58, 68
background 58
BUFFPOOLS macro 58
SASPSB parameter 61

CA (change accumulation) 139, 148, 151
data set 146, 149, 286
CA group 146
GRPIVP 149
last CA execution record 146
record 148
CBPDO (Custom-built Product Delivery Option) 280, 309, 331
CCB (conversational control block) 34
central processor complex (CPC) 284
certificate revocation list (CRL) 360
CF (coupling facility) 104
change accumulation (CA) 139, 151
data set 146, 149, 286
many versions 148
CHANGE.CAGR P
command 148
command syntax 148
CHANGE.DB command 313
CHANGE.DBDS command 313
CHANGE.RECON
CMDAUTH 317
MINVERS command 313
syntax 158
Upgrade 156, 317
UPGRADE CHECKUP command 160
UPGRADE command 160, 317
CHANGE.RECON 152, 283
CHANGE.RECON MINVERS command 313
CHNG 162–163
CI size 97
CIB (communication interface block) 34
CK field 84–85
CLB (communication line block) 34
CLEANUP.RECON
CAGRANGE command 146
command 140, 145–146, 316–317
command syntax 146
resource 316
client application 119–120, 327
different types 348
TCP/IP communications 329
CLLE (common latch list element) 38
CMDP parameter 310
CNT (communication name table) 34
COBOL 441
coeexistence 137, 280–281
IMS 12 281
cold start 15, 108, 162, 283
descriptor definitions 17
IMS 171
command
IMS 225, 233, 285
command codes 106
command processor 327
commands
IMS 12 14, 33, 123, 144, 230, 313
commit-then-send 130
common latch list element (CLLE) 38
Common Queue Server (CQS) 4, 9, 26, 47–48, 284
enhancements in IMS 12 51
structure
event 51
overflow event 52
trace record 50
Common Service Layer (CSL) 12, 20, 200, 204, 284, 329
address space 204
section 20, 208
common storage area (CSA) 59–61, 68
communication
IMS 12 119, 330
IMS to IMS 162
communication interface block (CIB) 34
communication line block (CLB) 34
communication name table (CNT) 34
communication terminal block (CTB) 34
Communication Translate Table (CTT) 34
communications restart block (CRB) 34
COMPID (component ID) 281
completion code 25
12A 28
12B 28
12C 28
component ID (COMPID) 281
concatenated key 74
configuration 8, 48, 50, 138, 164, 169, 207, 431
configuration member 128, 191, 207, 303, 319
connection bundle 366
control block 28, 98, 108, 123, 310
Control Center 14
control region 14, 156, 284
parameter 115
conversational 116, 134
conversational control block (CCB) 34
correlator file 374
coupling facility (CF) 104
CPC (central processor complex) 284
CQS (Common Queue Server) 4, 9, 26, 47–48, 284
enhancements in IMS 12 51
structure
  event 51
  overflow event 52
trace record 50
CQS trace record 50
CRB (communications restart block) 34
CREATE OTMADESC 163
CRL (certificate revocation list) 360
cross-system coupling facility (XCF) 4, 110, 156, 188, 201, 285, 319
  communication 119
  group 208
  name 208, 215
  name parameter value 216
  indicator 111
  input transaction message 119
CSA (common storage area) 59–61, 68
CSL (Common Service Layer) 12, 20, 200, 204, 284, 329
  address space 204
  section 20, 208
CSLOIxxx 210
CSLRIxxx 208
CSLSIxxx 209
CSLURP10 222
CTB (communication terminal block) 34
CTT (Communication Translate Table) 34
custom IMS 324
Custom-built Product Delivery Option (CBPDO) 280, 309, 331
cylinder address
  65,519 54
  65,520 54
cylinder-managed space 54

D
DASD volume
  characteristic 53
  growth 53
data
  IMS 2, 304
data entry database (DEDB) 6, 33, 76–77
    area
      control block 33
      name list entry 33
data management block (DMB) 25, 34, 68
    number 108
data management block list (DMBL) 36
data set 25, 43, 68, 73, 137–138, 205, 284, 286
    DD names 84
    following types 54
    minimum number 44
    volume serial information 151
data sharing 7, 77
data space mapping entry (DSME) 33
data type 45, 327
  reference information 327
database 1, 11, 17, 19, 68, 138, 186, 247, 283, 285
    IMS 2, 105, 107, 311, 314
  quiesce 313
Database Control (DBCTL) 6, 17, 31, 33, 105
  environment 34
database description (DBD) 27, 142
Database Recovery
  utility 76
database recovery job 286
Database Recovery Control (DBRC) 7, 74, 137, 283, 442
  API 142, 313
  Command API request 317
  command authorization 317
  commands 77, 137, 313, 316
  enhancements 156
  IMS 12 140
  utility 138, 316
Database Reorg Expert 396
Database Repair Facility 393
DATABASE section 315
Database Solution Pack for z/OS 394
DB/DC 17, 31, 33–34
DB2 2, 107, 126, 286, 310
DBBF 97
DBCTL (Database Control) 6, 17, 31, 33, 105
  environment 34
DBD (database description) 27, 142
DBDGEN for secondary index 83
DBFX 97
DBRC (Database Recovery Control) 7, 74, 137, 283, 442
  API 142, 313
  Command API request 317
  command authorization 317
  commands 77, 137, 313, 316
  enhancements 156
  IMS 12 140
  utility 138, 316
DCCTL 17, 31, 34
DCCTL environment 34
DD statement 20, 284, 315
STORCLAS parameter 46
DDNAMES 73
DEDB (data entry database) 6, 33, 76–77
  area
    control block 33
    name list entry 33
DEDB master control block (DMCB) 33
definitional change 200
DELETE command 17, 19, 21, 140, 163–164, 227
DELETE.ALLOC command 313
DELETE.LOG command 313
dependent region 9, 30–32, 119–120
  ASID 65
directory block 37
descriptor creation 22
descriptor definition 15, 17, 19, 214
EAS (extended address space) 53
EAV (extended address volume) 11, 54
  support 54, 57
EAV support 55
ECB (event control block) 34
ECNT (extended communication name table) 34
ECSA (extended common service area) 3, 41, 45–46, 97–98
EMH (Expedited Message Handler) 49
EMHB (Expedited Message Handler Block) parameter 310
end-of-memory (EOM) 64
  EOT trace 65
end-of-task (EOT) 64
  trace 65
Enterprise Suite V2.1 contents 324
environment 2, 17, 110, 455
EOM (end-of-memory) 64
  EOT trace 65
EOT (end-of-task) 64
  trace 65
EPS (extended pointer set) 76
ETR (external throughput rate) 125
event control block (ECB) 34
exit 5, 20, 41, 76–78, 109, 118, 193, 255, 284, 443
calls 50, 77, 106, 133
routine 78, 83, 91, 132, 284–285
user 9, 50, 91, 132, 256, 284–285
Exit Interface Block Data Store (XIBDS) 319
exit routine 5, 50, 83, 92, 118, 132, 285, 308–309, 316
  IMS 12 119, 316
EXITDEF parameter 318
Expedited Message Handler (EMH) 314
Expedited Message Handler Block (EMHB) 34
  parameter 310
EXPORT command 21, 215, 221
  panels 264
export data 291
EXPRT command 21
  DEFN command 21
target 220
exporting resource 253, 256
extended address space (EAS) 53
extended address volume (EAV) 11, 54
  support 54, 57
extended common service area (ECSA) 3, 41, 45–46, 97–98
extended communication name table (ECNT) 34
extended pointer set (EPS) 76
external recovery facility (XRF) 17, 30, 138
External throughput rate (ETR) 125
external throughput rate (ETR) 125

**F**
Fast Database Recovery (FDBR) 17
Fast Path 5, 15, 18, 20, 67, 76, 78, 80, 134, 311, 313
  buffer pool 97
Fast Path Secondary Index (FPSI) 95
Fast Path Solution Pack for z/OS 390
FDBR 30, 284
FDBR (Fast Database Recovery) 17
FIELD statement 83

first time
  adding an MSC link 184
  DRD function 21
  IMS 12 IVP dialog 292
five-numeric character 60
FMID (function modification ID) 281
FPSI (Fast Path Secondary Index) 95
FPWP parameter 310
front-end IMS
  interact 116
  system 125
full function database buffer pools 68
full segment logging option 77
FULLSEG 77
function modification ID (FMID) 281

**G**
GDG (generation data set group) 51
generation data set group (GDG) 51
given PSB
  work area size 60
global online change function changes resources online 25
GRPMAX value 148
GSAM database 285

**H**
HALDB (high availability large database) 5, 72–73, 152, 238
Hardware Compression Extended 390
Hardware Data Compression (HDC) 309
HDAM (hierarchical direct access method) 5, 67–68
HDC (Hardware Data Compression) 309
HIDAM (hierarchical indexed direct access method) 5, 68, 74
hierarchical direct access method (HDAM) 5, 67–68
hierarchical indexed direct access method (HIDAM) 5, 68, 74
high availability large database (HALDB) 72
high availability large databases (HALDB) 5, 73, 152, 238
High Performance Fast Path Utilities 391
High Performance Image Copy 392, 394
High Performance Load 398
High Performance Pointer Checker 395
High Performance Prefix Resolution 399
High Performance Unload 397

high-order bit 157
HiOP parameter 310
HPFPU Build Index function (INDEXBLD) 95
HWS 128, 319
HWSCFGxx 319
HWSCFGxx member 319
HWSIMSO0 exit routine 309, 319
HWSIMSO1 exit routine 309, 319
HWSJAVA0 exit routine 194
HWSP1410W message 319
HWSRCORD 189, 319
HWSSMPL0 319
HWSSMPL0 exit routine 194, 309
HWSSMPL1 319
HWSSMPL1 exit routine 194, 309
HWSUINIT exit routine 194

**I**
IBM Tools Base for z/OS 386
ICONTR 319
IDE (interactive development environment) 324
ILDLS (indirect list data set) 76
ILK (indirect list key) 84
ILOGREC 285
ILS (isolated log sender) 320
image copy 139, 286, 312
  successful completion 142
IMPORT command 15–16, 21, 219
  considerations 17
  panels 267
IMPORT DEFN command 12, 251
  source 251
IMPORT DFN command 255
IMPORT UPDATE command 15
imported definition 15, 252
IMS 10 8, 57, 64, 74, 129, 156, 282, 313, 324
change accumulation data sets 286
default values 320
IMS 12 284, 316
upgrade processing 157
IMS 11 8, 57, 69, 127, 132, 145, 156, 189, 280, 282, 313, 324
  64-bit Fast Path buffer manager 98
  archive job 160
database 313
DBRC 283
DFSAAPL menu 288
DLIBATCH 285
emergency 98
Enterprise Suite V2.1 324
environment 129
IVP 287, 289
IVP dialog 293
IVP variables 287
level 312
message queue 312
path name 318
RECON data 316
RECON upgrade 159
system 156, 283, 299, 316
upgrade processing   157
value   299
work   284–285
IMS 11 IVP
dialog   288
phase selection menu   290
variable   293
IMS 12   14, 21, 110, 139, 163, 324
CBPDO   280
changed log records   285
coeistence   281
command   14, 33, 123, 144, 230, 313
communication   119, 330
CQS enhancements   51
DBRC   140
diagnosis   316
DRD enhancement   21
EAV support   54
Enterprise Suite V2.1   324
exit routine   119, 316
explicit migration considerations   314
full function database buffer pools   68
host z/OS system   310
installation   308
message   124
new function   190
new support   170
object modules   309
program directory   308, 318
PSP upgrade name   308
RACF return code   330
release planning   308, 313
RRS dependency   110
SPOC enhancements   14
system administration   207–208, 308, 310
system definition   59, 62, 113, 209, 213–214, 221
UPDATE option   12
IMS 12.01.00
Database Manager   280
DB Level Tracking   280
Recovery Level Tracking   280
Transaction Manager   280
IMS 12.1   312
IMS 8   110
IMS Application
menu   12, 19, 263, 304
IMS application   130, 263, 324
callout request   348, 364
developer   324
development task   324
program   130–131, 134, 329
IMS Connect   2, 115, 126, 161–162, 284, 431, 435
commands   5, 165, 175, 319
IMS Control Center   14
IMS Database Recovery Control facility   138
IMS Database Recovery Facility   95
IMS Database Solution Pack for z/OS   386
IMS Dump Formatter   51
IMS Enterprise Suite   9, 311, 323, 328
reference information   328
SOAP Gateway
Project   374
V2.1   324
IMS Explorer
graphical editor   333
project   333
IMS Fast Path Solution Pack for z/OS   386
IMS Hardware Compression Extended   390
IMS High Performance Fast Path Utilities   95
IMS Performance Solution Pack for z/OS   386
IMS Queue Control Facility for z/OS (QCF)   312
IMS Recovery Solution Pack for z/OS   386
IMS request message (IRM)   327
IMS resource
definition
repository   21
IMS Single Point   165
IMS SOAP Gateway   449
IMS Tools Common services   390
IMS Tools Distributed Access Infrastructure   390
IMS Tools Knowledge Base   390
IMS.PROCLIB   26, 162, 168
IMSID   26, 184, 187, 220
IMSPlex   12, 115, 171, 203–204, 314, 443
name   196
IMSpex   3, 12, 111, 115, 200–202, 284, 318, 435
environment   25, 27
migration   318
IMSPLEX Name   240
IMSRSC repository   15, 199, 215, 287
administrative functions   230
non-system RDDS   228
inactive IMS
system   240
Index Builder   398
Index Builder tool   76
INDEXBLD function   95
indirect list data set (ILDS)   76
indirect list key (ILK)   84
individual OLDS   43
indoubt work   255
INIT.OLC PHASE(COMMIT)   29
INIT.OLC PHASE(PREPARE) TYPE(ACBMBR)   28
INITIATE   24
INITIATE OLC
command   25, 29
command master   25
phase   25
input message   116
installation
IMS   12   308
Installation Verification Program (IVP)   115, 286–287, 436, 443
application   275
job, additional documentation   300
installing   280, 283
interactive development environment (IDE)   324
Interactive Problem Control System (IPCS)   64
verb exit   64
Interactive System Productivity Facility (ISPF)   14, 190,
Internal Resource Lock Manager (IRLM) 6, 102–103, 280–281, 436
internal throughput rate (ITR) 125
intersystem communication (ISC) 311
IOPCB 116
nor message switch 110, 195, 280
reply 116, 118
IOPCB nor message switch 280
IP address 169, 360
IPCS (Interactive Problem Control System) 64
verb exit 64
IRLM (Internal Resource Lock Manager) 6, 102–103, 280–281, 436
IRM (IMS request message) 327
ISC (intersystem communication) 311
isolated log sender (ILS) 320
ISPF (Interactive System Productivity Facility) 14, 190, 263, 288
ITR (internal throughput rate) 125
IVP (Installation Verification Program) 115, 286–287, 436, 443
application 275
job, additional documentation 300
IVP dialog 287–288
execution phase 300
IVP Variable Export Utility 309
J
Java 8, 280–281, 311
IMS 281
Java batch processing (JBP) 325
Java Database Connectivity (JDBC) 2, 107
call 107
driver 107
Java EE Connector Architecture (JCA) 311, 314
Java message processing (JMP) 325
Java Messaging Service (JMS) 325
Java technology 381
JBP (Java batch processing) 325
JCA (Java EE Connector Architecture) 311, 314
JCL (job control language) 8, 14, 26, 41, 75, 96, 108, 137–138, 171, 189, 194, 206, 208, 284, 291, 301, 442–443
member CAJCL 151
sample 46, 159
skeletal 138
execution member 143
member CAJCL 151
statement 241
JDBC (Java Database Connectivity) 2, 107
call 107
driver 107
JMP (Java message processing) 325
JMS (Java Messaging Service) 325
job control language (JCL) 8, 14, 26, 41, 75, 96, 108, 137–138, 171, 189, 194, 206, 208, 284, 291, 301, 442–443
member CAJCL 151
sample 46, 159
skeletal 138
execution member 143
member CAJCL 151
statement 241
K
key-sequenced data set (KSDS) 202
KSDS (key-sequenced data set) 202
L
Language Environment 21
LCB (link control block) 35
LCHILD statement 82, 93
multiple database names 91
LCLICON 170
LCLPLKID 170
level flow 115
Library Integrity Utility 393, 395
link control block (LCB) 35
link link block (LLB) 35
link name 177
link terminal block (LTB) 35
LLB (link link block) 35
load
module 30
local IMS 25, 164
corresponding MSC configuration statement 183
definition information 234–235
MSC definition 187
resource definition 234
system IMS2 169
local SID value 256
format 310
IMS 12 285
logical relationships 68, 74, 333
LPAR 8, 201, 283, 319, 432, 436
LTB (link terminal block) 35
LUMC parameter 310
LUMP parameter 310
M
main storage database (MSDB) 34, 313
maintenance 31, 106, 139, 308, 455
Manage Resources (MR) 263
application 265
DEFN panel 272
QUERY command panels 274
master RS 201
MDA (members for dynamic allocation) 47
MDB (message-driven bean) 382
members for dynamic allocation (MDA) 47
message format buffer pool (MFBP) 63
<table>
<thead>
<tr>
<th>Term</th>
<th>Page(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message Format Service (MFS)</td>
<td>25</td>
</tr>
<tr>
<td>message processing program (MPP)</td>
<td>379</td>
</tr>
<tr>
<td>message queue data sets</td>
<td>57, 311</td>
</tr>
<tr>
<td>message-driven bean (MDB)</td>
<td>382</td>
</tr>
<tr>
<td>messages</td>
<td></td>
</tr>
<tr>
<td>IMS 12</td>
<td>124</td>
</tr>
<tr>
<td>metadata</td>
<td>323</td>
</tr>
<tr>
<td>method call</td>
<td>330, 384</td>
</tr>
<tr>
<td>MFBP (message format buffer pool)</td>
<td>63</td>
</tr>
<tr>
<td>MFS (Message Format Service)</td>
<td>25</td>
</tr>
<tr>
<td>migration</td>
<td>5, 47, 74, 110–111, 137, 194, 200, 215, 279, 282</td>
</tr>
<tr>
<td>consideration</td>
<td>113, 308, 313</td>
</tr>
<tr>
<td>to repository</td>
<td>205</td>
</tr>
<tr>
<td>MINVERS</td>
<td>114, 156, 312, 443</td>
</tr>
<tr>
<td>MODBLKS</td>
<td>17, 20, 199–200</td>
</tr>
<tr>
<td>MODBLKS data</td>
<td>171, 200, 221</td>
</tr>
<tr>
<td>MODBLKS resource</td>
<td>200</td>
</tr>
<tr>
<td>mode</td>
<td>44, 110, 130, 138, 316</td>
</tr>
<tr>
<td>movement between queue</td>
<td>109</td>
</tr>
<tr>
<td>MPORT</td>
<td>21</td>
</tr>
<tr>
<td>MPP</td>
<td>111</td>
</tr>
<tr>
<td>MPP (message processing program)</td>
<td>379</td>
</tr>
<tr>
<td>MQ</td>
<td></td>
</tr>
<tr>
<td>message</td>
<td></td>
</tr>
<tr>
<td>flow</td>
<td>135</td>
</tr>
<tr>
<td>level</td>
<td>135</td>
</tr>
<tr>
<td>MR (Manage Resources)</td>
<td>263</td>
</tr>
<tr>
<td>application</td>
<td>265</td>
</tr>
<tr>
<td>DEFN panel</td>
<td>272</td>
</tr>
<tr>
<td>QUERY command panels</td>
<td>274</td>
</tr>
<tr>
<td>MSC (Multiple Systems Coupling)</td>
<td>4, 132, 161, 170, 256, 284–285, 436</td>
</tr>
<tr>
<td>generic link</td>
<td>185</td>
</tr>
<tr>
<td>generic support</td>
<td>183</td>
</tr>
<tr>
<td>link</td>
<td>170</td>
</tr>
<tr>
<td>new status</td>
<td>178</td>
</tr>
<tr>
<td>local IMS</td>
<td>183, 187</td>
</tr>
<tr>
<td>TCP/IP</td>
<td>170</td>
</tr>
<tr>
<td>MSDB (main storage database)</td>
<td>34, 313</td>
</tr>
<tr>
<td>MSPLINK</td>
<td>4, 170</td>
</tr>
<tr>
<td>multiple field</td>
<td>82</td>
</tr>
<tr>
<td>multiple IMS</td>
<td></td>
</tr>
<tr>
<td>address space</td>
<td>14</td>
</tr>
<tr>
<td>subsystem</td>
<td>48</td>
</tr>
<tr>
<td>system</td>
<td>202</td>
</tr>
<tr>
<td>Multiple Systems Coupling (MSC)</td>
<td>4, 132, 161, 170, 256, 284–285, 436</td>
</tr>
<tr>
<td>generic link</td>
<td>185</td>
</tr>
<tr>
<td>generic support</td>
<td>183</td>
</tr>
<tr>
<td>link</td>
<td>170, 178</td>
</tr>
<tr>
<td>local IMS</td>
<td>183, 187</td>
</tr>
<tr>
<td>TCP/IP</td>
<td>170</td>
</tr>
<tr>
<td>multisystem</td>
<td>110, 112</td>
</tr>
<tr>
<td>MVS</td>
<td>37</td>
</tr>
<tr>
<td><strong>N</strong></td>
<td></td>
</tr>
<tr>
<td>NAK response</td>
<td>124, 329</td>
</tr>
<tr>
<td>NAME parameter</td>
<td>170</td>
</tr>
<tr>
<td>network connectivity</td>
<td>170</td>
</tr>
<tr>
<td>new or changed resource definitions</td>
<td>252</td>
</tr>
<tr>
<td>NOFULLSEG</td>
<td>77</td>
</tr>
<tr>
<td>non-shared queues</td>
<td>124</td>
</tr>
<tr>
<td>numeric character</td>
<td>60</td>
</tr>
<tr>
<td><strong>O</strong></td>
<td></td>
</tr>
<tr>
<td>ODBA (Open Database Access)</td>
<td>8, 314</td>
</tr>
<tr>
<td>ODBM (Open Database Manager)</td>
<td>195–197, 314</td>
</tr>
<tr>
<td>Offline Dump Formatter utility</td>
<td>64, 283</td>
</tr>
<tr>
<td>module</td>
<td>64</td>
</tr>
<tr>
<td>OLC Phase</td>
<td>24</td>
</tr>
<tr>
<td>OLCSTAT data</td>
<td>26</td>
</tr>
<tr>
<td>OLDS</td>
<td></td>
</tr>
<tr>
<td>individual</td>
<td>43</td>
</tr>
<tr>
<td>OLDS (online log data set)</td>
<td>3, 30–31, 38, 41, 43, 115, 285</td>
</tr>
<tr>
<td>OM (Operations Manager)</td>
<td>5, 12, 17, 24, 71, 152, 204, 208–210, 284, 329, 436</td>
</tr>
<tr>
<td>Open Database</td>
<td>2, 107, 314</td>
</tr>
<tr>
<td>Open Database Access (ODBA)</td>
<td>8, 314</td>
</tr>
<tr>
<td>Open Database Manager</td>
<td>314</td>
</tr>
<tr>
<td>Open Database Manager (ODBM)</td>
<td>195–197, 314</td>
</tr>
<tr>
<td>Open Transaction Manager Access (OTMA)</td>
<td>4, 31, 33, 110, 126–127, 156, 162, 189, 311, 314, 319</td>
</tr>
<tr>
<td>client</td>
<td>124–125, 314</td>
</tr>
<tr>
<td>help</td>
<td>129</td>
</tr>
<tr>
<td>instance</td>
<td>129</td>
</tr>
<tr>
<td>only limited OTMA transaction monitoring</td>
<td>314</td>
</tr>
<tr>
<td>CM1</td>
<td></td>
</tr>
<tr>
<td>SLO0 transaction monitoring message</td>
<td>116, 121</td>
</tr>
<tr>
<td>transaction</td>
<td>132</td>
</tr>
<tr>
<td>customer</td>
<td>110</td>
</tr>
<tr>
<td>descriptor</td>
<td>162–163, 314–315</td>
</tr>
<tr>
<td>Destination Resolution</td>
<td></td>
</tr>
<tr>
<td>exit routine member client application</td>
<td>127</td>
</tr>
<tr>
<td>Destination Resolution exit routine</td>
<td>317</td>
</tr>
<tr>
<td>member client instance</td>
<td>126</td>
</tr>
<tr>
<td>support for asynchronous IMS to IMS communications</td>
<td>162</td>
</tr>
<tr>
<td>transaction</td>
<td></td>
</tr>
<tr>
<td>expiration</td>
<td>135</td>
</tr>
<tr>
<td>instance block</td>
<td>38</td>
</tr>
<tr>
<td>monitoring</td>
<td>314</td>
</tr>
<tr>
<td>monitoring IMS</td>
<td>12</td>
</tr>
<tr>
<td>processing</td>
<td>132</td>
</tr>
<tr>
<td>transaction pipe</td>
<td>125</td>
</tr>
<tr>
<td>operations</td>
<td>12, 111, 138, 204, 230, 313, 321, 436</td>
</tr>
<tr>
<td>Operations Manager (OM)</td>
<td>5, 12, 17, 24, 71, 152, 204, 208–210, 284, 329, 436</td>
</tr>
<tr>
<td>OSAM</td>
<td>3, 43, 53, 69</td>
</tr>
<tr>
<td>buffer pool change</td>
<td>70</td>
</tr>
<tr>
<td>OSAM (Overflow Sequential Access Method)</td>
<td>43, 69</td>
</tr>
</tbody>
</table>
OSAM buffer pool 69, 71
OSAM buffer pool specification 70
OTMA (Open Transaction Manager Access) 4, 31, 33, 110, 126–127, 156, 162, 189, 311, 314, 319
client 124–125, 314
help 129
instance 129
only limited OTMA transaction monitoring 314
CM1
SL0 transaction monitoring message 116, 121
transaction 132
customer 110
descriptor 162–163, 314–315
Destination Resolution
exit routine member client application 127
Destination Resolution exit routine 317
member client instance 126
support for asynchronous IMS to IMS communications 162
transaction expiration 135
instance block 38
monitoring 314
monitoring IMS 12 280
processing 132
transaction pipe 125
output message 5, 124, 152, 379
Overflow Sequential Access Method (OSAM) 43, 69

 parameter value 83, 208, 211, 327
    complete description 208
partition specification table (PST) 36–37
partitioned data set (PDS) 138
member 300–301
PCB (program communication block) 5, 36, 82, 90, 133, 162–163, 324
PDS (partitioned data set) 138
member 300–301
performance 2, 44–45, 67, 78, 109–110, 139, 238, 286, 310
physical path 183
PK61583 312
PK74017 134
PK86426 134
PLEX 183
PM05984 134
PM05985 134
PM20292 132
PM20293 132
PM21167 389
PM21939 391
PM21942 391, 394
PM21945 394
PM21961 391, 394, 423
PM22116 394
PM22118 394
PM22119 394

PM22120 394
PM22121 394
PM24860 424
PM25552 394
PM31420 99
PM31908 391
PM32394 424
PM34347 391, 394
PM36509 391
PM37894 95, 391
PM41761 235
PM46494 391, 394, 423
point-to-point (PTP) domain 381
port 132, 190, 319
PPT 207
Preventive Service Planning (PSP) 280
    bucket 280
    required fixes 281
    upgrade name 308
previous version 26, 76, 282–283
PRILOG 316
PROCLIB 20, 26, 68, 73, 97, 115, 119, 162, 205, 207, 291, 436
    data 70, 291, 302
    control statement members 308
    DFSDFxxx proclib member 70
    validating user-created members 310
ProductPac 309
program communication block (PCB) 5, 36, 82, 90, 133, 162–163, 324
program directory
    IMS 12 308, 318
program specification block (PSB) 25, 68, 285, 324
    pool 58, 68
    work are pool 60
    work area pool 59
    work pool 58, 68
PROGxx 309
project 110, 431
PSB (program specification block) 25, 68, 285, 324
    pool 58, 68
    work area pool 59–60
    work pool 58, 68
PSBGEN 94, 96
PSP (Preventive Service Planning) 280
    bucket 280
    required fixes 281
    upgrade name 308
PST (partition specification table) 36–37
PTF 281, 284
PTP (point-to-point) domain 381
Pub/Priv key pair 359–360
PURG 162

QoS (Quality of Service) 364
QRY IMSCON Type 167
Quality of Service (QOS) 364
QUERY 14, 21, 71, 163, 203–204
QUERY DBDESC 15
QUERY IMSCON Type  167
QUERY IMSCON TYPE(RMTIMSCON) NAME(*)
SHOW(ALL | showparm)  167
QUERY IMSCON TYPE(RMTIMSCON) NAME(rmtinm)
SHOW(ALL | showparm)  167
QUERY IMSCON TYPE(SENDCLNT) NAME(*)
SHOW(ALL)  168
QUERY MEMBER  28, 178
QUERY OTMADESC  163
QUERY RMTIMSCON NAME(*)  166
QUERY RMTIMSCON NAME(rmtinm)  166
QUERY TRAN  21
DESC  263
QUERY TRAN command  255
Queue Control Facility for z/OS  312

R
return code in IBM 12  330
RBA (relative byte address)  84
RBP (request block prefix)  37
RC (return code)  133, 246
RDD (resource definition data)  15, 21, 202
RDDS  200
RDDS (resource definition data set)  15, 20, 200, 205
  DRD
    environment  251, 253
    export  254
    import  240
RDS (repository data set)  201–202, 244
existing RDS  244
types  202–203
RDS (restart data set)  54, 72
RECON  7, 9, 43, 73, 75–76, 114, 137–138, 148, 283, 310, 320, 436
data  75, 108, 138, 283, 310
  backup copies  139
  CA group  146
  record keys  139
  records  316
  secondary extents  139
data set  7, 108, 138, 283, 311, 313, 316
  record type  139, 316
RECON data
subsystem records  312
recorder trace  161
GDG base  189
RECOVPD value  140
Redbooks website  461
refresh runtime resource definitions  252
relative byte address (RBA)  84
release  303, 305, 308
  appropriate service  308
  planning for IMS 12  308, 313
Syntax Checker  306
remote IMS  162
  connection  164
  datastore ID  164
Remote Site Recovery (RSR)  138, 320
remote system  162
  transaction definition  164
reorganization number  75–76
repository  200, 255–256, 302
  automatic import  214
  data  203
  migration to  205
repository data set (RDS)  201–202, 244
existing RDS  244
types  202–203
repository name  202, 205
Repository Server (RS)  51, 200–201, 244
  catalog repository  201
  user repository  203
request block  37
request block prefix (RBP)  37
request status message (RSM)  191, 330
RESLIB  66
resource  2–3, 12, 17, 31, 110, 139, 199–200, 280–281, 311
  access  12, 16, 201, 212
  adapter  311, 314
  creation  22
  definition  15, 17, 21, 200, 203, 235, 252
in DRD-enabled systems  15
list  237, 253
name  4, 37, 202, 204, 231, 234, 240
online  25
profile  240, 260
structure  26, 49, 204, 231
type  20, 36, 261
return code in IMS 12  330
resource definition data (RDD)  15, 21, 202
resource definition data set (RDDS)  15, 20, 200, 205
  DRD
    environment  251, 253
    export  254
    import  240
Resource Manager (RM)  26, 200–201, 204, 209, 284, 436
  address space  211
  IMS Repository Server  227
  service  214
  utility  201, 204
Resource Recovery Services (RRS)  5, 110–111
dependency  110
restart  15, 108, 162, 283
descriptor definitions  17
IMS  171
restart data set (RDS)  54, 72
return code (RC)  133, 246
REXX SPOC API  14
RM (Resource Manager)  26, 200–201, 204, 209, 284, 436
  address space  211
  IMS Repository Server  227
  service  214
  utility  201, 204
incoming requests 201
RMTIMSCON connection 165
root segment 78, 80
RRS (Resource Recovery Services) 5, 110–111
dependency 110
RS (Repository Server) 51, 200–201, 244
catalog repository 201
user repository 203
RSM (request status message) 191, 330
RSR (Remote Site Recovery) 17, 30, 138, 320
runtime resource definition 220

S
same IMS 126–127, 182
SAML (Security Assertion Markup Language) 348
  2.0
    unsigned assertion 326
    unsigned sender-vouches token 356
    unsigned token 348
    assertion 353
    profile 353
SAP (save area prefix) 37
SAS (secondary address space) 61
save area prefix (SAP) 37
scheduling 58, 108, 239
SCHEDxx 309
SCI (Structured Call Interface) 12, 194, 204, 209, 284, 436
SDFSRESL 283, 302
search field 83, 86, 202
secondary address space (SAS) 61
secondary index 75, 79, 205, 333
  key 79
  new DEDB 96
  SEGM statement 83
  sequence field 87
  XDFLD statement 82
Secure Sockets Layer (SSL) 327
security 5, 126, 169, 188, 212, 316, 319
Security Assertion Markup Language (SAML) 348
  2.0
    unsigned assertion 326
    unsigned sender-vouches token 356
    unsigned token 348
    assertion 353
    profile 353
Security Services Technical Council (SSTC) 353
SEGM statement 82
segment search argument (SSA) 83
send-then-commit 125, 130, 156
SENSEG statement 81, 90
server property 348
ServerPac 309
service level agreement (SLA) 30
SET AOSLOG command 38, 123
SETRACF 198
Shared Message Queue (SMQ) 111
shared queue 5, 11, 20, 47, 109–110, 183–184, 436
single point 12, 161, 204
single point of control (SPOC) 12, 71, 165, 167, 179, 204
skeletal JCL 138
  execution member 143
  member CAJCL 151
SLA (service level agreement) 30
SLDS (system log data set) 3, 41, 43, 152
small programming enhancement (SPE) 74, 106, 134, 157, 283
SMEM=YES 168
SMP/E 281, 308–309
SNAP DB 33
SOAP 347
SOAP Gateway 324, 358, 449
  AT-TLS configuration 365
  AT-TLS feature 360
  data exchanges 364
  maximum connections 365
  security references 376
SOAPheader 350
sockets 168, 175, 178, 319
Solution Packs 386
source segment 78, 84
  concatenated key 84
spare RDS 244
SPARE status 226, 248
SPE (small programming enhancement) 74, 106, 134, 157, 283
specified CA group, recovery period 148
specified RDDS, stored resource definitions 251
SPOC (single point of control) 12, 71, 165, 167, 179, 204
SQL access 337
SSA (segment search argument) 83
SSL (Secure Sockets Layer) 327
SSLPORT 445
SSTC (Security Services Technical Council) 353
standard SQL 333
STARTRMT rmtimscon 166
startup procedure 207, 319
stateless session bean 382
status code 92
STOPRMT rmtimscon 166
storage 2, 11, 30–31, 68, 126, 152, 188, 190, 230, 310, 313
stored resource definitions 214
STRUCTURE Name 49
Structured Call Interface (SCI) 12, 194, 204, 209, 284, 436
subsystem 48, 138, 284, 286
  databases access 138
super member 168
SXPL 284
symbolic keyword 143
synchronization level
  None 110
  SYNCPT 110
synchronous call-out 382
syncpoint 5, 110, 116
Syntax Checker 285, 303
  previous release 321
syntax view 263
  equivalent panel 273
Virtual Telecommunications Access Method (VTAM) 4, 170, 436
Volume 3 124, 141, 313
VSAM (Virtual Storage Access Method) 3, 44, 52, 69, 138, 202
VTAM (Virtual Telecommunications Access Method) 4, 170, 436

W
WADS (write-ahead data set) 41
wait for input (WFI) 110
web service 332, 374
  artifact 324
  connection bundle entry name 377
  consumer 348
  provider scenario 326
  request 348
  server 348
Web Services Description Language (WSDL) 348
WebSphere 2, 109, 126
WebSphere Application Server for z/OS 314
WFI (wait for input) 110
workload balancing 48
write to operator with reply (WTOR) 5, 13, 31, 165, 167, 175
write-ahead data set (WADS) 41
WSDL (Web Services Description Language) 348
WTOR (write to operator with reply) 5, 13, 31, 165, 167, 175

X
XCF (cross-system coupling facility) 4, 110, 156, 188, 201, 285, 319
  communication 119
  group 208
    name 208, 215
    name parameter value 216
  indicator 111
  input transaction message 119
XDFLD statement 82, 84, 86
XFACILIT class 260
  catalog repository 261
XIBDS (Exit Interface Block Data Store) 319
XML 71, 161, 192, 298–299
  converter 192, 379
  format 330
    DBD metacode 338
    return output 330
XRF (extended recovery facility) 17, 30, 138

Z
z/OS 2, 13, 49, 51, 126, 135, 141, 167–168, 201, 204, 227, 282, 291, 432
  1.12 106
  Enterprise Suite V2.1 324
IBM IMS Version 12 Technical Overview
IBM IMS Version 12
Technical Overview

Explore the new features and functions of IMS 12

IBM Information Management System (IMS) provides leadership in performance, reliability, and security to help you implement the most strategic and critical enterprise applications. IMS also keeps pace with the IT industry. IMS, Enterprise Suite 2.1, and IMS Tools continue to evolve to provide value and meet the needs of enterprise customers.

With IMS 12, integration and open access improvements provide flexibility and support business growth requirements. Manageability enhancements help optimize system staff productivity by improving ease of use and autonomic computing facilities and by providing increased availability. Scalability improvements have been made to the well-known performance, efficiency, availability, and resilience of IMS by using 64-bit storage.

IBM IMS Enterprise Suite for z/OS V2.1 components enhance the use of IMS applications and data. In this release, components (either orderable or downloaded from the web) deliver innovative new capabilities for your IMS environment. They enhance connectivity, expand application development, extend standards and tools for a service-oriented architecture (SOA), ease installation, and provide simplified interfaces.

This IBM Redbooks publication explores the new features of IMS 12 and Enterprise Suite 2.1 and provides an overview of the IMS tools. In addition, this book highlights the major new functions and facilitates database administrators in their planning for installation and migration.

SG24-7972-00

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