

Migration to IBM CICS Transaction Server for z/VSE

Klaus Wacker

Ingolf Salm





International Technical Support Organization

Migration to IBM CICS Transaction Server for z/VSE

May 2017

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

First Edition (May 2017)

This edition applies to Version 2, Release 1, CICS Transaction Server for z/VSE (5655-VSE).

© Copyright International Business Machines Corporation 2017. All rights reserved.

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Notices	vii
Trademarks	viii
Preface	ix
Authors	ix
Now you can become a published author, too!	x
Comments welcome	x
Stay connected to IBM Redbooks	x
Chapter 1. Introduction	1
1.1 z/VSE 6.1 overview	2
1.1.1 CICS TS for z/VSE 2.1	2
1.1.2 Other items notable in z/VSE 6.1	10
1.1.3 Migration issues	11
1.1.4 Migration planning summary	11
1.2 z/VSE Version 6: What will be next	12
Chapter 2. Installation and tailoring	15
2.1 Planning	16
2.1.1 Planning steps for installation	16
2.1.2 Hardware considerations	16
2.1.3 Software considerations	17
2.1.4 Other considerations	18
2.2 Installing z/VSE 6.1 and CICS Transaction Server	18
2.3 Tailoring the CICS Transaction Server	19
2.3.1 Reviewing virtual storage requirements for CICS TS	19
2.3.2 Using a CICS system definition data set file	20
2.3.3 Defining system files for a second CICS TS	21
2.3.4 Customizing the DFHSIT table	30
2.3.5 Customizing the DFHPLT tables	35
2.3.6 Tailoring the CICS start-up job stream	38
2.3.7 Autoinstall of resources	40
2.4 Migration from CICS/VSE 2.3	47
2.4.1 Migrating the DFHPCT and DFHPPT tables	47
2.4.2 Customizing and migrating the DFHFCT table	52
2.4.3 Migrating the DFHTCT table	57
2.4.4 Other considerations	62
Chapter 3. Security	65
3.1 z/VSE Version 6 Release 1 security	66
3.1.1 z/VSE V6R1 security options	66
3.1.2 Basic security manager	67
3.1.3 Security Server	69
3.1.4 Batch security with DTSECTAB	75
3.1.5 BSM logging and reporting	75
3.1.6 System Authorization Facility	76
3.1.7 Maintaining BSM security profiles	77
3.1.8 Migrating BSM profiles	81
3.1.9 BSM cross-reference reports	82

3.1.10 External security manager installation	83
3.2 CICS Transaction Server for z/VSE 2.1 security.	83
3.2.1 CICS TS Sign-on security provided by the BSM.	83
3.2.2 User sign-on and sign-off	94
3.2.3 User password checking.	96
3.2.4 CICS TS security parameters	96
3.2.5 CICS TS transaction-attach security.	96
3.2.6 CICS default user ID	100
3.2.7 Security for program list table programs at startup	102
3.2.8 Resource security checking	102
3.2.9 Command security checking not supported by BSM	102
3.2.10 Surrogate user security.	103
3.2.11 Security on intercommunication	103
3.2.12 CICS Report Controller.	104
3.2.13 Printer security	105
3.2.14 Terminal security.	105
3.3 Security Migration Aid	105
3.4 Summary.	106
Chapter 4. CICS Explorer	107
4.1 Overview of CICS Explorer Workbench	109
4.1.1 Selecting CICS Explorer views	111
4.2 Supported Operations views.	111
4.3 Supported CICS Management Interface resources	113
4.3.1 Using an HTTP GET request for information about resources	114
4.3.2 Using an HTTP PUT request to process a single resource	115
4.4 Restrictions when connecting to a CICS TS for z/VSE system.	117
4.5 Installing and configuring the CICS Explorer	117
4.5.1 Configuring the z/VSE host for use with CICS Explorer	117
4.5.2 Starting and stopping the CICS Explorer server-part	120
4.5.3 Obtaining a copy of the CICS Explorer Client.	120
4.5.4 Installing CICS Explorer Client	121
4.6 Configuring a new connection to a CICS system	121
4.6.1 Adding a set of CICS Explorer credentials	121
4.6.2 Adding a CICS Explorer connection	123
4.7 Using a connection to a CICS system.	125
4.7.1 Changing a CICS Explorer user workspace	126
4.8 Using Job EYUPARM to enter or modify debugging commands	126
4.9 Messages generated when using the CICS Explorer	128
Chapter 5. Web support	129
5.1 Connection security.	130
5.1.1 Transport-layer security versus message-layer security.	130
5.2 Using SOAP for inter-program communication.	130
5.2.1 z/VSE support for web services and SOAP overview.	131
5.2.2 SOAP syntax overview	131
5.2.3 Web Service Security overview.	132
5.2.4 Using authentication with web Service Security	132
5.2.5 How the z/VSE host can act as the SOAP server.	134
5.2.6 Using web Service Security features when z/VSE acts as the SOAP server	136
5.2.7 How the z/VSE host can act as the SOAP client	137
5.2.8 Using web Service Security features when z/VSE acts as the SOAP client.	138
5.3 3270 bridge	138

5.3.1 DFHWBTTA	140
5.3.2 Programs with BMS support	140
5.4 ECI/CICS Transaction Gateway	156
5.4.1 How the ECI and CICS Transaction Gateway are used	157
5.4.2 How the CICS Transaction Gateway accesses CICS	158
5.4.3 External Call Interface	158
5.5 CICS Listener	159
5.6 Evaluation	162
Chapter 6. Channels and containers	163
6.1 COMMAREA overview	164
6.2 Passing data requirements	165
6.3 Channels and containers approach	165
6.3.1 General concepts	166
6.3.2 Channels	167
6.3.3 Containers	172
6.3.4 CICS read-only containers	173
6.3.5 Data conversion	173
6.3.6 Benefits of using channels and containers	174
6.3.7 Porting COMMAREA to channels and containers	175
6.3.8 Converting COMMAREA to channels example	177
Chapter 7. CICS customization	183
7.1 System generation	184
7.2 Initialization and termination processing	184
7.2.1 System initialization overlays (CICS/VSE)	184
7.2.2 Program list table programs	184
7.3 User exits	186
7.4 Global user exits	186
7.4.1 Task-related user exits	187
7.5 User-replaceable modules	187
7.5.1 z/VSE-supplied URMs	188
7.6 System programmer interfaces	189
7.6.1 System programming macros	189
7.6.2 Programmable interface to CEMT	189
7.6.3 System Programming Interface	191
Chapter 8. Performance and tuning	193
8.1 Virtual storage considerations	194
8.1.1 VSE startup parameters	194
8.1.2 Shared Virtual Area	195
8.1.3 CICS partition layout	196
8.1.4 Storage requirements for maximum task specification	201
8.2 System resource requirements	202
8.2.1 Real storage considerations	202
8.3 Statistics and monitoring	203
8.3.1 Setting up the DMF	203
8.3.2 Statistics	210
8.3.3 Monitoring	215
Chapter 9. CICS application program considerations	217
9.1 Compatibility	218
9.1.1 Changes to API commands in CICS TS	218
9.1.2 Changes to rounding for ASKTIME and FORMATTIME commands	219

9.1.3 Changes to INQUIRE SYSTEM command	220
9.2 Migrating macro-level applications	220
9.2.1 DFHMSCAN	220
9.2.2 CICS Application Migration Aid	222
9.3 z/VSE compile dialogs	223
9.4 CICS Basic Mapping Support (BMS)	224
Chapter 10. CICS problem determination	227
10.1 CICS Transaction Server for z/VSE 2.1 environment overview	228
10.2 CICS tracing	228
10.2.1 Trace levels	229
10.2.2 Control options	229
10.2.3 Default SIT options	230
10.2.4 CETR overview	231
10.2.5 Using CETR	233
10.2.6 Tracing scenarios	235
10.2.7 Trace formatting	236
10.3 CICS dumps	245
10.3.1 Preparing for stand-alone dump	246
10.3.2 Processing the CICS dump from a stand-alone disk	246
10.3.3 Program check and abend information	248
10.3.4 Default SIT options	249
10.4 DUMP TABLE facility	250
10.4.1 Overview	250
10.4.2 Transaction Dump table	250
10.4.3 System Dump table	251
10.4.4 Dump suppression for ASRA and ASRB abends	251
10.5 CSFE and CEDF	252
10.5.1 Changes to the CSFE DEBUG transaction	252
10.5.2 CEDF support for remote transactions	252
Appendix A. IBM-supplied CSD groups	253
Appendix B. CICS TS statistics output examples	257
Sample Statistics Program output	258
Abbreviations and acronyms	265
Related publications	269
IBM Redbooks	269
Other publications	269
Online resources	270
Help from IBM	270

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

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 grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing, IBM Corporation, North Castle Drive, MD-NC119, Armonk, NY 10504-1785, US

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 jurisdictions 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 provide in any way it believes appropriate without incurring any obligation to you.

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

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.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

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 actual people or business enterprises 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. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at “Copyright and trademark information” at <http://www.ibm.com/legal/copytrade.shtml>

The following terms are trademarks or registered trademarks of International Business Machines Corporation, and might also be trademarks or registered trademarks in other countries.

C/370™	OS/390®	WebSphere®
CICS®	POWER®	z Systems®
CICS Explorer®	RACF®	z/Architecture®
CICSplex®	Redbooks®	z/OS®
DB2®	Redbooks (logo)  ®	z/VM®
IBM®	RS/6000®	z/VSE®
IBM z Systems®	System z®	z10™
IBM z13s™	System z10®	z13®
Language Environment®	VTAM®	z13s™

The following terms are trademarks of other companies:

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

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.

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

Preface

The IBM® CICS® Transaction Server for z/VSE® (CICS TS for z/VSE) 2.1 provides functions to improve application programming, system programming, system management, and data security and availability. With CICS TS for z/VSE 2.1, you can use the extended functionality of Basic Security Manager. CICS TS for z/VSE 2.1 can be administrated by the IBM CICS Explorer® function on a workstation, which allows CICS management in a convenient way.

This IBM Redbooks® publication provides information to help you install, tailor, and configure the CICS TS for z/VSE 2.1 product. The book is intended for IBM z/VSE customers and IBM technical personnel who are responsible for planning and migrating to IBM z/VSE 6.1 and CICS TS for z/VSE 2.1.

The book also provides information to help you understand the affect of migrating to CICS TS for z/VSE 2.1. It provides detailed guidance and samples for installing and configuring CICS TS for z/VSE 2.1.

Also included in the book is a description of the CICS TS for z/VSE 2.1 features and capabilities and the affect of removing obsolete functions. The book also covers security and performance issues and provides samples for first level problem determination through the use of memory dumps or the use of trace tools.

Authors

This book was produced by a team of specialists from around the world working with the International Technical Support Organization, Poughkeepsie Center.

Klaus Wacker is a Senior IT Specialist in IBM Systems group that is based at Boeblingen Laboratory, Germany. He has over 30 years of experience in operating system development for IBM z® Systems. He holds a degree in Computer Science from Technical University in Stuttgart, Germany. His areas of expertise include VSE system integration and security and CICS application and system programming. He is the author of several articles about z/VSE and Linux on IBM z Systems®.

Ingolf Salm is a Senior Technical Staff Member who is responsible for the z/VSE system design and release content. He has many years of experience with z/VSE and the mainframe. Ingolf published many articles and was co-author of several IBM Redbooks publications and a Linux book.

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

LindaMay Patterson
Bill White
International Technical Support Organization

Heinz Hagedorn
Ingo Franzki
Jens Remus
IBM Laboratory Boeblingen

Now you can become a published author, too!

Here's 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:

<https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm>

- Stay current on recent Redbooks publications with RSS Feeds:

<http://www.redbooks.ibm.com/rss.html>



Introduction

IBM z/VSE 6.1 introduces a version of the IBM CICS Transaction Server for z/VSE (CICS TS for z/VSE 2.1). This version is the first major enhancement since 2000.

CICS TS for z/VSE 2.1 is based on IBM CICS TS for VSE/ESA 1.1.1 and replaces CICS TS for VSE/ESA 1.1.1 in z/VSE 6.1. There is no need to recompile or relink CICS applications if they do not use CICS TS internal control blocks or CICS interfaces.

This chapter describes key functions of CICS TS for z/VSE 2.1 and other enhancements and changes that are included in this release, and includes the following topics:

- ▶ 1.1, “z/VSE 6.1 overview” on page 2
- ▶ 1.2, “z/VSE Version 6: What will be next” on page 12

1.1 z/VSE 6.1 overview

CICS TS for z/VSE 2.1 provides functions that improve application and system programming, system management, and the reliability and integrity of your CICS system. CICS TS for z/VSE 2.1 includes the following key highlights:

- ▶ Supports channels and containers through the EXEC CICS API for use within CICS programs. The channels and containers API was first introduced with the IBM CICS Transaction Server for z/OS® 3.1 and extended several times. CICS TS for z/VSE supports a subset of the functionality, which is available on CICS Transaction Server for z/OS.
- ▶ Provides extensions for the support of the IBM CICS Explorer a workstation-based management tool for easy system administration and operating.
- ▶ Provides various options to integrate the CICS applications into the internet. Among the options are the CICS Listener, support for SOAP, and the 3270 Bridge to integrate 3270 applications without the necessity of code changes to the application. CICS TS also provides API commands to implement web-aware applications.
- ▶ Includes cipher suites AES 128/AES 256 for CICS web support. CICS web support is enhanced to allow access with AES 128 and AES 256 cipher suites for improved security.
- ▶ Includes extensions for CICS API to inquire the OS level and modified time handling.
- ▶ Uses the extended functionality of the Basic Security Manager (BSM), which was introduced after its original introduction in VSE/ESA 2.4. For most customers, the security functions of BSM are sufficient.
- ▶ Makes RPG II online programs usable with CICS TS.
- ▶ Supports IBM Language Environment®-conforming main assembler programs and single source COBOL/VSE programs for batch and online.
- ▶ Support IBM MQ Trigger Monitor to supplement the IBM MQ Client for z/VSE.
- ▶ Provides up-to-date online information by using the IBM Knowledge Center CICS Transaction Server for z/VSE V2.1.
- ▶ TCP/IP functionality is supplied by two vendors. Both vendors introduce enhanced functionality in z/VSE 6.1, including support for firewall and improved Transport Layer Security/Secure Sockets Layer (TLS/SSL) cryptography.
- ▶ Use of an installation disk (since z/VSE 5.2) for tape-less operation is also provided.

1.1.1 CICS TS for z/VSE 2.1

When migrating to CICS TS, be aware of the items that are described in this section.

Decisions to make: Functions that are introduced require decisions as to when and if these functions will be implemented within a production environment.

CICS TS code

CICS TS for z/VSE 2.1 is a release of CICS for the z/VSE platform that is based on CICS TS for VSE/ESA 1.1.1 with support for channels and containers and data conversion enhancements that are ported from CICS TS for z/OS 3.1.

Domain architecture

Selected areas of CICS TS code are structured into domains, which improves code quality and serviceability and extends 31-bit support for almost all CICS TS code. This ability allows for better use of Extended Architecture Support.

Figure 1-1 shows an overview on the CICS TS Domain architecture, which is described next.

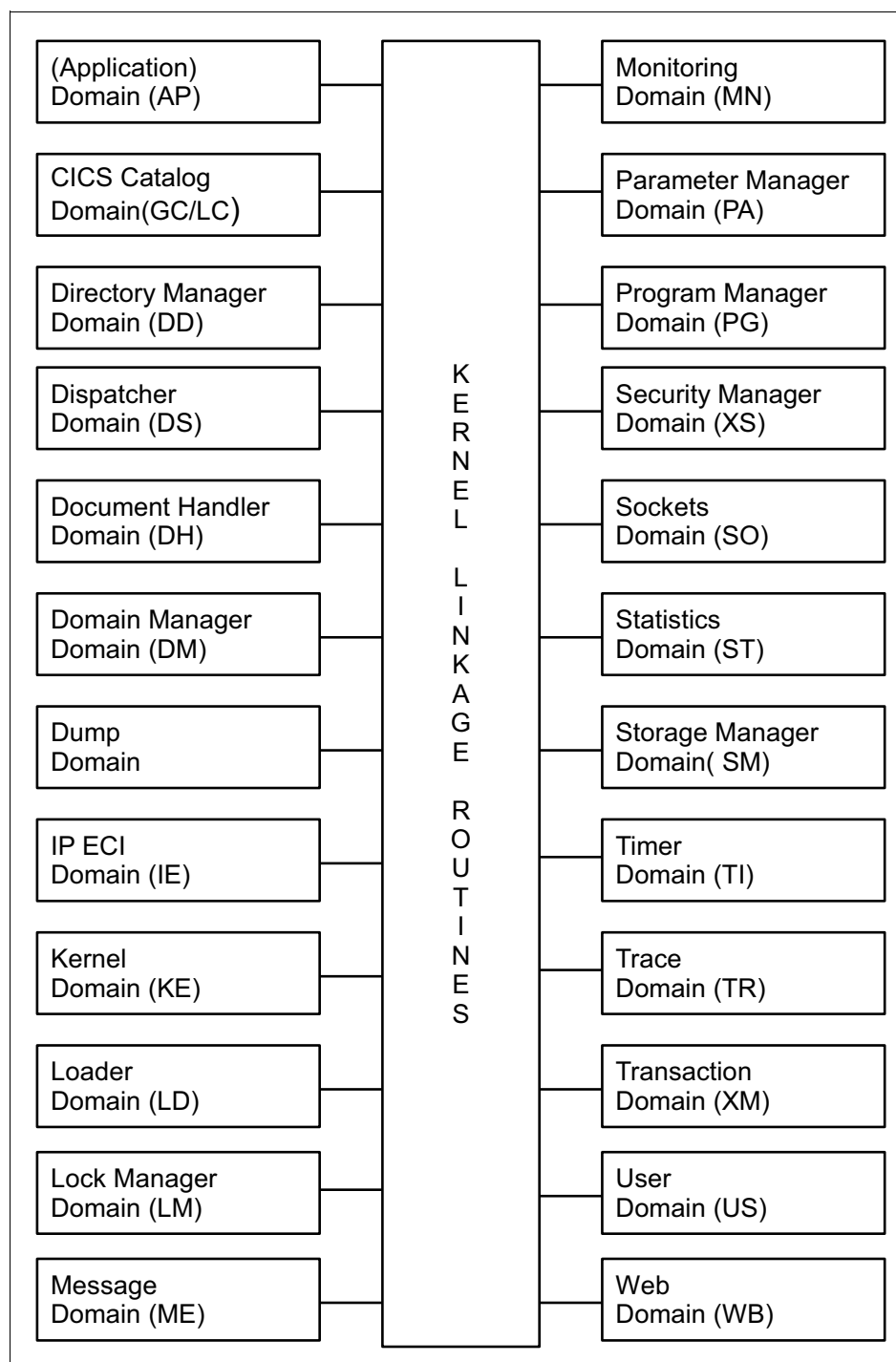


Figure 1-1 CICS TS Domain structure

The following domains are shown in Figure 1-1 on page 3:

► Application domain (AP)

Application programs run in the application domain. The CICS management programs within the AP domain are not restructured. However, the AP domain and other domains communicate through the domain interface.

► Catalog domains

The following catalog domains are available:

- Local catalog domain (LC)
- Global catalog domain (GC)

The catalog domains are used by the other domains to hold information that relates to an orderly restart. They allow CICS to read, write, and purge records on the local and global catalog data sets so that a record of the status of CICS can be maintained when CICS is not running. The catalog domains use a common set of programs to provide a domain interface to VSAM KSDS data sets, and they conceal the underlying VSAM operations from the calling domains.

The local catalog holds information relevant to a particular CICS system, including a list of domains.

The global catalog holds information that is applicable to the entire CICS system. Therefore, in an extended recovery facility (XRF) system that consists of one active and one alternative CICS system, there are two local catalogs and one global catalog. Conversely, in a non-XRF system, there is one local catalog and one global catalog.

► Directory manager domain (DD)

The directory manager domain is a service component that provides the following resource table lookup services for the other domains (except the application domain, which uses the table manager program):

- Transaction definitions
- Remote transaction definitions
- Transaction classes
- Transaction profile (TP) names
- User attributes
- Programs
- Basic mapping support (BMS) map sets and partition sets

The table manager program handles the following items:

- Connections
- Terminals
- Sessions
- Modegroups
- Files
- Profiles
- Autoinstall terminal models

► Dispatcher domain (DS)

The dispatcher domain controls attaching, running, and detaching tasks and controls the scheduling of VSE tasks.

► Document handler domain (DH)

The document handler domain manages CICS documents.

- ▶ **Domain manager domain (DM)**
The domain manager domain maintains (by using catalog services) permanent information about the status of individual domains. The domain manager domain also attaches initialization and termination tasks for the other domains.
- ▶ **Dump domain (DU)**
The dump domain produces storage dumps and handles the associated data sets (transaction and system dump data sets) and status in the CICS system.
- ▶ **Internet Protocol external call interface (IP ECI) domain (IE)**
The IP ECI domain provides services that are used by the CICS External Presentation Interface (EPI) protocol over IP connections.
- ▶ **Kernel domain (KE)**
The kernel domain is the main CICS control structure. The kernel tracks the existence of domains and is involved in every call from one domain to another, which provides a consistent linkage and recovery environment for CICS.

Serious system errors can result in the termination of the kernel domain by using a system dump that contains diagnostic and problem determination information.
- ▶ **Loader domain (LD)**
The loader domain is used by the other domains to gain access to storage-resident copies of nucleus and application programs, maps, and tables.
- ▶ **Lock manager domain (LM)**
The lock manager domain provides locking and associated queuing facilities for CICS resources. Each resource is associated with a unique lock name that is used to access locking facilities.
- ▶ **Message domain (ME)**
The message domain acts as a centralized repository for storing and issuing CICS messages for all parts of restructured CICS. This approach ensures consistency of messages that are issued by the central controlling structures of CICS.
- ▶ **Monitoring domain (MN)**
The monitoring domain controls all monitoring functions within CICS.

Monitoring data is written to Data Management Facility (DMF) data sets and can be used for subsequent processing by the DFHMNDUP monitoring utility program.
- ▶ **Parameter manager domain (PA)**
The parameter manager domain controls the process of applying system initialization parameters to CICS. Initialization occurs by using standardized interfaces that provide an improved method of communicating between functional areas of CICS.
- ▶ **Program manager domain (PG)**
The program manager domain supports the following areas of CICS:
 - Program control functions, such as EXEC CICS LINK, XCTL, LOAD, RELEASE, and RETURN.
 - Transaction abend and condition handling functions, such as EXEC CICS ABEND, HANDLE ABEND, HANDLE CONDITION, and HANDLE AID.
 - Related functions, such as invoking user-replaceable programs, global user exits, and task-related user exits.
 - Autoinstall for programs, map sets, and partition sets.

► Security manager domain (XS)

The security function (previously provided by the security identification program DFHACEE) controls the following elements:

- Multiple sign-on by the same user
- Security messages
- Warnings when a password is about to expire
- Idle terminal sign off timeout

It also provides an optional facility for checking user authority to run transactions and access resources.

The security function is split into the following domains:

- User
- Security

The security domain manages the security capabilities of users and handles all the interfaces to the external security manager (ESM).

► Sockets domain (SO)

The sockets domain provides TCP/IP services to CICS. It includes a TCP/IP listener system task, the TCPIP SERVICE resource to manage the listener, and domain gates to operate on a TCP/IP connection.

► Statistics domain (ST)

The statistics domain controls the collection of resource statistics for a CICS system. The statistics domain collects statistics data at a user-specified interval, at a system quiesce, or at a logical end-of-day, and when requested by a user. Statistics data is written to DMF data sets, and can then be used by the statistics offline utility DFHSTUP to produce formatted reports.

► Storage manager domain (SM)

The storage manager domain manages virtual storage requests for the CICS system.

► Timer domain (TI)

The timer domain provides interval timing and alarm clock services for the other domains. The timer domain also provides date and time facilities and conversion functions, including the ability to synchronize the CICS local time with the operating system clock when the system operator adjusted the time zone.

► Trace domain (TR)

The trace domain is used by CICS system code and application programs to record details of the sequence of events occurring in the CICS system. The basic unit of information that is created for this purpose is the trace entry. The trace domain can write entries to the following traces:

- Internal, which is a wraparound table in main storage in a CICS address space
- Auxiliary, which is a pair of CICS controlled SAM data sets that are used alternatively

► Transaction manager domain (XM)

The transaction manager domain provides transaction-related services to perform the following actions:

- Create tables
- End, purge, and inquire about tasks
- Manage transaction definitions and classes

The transaction manager domain also provides a transaction environment that allows other CICS components to implement transaction-related services.

- ▶ User domain (US)

The user domain restructures the function that was provided by the sign-on facility.

- ▶ Web domain (WB)

The web domain manages interaction between CICS and web clients or between CICS as an HTTP client and servers on the internet.

Using IBM z/Architecture subsystem storage protection

CICS TS uses the IBM z/Architecture® subsystem storage protection facility in a way that enables you to prevent CICS code and control blocks from being overwritten accidentally by your user application programs.

It does not prevent the following situations:

- ▶ Deliberate overwriting of CICS code or control blocks. CICS cannot prevent a user application from obtaining the necessary access key to modify CICS storage.
- ▶ Application programs and data being overwritten by another application program. Programs can be protected if they are written to reentrant standards and are link-edited with the SVA attribute. CICS loads these programs into read-only storage when the RENTPGM=PROTECT system initialization parameter is used.

CICS storage protection is optional. However, if you want to use it, you must have the hardware facility that supports the z/Architecture subsystem storage protection facility. For more information about suitable machines, see [CICS Transaction Server for VSE/ESA Release Guide](#), GC33-1645.

Virtual storage constraint relief

CICS TS continues the process of moving as much CICS code and control blocks as possible above the 16 MB line, which leaves more available storage below 16 MB for user applications that require it. Only CICS code and data areas that have a dependency on 24-bit addressing are left below the 16 MB line, which provides considerable virtual storage constraint relief (VSCR).

External CICS interface

The external CICS interface (EXCI) is an API that allows a non-CICS program (a client program) that is running in z/VSE to call a program (a server program) that is running in the CICS TS region. A communication area enables the exchange of information, with the CICS program being started as though it were linked by another CICS program.

Front-end programming interface

The front-end programming interface (FEPI) is used as a front end to application programs, which allows you to extend or use these applications differently without changing the application programs.

FEPI increases the flexibility of your system by providing the following functions:

- ▶ Allows CICS application programs to be used in different ways, combinations, and environments, and on different systems without changing them.
- ▶ Enables a new program to access an existing program by acting as though it were a terminal accessing the existing program.
- ▶ Allows you to write programs in a way that adds new function to old programs.

System management and resource definition online (RDO)

Consider the following points regarding the system management improvements in the CICS TS:

- ▶ Dynamic transaction routing (DTR) program is provided to maintain better information about the state of routed transactions. The dynamic transaction routing mechanism performs the following functions:
 - Enables a DTR program to make more intelligent routing decisions; for example, based on workload goals.
 - Detects inter-transaction affinities as they are created.
- ▶ You can dynamically define programs, mapsets, and partition sets on their first usage.
- ▶ You can define which CICS journals should be automatically archived by CICS when they are closed for output.
- ▶ Improved monitoring and statistics in CICS are provided by writing statistical and monitoring data to Data Management Facility Data Handler data sets, rather than to CICS journals. This data can be used for subsequent processing by the monitoring utility program DFHMNDUP or by the statistical utility program DFHSTUP.
- ▶ You can use RDO to define the following resources that are associated with CICS files:
 - VSAM files
 - Remote VSAM files
 - Remote DAM files
 - VSAM local shared resource (LSR) pools
 - Shared data tables

Shared data tables

Shared data tables (SDT) make the data tables facility that was introduced in CICS/VSE 2.2 obsolete. By using z/Architecture data spaces and cross-memory services, the use of SDT includes the following important advantages:

- ▶ They reduce the cost of file sharing by eliminating the overhead of function shipping for all read and browse requests. The overhead of function shipping requests applies to remote update requests only.
- ▶ They provide support for browse requests between CICS regions in the same VSE image with good performance, full integrity, recovery, and availability.

You can define an SDT that is not shared between CICS systems. Read and browse performance is significantly better than compared to using VSAM LSR or NSR buffering for the file.

Security options

CICS TS does not provide any internal security. You can secure your system by using the following methods:

- ▶ As in the basic form of an external security manager (BSM) that is provided with the z/VSE system package.

The BSM provides sign-on and transaction-attach security and resource security. User IDs and operator data that were defined in the CICS sign-on table (DFHSNT) must be defined to the BSM by using the z/VSE Interactive Interface. Transaction-attach security is provided by the BSM only by using profile definitions. Securing transactions by using DTSECTXN (as in previous z/VSE versions) is no longer supported.

- Using an external security manager (ESM) that conforms to the VSE/ESA System Authorization Facility (SAF) interface.

To provide the necessary security for CICS regions, CICS issues RACROUTE calls by using the System Authorization Facility (SAF) to route authorization requests to an ESM at appropriate points within CICS transaction processing.

Cipher suites AES 128/AES 256 for CICS web support

CICS web support is enhanced to allow access with AES 128 and AES 256 cipher suites for improved security. Specifically, cipher suite TLS_RSA_WITH_AES_128_CBC_SHA for 128-bit encryption (hex code 2F) is introduced in CICS TS for z/VSE V2.1 with cipher suite TLS_RSA_WITH_AES_256_CBC_SHA for 256-bit encryption (hex code 35).

Extensions to CICS API

The CICS API features the following extensions:

- EXEC CICS INQUIRE OSLEVEL

The OSLEVEL option is added to the **INQUIRE SYSTEM** command. This option returns a 6-byte field that shows the version, release, and modification level of the z/VSE system where CICS TS for z/VSE is running.

- EXEC CICS ASKTIME ABSTIME

The ABSTIME value that is returned by the **EXEC CICS ASKTIME** command is no longer rounded to the nearest 1/100 second. The absolute time that is returned is the system time-of-day clock, which is adjusted for the local time zone offset truncated to the millisecond, and returned as a packed decimal of length 8 bytes.

- EXEC CICS FORMATTIME

Before CICS TS for z/VSE V2.1, the **EXEC CICS FORMATTIME** command rounded up a returned time if the number of milliseconds was greater than 500. In CICS TS for z/VSE V2.1, this rounding is no longer performed, and the returned time is shown with the number of completed seconds.

TCP/IP options

CICS uses TCP/IP for internet connectivity. The following options are provided by vendors:

- IBM IPv6/VSE 1.2, which is licensed from Barnard Software Inc. (BSI)
- IBM TCP/IP for z/VSE 2.1, which is licensed from CSI International

z/VSE also supports the Linux Fast Path option (since z/VSE 4.3), which uses the Linux on z Systems networking facilities for TCP/IP communication without the use of a local TCP/IP stack.

Optional program DL/I VSE 1.12

The optional DL/I VSE 1.12 program, which is upwardly compatible with DL/I 1.10, runs under the control of the CICS TS. DL/I 1.12 contributes to virtual storage constraint relief and moves DL/I resources above 16 MB.

Note: DL/I 1.12 is the only DL/I release that can run on z/VSE 5.1, 5.2, and 6.1.

IBM MQ Client for VSE and IBM MQ Trigger Monitor

The IBM MQ Client for z/VSE is seen as a replacement for the IBM WebSphere® for z/VSE 3.0, which was withdrawn from service in 2015. To supplement the IBM MQ Client functionality, an IBM MQ trigger monitor is provided in the z/VSE base functionality. The trigger monitor can start a CICS TS program when a message arrives on a WebSphere MQ server queue.

The IBM MQ trigger monitor function is included with the z/VSE 6.1 base operating system and is available by using PTF for z/VSE V5.

For more information about the IBM MQ Trigger Monitor function, see [Migrating from MQ Server on z/VSE to MQ Client using the z/VSE MQ Client Trigger Monitor](#).

Support for programming languages

CICS TS for z/VSE 2.1 supports only those applications that use the CICS command-level API and are compiled with the following compilers:

- ▶ COBOL for VSE/ESA
- ▶ C for VSE/ESA
- ▶ PL/I for VSE/ESA
- ▶ High-level assembler
- ▶ RPG II

Programs that are compiled with the VS COBOL II and DOS/VS COBOL compilers are supported if they are link-edited against the Language Environment/VSE runtime library. CICS TS for z/VSE 2.1 does not support IBM C/370™ or DOS PL/I compiled programs.

CICS TS for z/VSE 2.1 supports Language Environment conforming main assembler programs, which allows Language Environment z/VSE assembler macros to be called. For more information, see [the IBM z/VSE home page](#).

VS COBOL II, DOS/VS COBOL, C/370, and DOS PL/I are no longer available as VSE/ESA optional products.

1.1.2 Other items notable in z/VSE 6.1

This section describes other items that are notable in z/VSE 6.1.

z/VSE Installation disk

Since z/VSE 5.2, an installation disk can be used for initial installation. This disk reduces dependency from a tape drive and fosters tape-less computer environments.

z/VSE system layout

During initial installation, you must choose one of the predefined environments as the basis for your system. If none of the environments meet your needs, select the most suitable environment and tailor it according to your requirements. For more information about predefined environments, see Table 2-2 on page 18.

Label area on virtual disk

VSE/ESA 2.4 automatically loads the label area onto virtual disk File Description File (FDF), which is a directly accessed data space.

Access control for VSE/POWER spool entries

IBM VSE/POWER® security provides spool access protection for entries in the RDR, LST, PUN, and XMT queues. This protection can be activated by setting the IPL SYS statement SEC=YES and SECAC=SYS in the SET statement in the VSE/POWER startup procedure. The SECAC operand in JOB, LST, and PUN statements can be used to provide further protection for spool access.

This support is available with the BSM and with an ESM.

PTF application from disk

Since VSE/ESA 2.4, you can apply service to your system from a VSAM-managed SAM file on disk, or you can use the z/VSE Virtual Tape (VTAPE) function to apply PTFs.

1.1.3 Migration issues

The complexity and variety of possible CICS configurations means that there is no single list of instructions for migration planning and implementation. Migration to the CICS TS for z/VSE 2.1 requires careful planning and change management. The following tasks must be completed:

- ▶ Understand the practical issues and detailed product changes intrinsic in migrating to the CICS TS for z/VSE 2.1.
- ▶ Plan a satisfactory conversion strategy.
- ▶ Plan the required actions to achieve that strategy.
- ▶ Accurately estimate the resources and time that is required to achieve that strategy.

You can adopt good change management practices by completing the following tasks:

- ▶ Making a minimum changes simultaneously.
- ▶ Testing all changes before placing them into production and having a fallback plan.
- ▶ Assessing the effect of the change before introducing it into production.
- ▶ Scheduling the change to have minimal impact on users of the system.

Generally, you should plan a phased conversion and cutover. For example, the following phases can be part of your strategy:

- ▶ Plan your migration.
- ▶ Position your system on CICS TS 1.1. (Clean up your existing system; for example, migrate DTSECTXN-based transaction security to profile based security.)
- ▶ Cutover to your new system in stages.

1.1.4 Migration planning summary

How you migrate your system depends on the current level of your system. Consider the following points:

- ▶ Ensure that you have the correct levels of software for the functions that you want to use.
- ▶ Ensure that you have the required hardware for the functions that you want to use.
- ▶ Do not run an XRF system with an active CICS system and an alternative CICS system at different release levels of CICS.
- ▶ Ensure that all third-party products you use can run under z/VSE 6.1 and the CICS TS. If they cannot run, you must obtain replacement products or PTFs.

1.2 z/VSE Version 6: What will be next

This section describes an IBM Statement of Direction.

IBM Statement of Direction disclaimer: IBM statements regarding plans, directions, and intent are subject to change or withdrawal without notice at the sole discretion of IBM. Information regarding potential future products is intended to outline general product direction and should not be relied on in making a purchasing decision. The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code, or functionality. Information about potential future products cannot be incorporated into any contract. The development, release, and timing of any future features or functionality described for our products remain at the sole discretion of IBM.

In addition to the functionality available with z/VSE 6.1, z/VSE will be further enhanced to use IBM z Systems z13®, IBM z13s™, and to use CICS functions that can provide constraint relief and allow for growing workloads.

Capability that is intended for delivery in a future release of IBM CICS Transaction Server for z/VSE includes (but is not limited to) the following enhancements:

- ▶ CICS Explorer enhancements are planned to:
 - Define new and change or delete CICS resources, such as programs, files, and transactions.
 - Monitor and control or update dynamic storage areas and global temporary storage queue statistics.
 - Use the “definitions views” for selected CICS resources.
- ▶ The following channels and containers enhancements are planned:
 - Support UTF-8 and UTF-16 in code page conversion by using Channels and Containers.
 - Add the APPEND parameter for PUT CONTAINER to append the specified data to data that is in a container.
 - Add the BYTEOFFSET parameter for GET CONTAINER to retrieve data beginning at a specified offset in a container.

The following enhancements that are related to CICS Transaction Server for z/VSE web services are planned:

- ▶ z/VSE SOAP Engine to use Channels and Containers

The existing z/VSE SOAP implementation integrates z/VSE CICS applications in a heterogeneous environment by using web services. User programs that use the z/VSE SOAP Engine are restricted by the COMMAREA and its 32 K limitation. To meet the needs of applications with growing data, z/VSE intends to use the CICS Channels and Containers API for the SOAP Engine.
- ▶ New z/VSE REST Engine with JSON support

Representational State Transfer (REST) is a software architecture style that consists of guidelines and best practices for creating web services. RESTful systems typically communicate over the Hypertext Transfer Protocol (HTTP) by using JavaScript Object Notation (JSON) or XML for the payload. z/VSE intends to provide a REST Engine that allows clients to provide RESTful web services that are running in a CICS environment. The REST Engine will support various payload types, including JSON and XML.

Enhancement to z/VSE Security:

- Basic Security Manager (BSM) enhancement

The z/VSE Basic Security Manager distinguishes between repositories for online and batch security definitions. The repository to protect batch resources is the phase DTSECTAB. To simplify the administration of batch resources, z/VSE intends to provide a common interface for online and batch resources. An Interactive UI (IUI) dialog box will be offered that builds a DTSECTAB with the specified resources.



Installation and tailoring

This chapter describes the necessary steps to install IBM z/VSE 6.1 with the IBM CICS Transaction Server (IBM CICS TS) and how to perform the migration and tailoring of CICS tables.

This chapter includes the following topics:

- ▶ 2.1, “Planning” on page 16
- ▶ 2.2, “Installing z/VSE 6.1 and CICS Transaction Server” on page 18
- ▶ 2.3, “Tailoring the CICS Transaction Server” on page 19
- ▶ 2.4, “Migration from CICS/VSE 2.3” on page 47

2.1 Planning

The planning phase of a system migration from pre-z/VSE 6.1 to z/VSE 6.1 is the most important part of the migration to ensure a successful migration.

2.1.1 Planning steps for installation

Installing the new version includes the following basic planning steps:

- ▶ Determine whether you use an installation disk or install by using the traditional tape device method.
- ▶ Define the optional products that you need to install.
- ▶ Define your hardware configuration.
- ▶ Determine the predefined environment that best fits your needs.
- ▶ Determine whether you want to run your system with security active.

2.1.2 Hardware considerations

This section does not cover all the hardware items. It describes only the main changes.

Processors

z/VSE 6.1 requires IBM System z10® or higher processor (z114, z196, zBC12, zEC12, z13, and z13s).

DASD

At installation time, you can select a system environment that suits your needs. The offered environments provide any of the following VSIZE environments:

- ▶ Environment A: 256 MB
- ▶ Environment B: 512 MB
- ▶ Environment C: 2 GB

The default page data set is allocated on the system DASD DOSRES. Ensure that the installation is done on DASD types that provide sufficient space (or better use the NOPDS option, running z/VSE without a page data set). The total virtual storage must fit into the available real storage.

Installation disk

Starting with z/VSE 5.2, installing z/VSE from a bootable installation disk is supported for an LPAR and an IBM z/VM® guest environment. z/VSE provides utilities to create a bootable installation disk, which is called installation disk for both the LPAR and the z/VM CMS environment. The layout of the installation disk is identical, no matter in which environment it was created.

In addition to the two DASD volumes that are used for system installation, a 3390 disk device with at least 500 cylinders is required for the installation disk.

For more information about how to create an installation disk, see *z/VSE Installation*, SC34-2678.

2.1.3 Software considerations

Fast Service Upgrade (FSU) to z/VSE 6.1 from prior releases is not possible. An initial installation of the base products and optional product installation is required.

You might have to apply more program temporary fixes (PTFs) for the system to run successfully. For more information about Preventive Service Planning (PSP) and the latest Recommended Service Level (RSL), see [the IBM z/VSE website](#).

To support adequate encryption for web applications, CICS Web Support requires more current cipher suites (AES128/256). For network connectivity, you have the choice from the following vendors that provide TCP/IP stacks for z/VSE:

- ▶ IBM IPv6/VSE 1.2: New release of the Barnard Software Inc. TCP/IP stack
- ▶ IBM TCP/IP for z/VSE 2.1: New version of the CSI International TCP/IP stack (requires new license)

Both vendors feature upgraded functionality for z/VSE 6.1. Both products require a license key to be used.

CICS Transaction Server 2.1 does not provide any means to support CICS/VSE macro applications, nor does it provide extensive tools for migrating such applications. If you still use CICS/VSE 2.3 because of applications that are not eligible for CICS TS, the migration path is by using CICS Transaction Server 1.1, which provides tools for migrating such applications. The last z/VSE release that can run CICS/VSE 2.3 was z/VSE 4.3, which allows for implementing a CICS coexistence environment, which offers a smooth migration path.

Consider the following points:

- ▶ CICS TS 1.1 cannot be used with z/VSE 6.1.
- ▶ LE/VSE is shipped as part of the z/VSE base.
- ▶ OSA/SF for z/VSE is included in the VSE base tape.

Table 2-1 lists the dropped z/VSE optional programs. For more information about optional products, see *z/VSE Planning*, SC34-2681.

Table 2-1 Dropped optional programs for z/VSE 6.1

Optional program	Program number
IBM WebSphere MQ for z/VSE V3.0	5655-U97
Emulation Program (EP) V1.14	5735-XXB
BM Advanced Communication Function/System Support Program (ACF/SSP) for VSE 4.8.1	5686-064
IBM Overlay Generation Language (OGL/370) 1.1.0	5688-191
IBM Advanced Communication Function/Network Control Program (ACF/NCP) 7.8.1	5648-063
IBM X.25 NCP Packet Switching Interface (NPSI) 3.9.0	5688-035

VS COBOL II, DOS/VS COBOL, C/370, and DOS PL/I are no longer available as z/VSE optional products. Consider the following points:

- ▶ CICS Transaction Server does not support C application programs that were compiled by using the C/370 Compiler. All such application programs must be recompiled z/VSE V6R1 using the C for VSE/ESA compiler and link-edited using Language Environment for VSE (LE/VSE).
- ▶ CICS Transaction Server does not support PL/I application programs that were compiled by using the DOS PL/I compiler. All such application programs must be recompiled by using the PL/I for VSE/ESA compiler and link-edited with LE/VSE.
- ▶ CICS command-level application programs that were compiled by using the VS/COBOL II and DOS/VS COBOL compilers should be link-edited with LE/VSE to obtain limited support.

If you have any CICS applications that were written in VS/COBOL II, you might need to review your COBOL runtime options.

- ▶ CICS Distributed Data Management (CICS/DDM) is not supported by CICS TS for z/VSE V2.1.

2.1.4 Other considerations

Ensure that all of your third-party or vendor software products are supported by z/VSE 6.1 and CICS Transaction Server for z/VSE 2.1. For more information about the required levels for z/VSE 6.1 and CICS TS, contact your vendor.

2.2 Installing z/VSE 6.1 and CICS Transaction Server

z/VSE is a pre-generated system that is easy to install. The CICS TS is installed as part of the base installation process. z/VSE is shipped on three base tapes; the second tape is the extended base tape and the third tape contains IBM DB2® help material. The z/VSE installation material is also available on a single DVD-ROM.

You can choose between an automatic installation or a manual installation. We recommend the automatic installation of environment B. Table 2-2 lists the available environments.

Table 2-2 IBM predefined environments

Environment	A	B	C
Number of address spaces	12 + 57	12 + 57	12 + 57
VSIZE	256 MB	512 MB	2 GB
Maximum number of active parts concurrently (NPARTS)	60	80	120
PASIZE	50 MB	150 MB	512 MB
ALLOC for CICS in F2	50 MB	50 MB	256 MB
ALLOC for CICS in F8	50 MB	150 MB	512 MB
Maximum size of data space (DSPACE)	20 MB	20 MB	20 MB

CICS TS delegates all security checks to an external approving authority. This authority can be the Basic Security Manager (BSM) included in z/VSE, or an external security manager provided by a vendor. Online security is enabled in z/VSE by default, and more security for batch operation is recommended and can be requested during initial installation (this request sets IPL SYS SEC=YES). For more information, see Chapter 3, “Security” on page 65.

2.3 Tailoring the CICS Transaction Server

This section provides information about the following main CICS TS tailoring tasks:

- ▶ Reviewing virtual storage requirements for CICS TS
- ▶ Using a CICS system definition data set file
- ▶ Defining system files for a second CICS TS
- ▶ Customizing the DFHSIT table
- ▶ Customizing the DFHPLT tables
- ▶ Tailoring the CICS start-up job stream
- ▶ Autoinstall of resources

2.3.1 Reviewing virtual storage requirements for CICS TS

The storage allocations that are defined for the CICS TS in environment C are 256 MB for F2 for CICSICCF and 512 MB for F8 for a second CICS. Figure 2-1 on page 20 shows the layout for the CICS TS address space.

For more information about CICS virtual storage requirements, see Chapter 8., “Performance and tuning” on page 193.

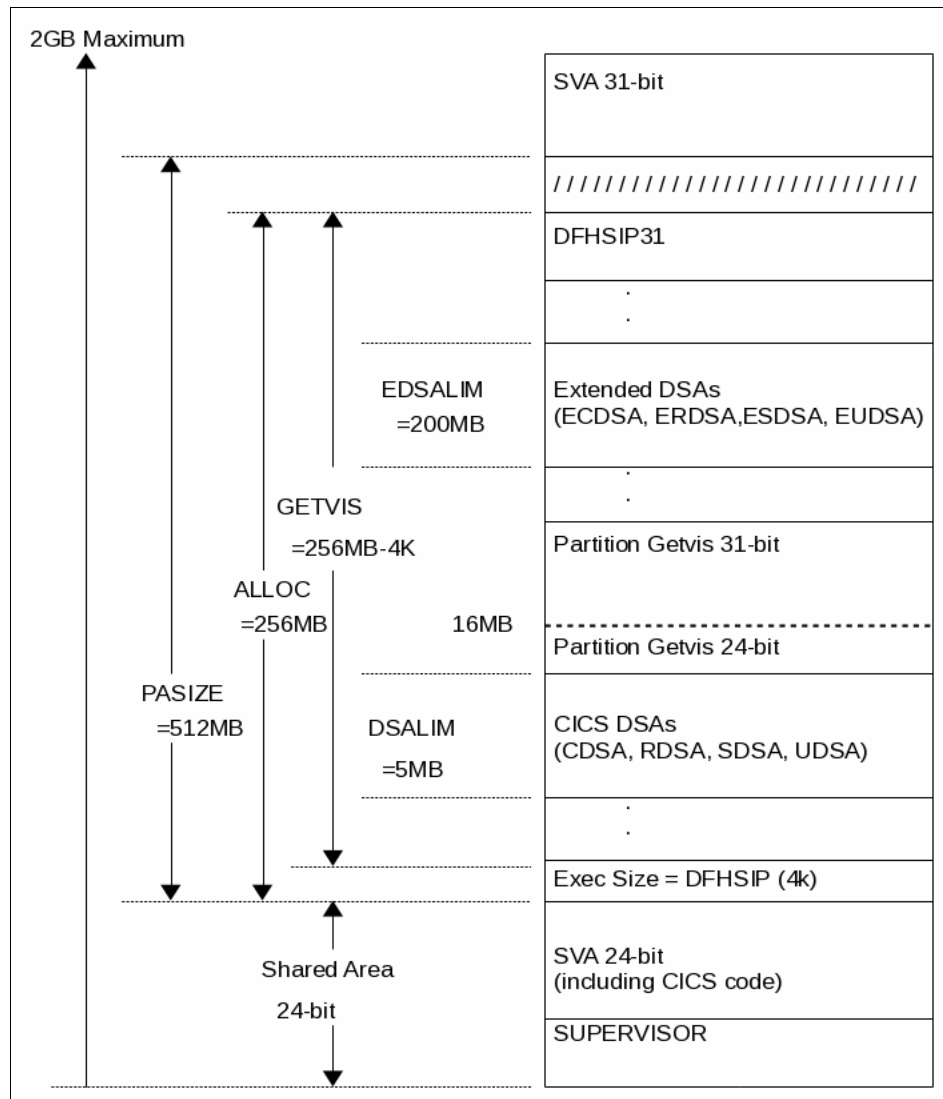


Figure 2-1 CICS address space

The minimum ALLOC that is required is 30 MB. Increasing partition sizes might also mean that you must increase the VSIZE to meet your total system requirements.

2.3.2 Using a CICS system definition data set file

The use of a CICS system definition data set (CSD) file for resource definitions is mandatory for CICS Transaction Server. This requirement makes the CSD file a critical resource of your system. You should ensure that this file is backed up regularly.

The DFHCSD data set is defined during the installation of z/VSE.

After initial installation of z/VSE 6.1, the DFHCSD data set contains the IBM-supplied CICS TS for z/VSE 2.1 and z/VSE 6.1 resource definitions.

If you want to migrate your own CSD from a previous z/VSE release, complete the following tasks:

- ▶ On z/VSE 6.1 update the CSD with your own resource definitions by using the DFHCSDUP utility program.
- ▶ To get your own resource definitions, run the DFHCSDUP utility program with the LIST ALL function on your previous z/VSE release to list all resource definitions in the CSD data set.

Alternatively, use the **DFHCSDUP EXTRACT** command to extract resource definition data from the CSD (from a list or from a group) and invoke a user program, which is the CICS supplied sample program DFH0CBDC). This program produces a file of DFHCSDUP DEFINE control statements. The file can be used to re-create or add resource definitions to any CSD by using DFHCSDUP.

If you use your own CSD as a base and include IBM-supplied resource definitions, ensure that you do not back-level the IBM-supplied resource definitions.

Resource definitions that are supplied by IBM include, for example, Language Environment resource definitions. You can use job skeleton SKCSDFIL in the ICCF library 59 to establish the IBM supplied resource definitions for z/VSE 6.1. For more information, see [IBM z/VSE Administration, Version 3 Release 1](#).

Note: Consider the following points:

- ▶ Do not use the **DFHCSDUP UPGRADE** command (except when skeleton SKCSDFIL is used).
- ▶ Do not use VSE/VSAM utilities to migrate your CSD data set to avoid back-level IBM supplied resource definitions.
- ▶ If you choose another method to migrate your CSD data set, ensure that you do not back-level the IBM-supplied resource definitions.

2.3.3 Defining system files for a second CICS TS

The system files that we used for our second CICS TS are listed in Table 2-3.

Table 2-3 System files used for second CICS TS

File name	File identifier	File type
DFHAUXT	CICS2.AUXTRACE (1)	SAM
DFHCSD	CICS.CSD (2)	VSAM KSDS
DFHDMFA	CICSTS.SYSTEM.DFHDMFA (2)	VSAM ESDS
DFHDMFB	CICSTS.SYSTEM.DFHDMFB (2)	VSAM ESDS
DFHDMPA	CICS2.DUMPA (Note 1)	SAM
DFHDMPB	CICS2.DUMPB (Note 1)	SAM
DFHGCD	CICS2.GCD	VSAM KSDS
DFHLCD	CICS2.LCD	VSAM KSDS

File name	File identifier	File type
DFHNTRA	CICS2.TD.INTRA	VSAM ESDS
DFHRSD	CICS2.RSD	VSAM KSDS
DFHTEMP	CICS2.DFHTEMP	VSAM ESDS
IESCNTL	VSE.CONTROL.FILE (Note 2)	VSAM KSDS
IESLDUM	VSE.LDAP.USER.MAPPING (Note 2)	VSAM KSDS
IESPRB	CICS2.ONLINE.PROB.DET.FILE	VSAM KSDS
IESROUT	VSE.MESSAGE.ROUTING.FILE (Note 2)	VSAM KSDS
IESTRFL	VSE.TEXT.REPSTORY.FILE (Note 2)	VSAM KSDS
BSTCNTL	VSE.BSTCNTL.FILE (Note 2)	VSAM KSDS

Consider the following points:

- ▶ The file is created dynamically when required.
- ▶ The file can be shared with DBDCCICS (the primary CICS TS).

Two VSAM data sets are used for cataloging information about the running CICS system.

The following file names are used:

- ▶ DFHLCD for the local catalog
- ▶ DFHGCD for the global catalog

The catalogs contain system information for use in WARM and EMERgency restarts, including takeovers by an alternative CICS partition if running with extended recovery facility (XRF). The global catalog can also contain information that is used in a cold start. If you delete and redefine a catalog, you must delete and redefine them both.

DFHRSD refers to a data set that is used only for system log processing during EMERgency restart.

The DFHDMF data sets are used for statistics and monitoring data. A minimum of two data sets should be available for use by DMF. For more information, see Chapter 8., “Performance and tuning” on page 193.

System file definitions and initialization

We defined the VSAM files for our second CICS partition by using the skeleton SKPREPC2 in ICCF library 59.

Figure 2-2 on page 23 shows the first part of the CICS system file definitions we used.

```

* $$ JOB JNM=VSAMDEF2,DISP=D,CLASS=0
// JOB VSAMDEF2 - DEFINE VSAM CLUSTERS FOR SECOND CICS
* *****
* DEFINE AND INITIALIZE VSAM FILES FOR CICS2
* *****
// EXEC IDCAMS,SIZE=AUTO
/*                                                     */
/* DELETE VSAM FILES                                  */
/*                                                     */
    DELETE (CICS2.GCD) CL NOERASE PURGE -
    CATALOG(VSESP.USER.CATALOG)
    DELETE (CICS2.LCD) CL NOERASE PURGE -
    CATALOG(VSESP.USER.CATALOG)
    DELETE (CICS2.ONLINE.PROB.DET.FILE) CL NOERASE PURGE -
    CATALOG(VSESP.USER.CATALOG)
    DELETE (CICS2.DFHTEMP) CL NOERASE PURGE -
    CATALOG(VSESP.USER.CATALOG)
    DELETE (CICS2.TD.INTRA) CL NOERASE PURGE -
    CATALOG(VSESP.USER.CATALOG)
    DELETE (CICS2.RSD) CL NOERASE PURGE -
    CATALOG(VSESP.USER.CATALOG)
    SET MAXCC = 0
/*                                                     */
/* DEFINE VSAM FILES                                  */
/*                                                     */
/*                                                     */
DEFINE CLUSTER(NAME(CICS2.GCD)                -
    RECORDSIZE (4089 4089)                    -
    RECORDS (2000 200)                        -
    KEYS (28 0)                               -
    REUSE                                     -
    INDEXED                                  -
    FREESPACE (10 10)                         -
    SHR(2)                                    -
    CISZ(8192)                                -
    VOL(SYSWK1 DOSRES))                       -
    DATA(NAME(CICS2.GCD.@D@))                -
    INDEX (NAME (CICS2.GCD.@I@))              -
    CATALOG(VSESP.USER.CATALOG)
DEFINE CLUSTER(NAME(CICS2.LCD)                -
    INDEXED                                  -
    RECORDSIZE (45 124)                       -
    RECORDS (3000 200)                        -
    KEYS (28 0)                               -
    REUSE                                     -
    FREESPACE (10 10)                         -
    SHR(2)                                    -
    CISZ(2048)                                -
    VOL(SYSWK1 DOSRES))                       -
    DATA(NAME(CICS2.LCD.@D@))                -
    INDEX (NAME (CICS2.LCD.@I@))              -
    CATALOG(VSESP.USER.CATALOG)

```

Figure 2-2 (Part 1 of 2) CICS system file definitions

Figure 2-3 shows part 2 of the CICS system file definitions.

```

DEF CLUSTER(NAME(CICS2.ONLINE.PROB.DET.FILE) -
    FILE(IESPRB) -
    VOL(SYSWK1 DOSRES) -
    RECORDS (300 100) -
    RECORDSIZE (4000 4089) -
    INDEXED -
    KEYS(2 0) -
    SHR(2)) -
    DATA (NAME (CICS2.ONLINE.PROB.DET.FILE.@D@) CISZ(4096)) -
    INDEX (NAME (CICS2.ONLINE.PROB.DET.FILE.@I@) CISZ(512)) -
    CATALOG(VSESP.USER.CATALOG)
/* */
DEF CLUSTER(NAME(CICS2.DFHTEMP) -
    VOL(SYSWK1 DOSRES) -
    RECORDS (100) -
    RECORDSIZE (16377 16377) -
    CISZ (16384) -
    NONINDEXED -
    SHR(2)) -
    DATA(NAME(CICS2.DFHTEMP.ESDS)) -
    CATALOG(VSESP.USER.CATALOG)
/* */
DEF CLUSTER(NAME(CICS2.TD.INTRA) -
    VOL(SYSWK1 DOSRES) -
    RECORDS (100) -
    RECORDSIZE (4089 4089) -
    CISZ (4096) -
    NONINDEXED -
    SHR(2)) -
    DATA(NAME(CICS2.TD.INTRA.ESDS)) -
    CATALOG(VSESP.USER.CATALOG)
/* */
DEF CLUSTER(NAME(CICS2.RSD) -
    INDEXED -
    RECORDSIZE (2000 2000) -
    RECORDS (250 100) -
    KEYS (22 0) -
    FREESPACE (20 20) -
    SHR(2) -
    VOL(SYSWK1 DOSRES)) -
    DATA(NAME(CICS2.RSD.@D@)) -
    INDEX (NAME (CICS2.RSD.@I@)) -
    CATALOG(VSESP.USER.CATALOG)

```

Figure 2-3 (Part 2 of 2) CICS system file definitions

Figure 2-4 shows sample code to initialize the Restart Data Set, Global Catalog, and Local Catalog files.

```
*      INITIALIZE THE CICS2 RESTART DATA SET
*
// DLBL DFHRSD,'CICS2.RSD',0,VSAM,CAT=VSESPUC
// DLBL DFHGCD,'CICS2.GCD',0,VSAM,                X
        CAT=VSESPUC
// EXEC IDCAMS,SIZE=AUTO
    REPRO INFILE -
        (SYSIPT -
        ENVIRONMENT -
        (RECORDFORMAT (FIXUNB) -
        BLOCKSIZE(80) -
        RECORDSIZE (80))) -
    OUTFILE (DFHRSD)
ACTL 0001
/*
// EXEC IDCAMS,SIZE=AUTO      INIT GCD FILE
    REPRO INFILE -
        (SYSIPT -
        ENVIRONMENT -
        (RECORDFORMAT (FIXUNB) -
        BLOCKSIZE(80) -
        RECORDSIZE(80))) -
    OUTFILE (DFHGCD)
+++++
/*
// DLBL DFHLCD,'CICS2.LCD',0,VSAM,                X
        CAT=VSESPUC
// LIBDEF *,SEARCH=(PRD2.CONFIG,PRD2.SCEEBASE,PRD1.BASE)
// EXEC DFHCCUTL,SIZE=300K      INITIALIZE CICS CATALOG
/*
```

Figure 2-4 Initializing CICS files

Create the system list that you name in the GRPLIST parameter by adding all resource definitions that are required by CICS (see Figure 2-5).

```
// LIBDEF *,SEARCH=(PRD2.CONFIG,PRD1.BASE,PRD2.SCEEBASE)
// EXEC DFHCSDUP,SIZE=600K      INIT AND LOAD CICS
DELETE GROUP(FCTC2)
DELETE LIST(VSELST2)
ADD GROUP(VSETYPE) LIST(VSELST2)
ADD GROUP(VSETERM) LIST(VSELST2)
ADD GROUP(VSETERM1) LIST(VSELST2)
APPEND LIST(DFHLIST) TO(VSELST2)
ADD GROUP(DFHRCF) LIST(VSELST2)
ADD GROUP(DFHCLNT) LIST(VSELST2)
ADD GROUP(CICREXX) LIST(VSELST2)
ADD GROUP(TCPIP) LIST(VSELST2)
ADD GROUP(VSEAI62) LIST(VSELST2)
ADD GROUP(EZA) LIST(VSELST2)
ADD GROUP(DFH$WBSN) LIST(VSELST2)
* $$ SLI MEM=IESZFCT2.Z
ADD GROUP(VSESPG) LIST(VSELST2)
ADD GROUP(FCTC2) LIST(VSELST2)
ADD GROUP(CEE) LIST(VSELST2)
LIST ALL
/*
```

Figure 2-5 Creating CSD GRPLIST VSELST2

Figure 2-6 shows the sample code to initialize the global variables definition in the CPUVARx procedure.

```
// JOB GLOBVAR - DEFINE GLOBAL VARIABLES
// SETPARM XNCPU=' '
// EXEC PROC=$COMVAR,XNCPU
// EXEC DTRSETP,PARM='CPUVAR&XNCPU'
SET XPARTC2='F8'
SET XUSEF8='CI'
SET XAPPLF8='PRODCICS'
/*
```

Figure 2-6 Global variables definition

Change the file's shareoption from 2 to 4, so that these files can be accessed by multiple CICS partitions, as shown in Figure 2-7.

```
* PLEASE CLOSE FILES IESROUT AND INWFILE ON DBDCCICS
* AND ALSO ON ALL OTHER CICS PARTITIONS USING THE FILES.
*       CEMT SET FI(XXXXXXX) CLOSE
* A RETURN CODE OF 4 IS OK. IF THE INWFILE DOES NOT EXIST, RETURN
* CODE WILL BE 12.
* IF OTHER FILES SHOULD ALSO BE SHARED AMONG SYSTEMS CHANGE
* THE SHAREOPTIONS ACCORDINGLY.
// PAUSE
// EXEC IDCAMS
  ALTER VSE.MESSAGE.ROUTING.FILE.@I@ -
  SHAREOPTIONS(4) -
  CATALOG(VSESP.USER.CATALOG)
  /**/
  ALTER VSE.MESSAGE.ROUTING.FILE.@D@ -
  SHAREOPTIONS(4) -
  CATALOG(VSESP.USER.CATALOG)
  /**/
  ALTER PC.HOST.TRANSFER.FILE.INDEX -
  SHAREOPTIONS(4) -
  CATALOG(VSESP.USER.CATALOG)
  /**/
  ALTER PC.HOST.TRANSFER.FILE.DATA -
  SHAREOPTIONS(4) -
  CATALOG(VSESP.USER.CATALOG)
/*
```

Figure 2-7 Shareoption change for sharing files with DBDCCICS

The catalog procedure with all non-shared CICS file labels is shown in Figure 2-8.

```

CATALOG DTRCICS2.PROC D=YES R=YES EOD=/+
// ASSGN SYS018,DISK,VOL=SYSWK1,SHR
// DLBL DFHMPA,'CICS2.DUMPA',0,VSAM,
CAT=VSESPUC,RECSIZE=7200,
DISP=(NEW,KEEP),RECORDS=(300,0)
// DLBL DFHMPB,'CICS2.DUMPB',0,VSAM,
CAT=VSESPUC,RECSIZE=7200,
DISP=(NEW,KEEP),RECORDS=(100,0)
// DLBL DFHAUXT,'CICS2.AUXTRACE',0,VSAM,
CAT=VSESPUC,RECSIZE=4096,
DISP=(NEW,KEEP),RECORDS=(400,0)
// DLBL DFHTEMP,'CICS2.DFHTEMP',0,VSAM,
CAT=VSESPUC
// DLBL DFHNTRA,'CICS2.TD.INTRA',0,VSAM,
CAT=VSESPUC
// DLBL DFHRSD,'CICS2.RSD',0,VSAM,
CAT=VSESPUC
// DLBL DFHLCD,'CICS2.LCD',0,VSAM,
CAT=VSESPUC
// DLBL DFHGCD,'CICS2.GCD',0,VSAM,
CAT=VSESPUC
// DLBL IESPRB,'CICS2.ONLINE.PROB.DET.FILE',,VSAM,
CAT=VSESPUC
// DLBL EYUPARM,'EYUPARM.FILE',0,SD
// EXTENT SYS045,SYSWK1,1,0,63210,1
/+

```

Figure 2-8 File labels procedure

Modify and catalog the CICS startup job and load it in the POWER reader queue, as shown in Figure 2-9 on page 29.


```

* $$ JOB JNM=CATCICS2,DISP=D,CLASS=0
// JOB CATCICS2                CATALOG CICS2 AND LDCICS2
// EXEC LIBR,PARM='MSHP'
// ACCESS S=IJSYSRS.SYSLIB
// CATALOG CICS2.Z REPLACE=YES
$$$$ JOB JNM=CICS2,DISP=L,CLASS=8,E0JMSG=YES
$$$$ LST CLASS=A,DISP=D,RBS=100
// JOB CICS2                STARTUP OF SECOND CICS WITHOUT ICCF
// OPTION SADUMP=5
// OPTION SYSDUMPC
// LIBDEF *,SEARCH=(PRD2.CONFIG,PRD2.TCPIPC,PRD1.BASED,PRD1.BASE,      X
//                          PRD2.PROD,PRD2.SCEEBASD,PRD2.SCEEBASE,PRD2.DBASE,      X
//                          PRD2.DFHDOC)
// LIBDEF DUMP,CATALOG=SYSDUMP.F8
// SETPARM XNCPU=''
// SETPARM XENVNR=''
// SETPARM XMODEF8=AUTO
// SETPARM XAPPLF8=''
// EXEC PROC=$COMVAR,XNCPU
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;SET XSTATF8=ACTIVE' **F8 ASSUMED
$$/*
// EXEC PROC=CPUVAR&XNCPU,XMODEF8,XAPPLF8,XENVNR **F8 ASSUMED
// SETPFIX LIMIT=256K
LOG
// ID USER=PRODCICS
NOLOG
// EXEC PROC=DTRCICS2                LABELS FOR CICS FILES
*   WAITING FOR VTAM TO COME UP
// EXEC IESWAITT
$$/*
*   WAITING FOR TCP/IP TO COME UP
* // EXEC REXX=IESWAITR,PARM='TCPIP00'
$$/*
// OPTION SYSPARM='00'
// ASSGN SYS020,SYSLST
// ASSGN SYS045,DISK,VOL=SYSWK1,SHR
* // ASSGN SYS023,DISK,VOL=DOSRES,SHR   IF JOURNALLING
// IF XENVNR = A THEN
// SETPARM ELIM=25M
// IF XENVNR = B THEN
// SETPARM ELIM=120M
// IF XENVNR = C THEN
// SETPARM ELIM=450M
// IF XMODEF8 = COLD THEN
// GOTO COLDST
// SETPARM XMODEF8=AUTO
// GOTO STARTCIC
/. COLDST
// SETPARM XMODEF8=COLD
/. STARTCIC

```

Figure 2-9 Catalog CICS startup and load into RDR queue (Part 1 of 2)

Figure 2-10 shows part 2 of the job to modify and catalog the CICS startup job and load it in the POWER reader queue.

```
// EXEC DFHSIP,SIZE=DFHSIP,PARM='APPLID=&XAPPLF8.,START=&XMODEF8.,EDSAL* .1.
          IM=&ELIM.,SI',DSPACE=2M,OS390 .2.
SIT=C2,STATRCD=OFF,NEWSIT=YES, .3.
$$$$ SLI MEM=IESVAEXC.Z,S=IJSYSRS.SYSLIB .4.
$/ *
// EXEC DTRSETP,PARM='CPUVAR&XNCPU;;SET XSTATF8=INACTIVE'
$/ *
$/ &
$$$$ EOJ
/+
CATALOG LDCICS2.PROC      REPLACE=YES DATA=YES
// EXEC DTRIINIT
      LOAD CICS2.Z
/*
/+
CONNECT S=IJSYSRS.SYSLIB:PRD2.SAVE
COPY CICS2.Z R=Y
/*
* // EXEC PROC=LDCICS2      LOAD CICS2 INTO RDR QUEUE
/*
/ &
* $$ EOJ
```

Figure 2-10 Catalog CICS startup and load into RDR queue (Part 2 of 2)

The following numbered statements (see Figure 2-10) explain how CICS is started in our startup job:

1. To override SIT parameters, we used the PARM parameter and also the SYSIN keyword (SI) to tell CICS to read system initialization parameters from the SYSIPT data set
2. The CICS TS must run in OS390 emulation mode. To activate OS390 emulation mode, code the OS390 parameter at the end of the // EXEC statement.
3. Overriding SIT with SYSIPT data.
4. SIT specifies SVA=YES uses exclude list to prohibit module usage from SVA.

2.3.4 Customizing the DFHSIT table

You must modify the system initialization tables by removing all obsolete parameters, and specifying the required values for new or changed parameters if you want to run with other than the defaults. After this action, reassemble the tables by using CICS TS macro libraries.

Note: If you use a DFHSITxx table that was built on a CICS TS for VSE/ESA system, the CICS TS for z/VSE startup fails and the following message is displayed (411 is the internal CICS release of CICS TS for VSE/ESA 1.1.1):

```
DFHPA1107 applid A 411 VERSION OF MODULE DFHSITxx WAS LOADED. CICS CAN ONLY
INITIALIZE WITH THE CURRENT LEVEL SIT.
```

Table 2-4 lists the major changes in system initialization parameters. For more information, see the *CICS Transaction Server for VSE/ESA System Definition Guide*.

Table 2-4 Summary of DFHSIT parameter changes

Area	Parameter	Remarks
Data conversion	CLINTCP	New
Data conversion	LOCALCCSID	New
Data conversion	SERVERCP	New

CICS system initialization table generation

The following definition is the source of the DFHSITC2 table. You can use this definition as a model to tailor and assemble your own table, depending on your requirements.

We used the DFHSIT skeleton from ICCF library 59.

Figure 2-11 on page 32 shows the DFHSIT for the second CICS TS.

```

*****
TITLE 'DFHSITC2  -- SIT FOR CICS TS - APPLID PRODCICS'
PUNCH ' CATALOG DFHSITC2.OBJ REP=YES'
DFHSIT TYPE=CSECT,
    AIEXIT=IESZATDX,          AUTO INSTALL TERMINALS
    AILDELAY=200,             AUTO INSTALL DEL TERM  PQ03810*
    AIQMAX=100,               AUTO INSTALL CONC TERMINALS
    AIRDELAY=700,             AUTO INSTALL ELAPS TIME
    AKPFREQ=200,              ACTIVITY KEYPOINTING FREQUENCY
    APPLID=PRODCICS,          CICS APPLICATION NAME
    AUXTR=OFF,                AUXTRACE OFF
    BMS=(FULL,,UNALIGN,NODDS), FULL BASIC MAPPING SUPPORT
    CLINTCP=437,              CODEPAGE CLIENT
    CLSDSTP=NOTIFY,           NOTIFICATION ISSUE PASS
    CMDPROT=YES,              VALIDATE START ADDRESSES
    CMDSEC=ASIS,              CMDSEC WILL BE HONORED
    CONFDATA=SHOW,            SHOW USER DATA IN TRACE
    CONFTXT=NO,               VTAM SHOW USER DATA
    CSDACC=READWRITE,         CSD MAY BE UPDATED
    CSDJID=NO,                JOURNALLING OF FILE REQUESTS
    CSDLSRNO=1,               CSD LOCAL SHARED RESOURCE
    CSDRECOV=NONE,            CSD NOT RECOVERABLE
    CSDSTRNO=4,               CSD SIMULTANEOUS ACCESS
    CWAKEY=USER,              COMMON WORK AREA KEY
    DATFORM=MMDDYY,           EXTERNAL DATE DISPLAY
    DBP=1$,                   DYN. BACKOUT (NO LOCAL DLI I/F)*
    DBUFSZ=2000,              DYN. ADJUSTED BY CICS
    DCT=C2,                   FOR SECOND CICS
    DFLTUSER=CICSUSER,        DEFAULT USER
    DIP=NO,                   BATCH INTERCHANGE PROGRAM
    DISMACP=YES,              ASRD ABEND IN CASE MACROS I/F
    DLI=NO,                   NO DL/I SUPPORT
    DLIOER=ABEND,             ABEND IN CASE OF A DLI IOERR
    DOCCODEPAGE=037,          CODE PAGE
    DSALIM=5M,                UPPER LIMIT OF STORAGE BELOW
    DTRPGM=DFHDYP,            PROGRAM DYNAMIC XACTION ROUTING*
    DTRTRAN=CRTX,             XACTION DYNAMIC XACTION ROUTING*
    DUMP=YES,                 IDUMP IN ABEND SITUATIONS
    DUMPDS=AUTO,              AUTO SWITCH DUMP DATA SETS
    DUMPSW=NEXT,              USERS MAY NOT USE DUMPSW S
    EDSALIM=25M,              DSA ABOVE THE LINE ENV. B
    ENCRYPTION=NORMAL,        SSL ENCRYPTION
    EODI=E0,                  END OF DATA SEQUENTIAL DEVICES
    ESMEXITS=INSTLN,          INSTLN PARAMETER IN RACROUTE
    FCT=NO,                   FOR SECOND CICS
    FEPI=NO,                  NO FRONT END PROGRAMMING I/F
    GMTEXT='z/VSE CICS2',     GOOD MORNING MESSAGE TEXT
    GMTRAN=IEGM,              LOGON TRANSACTION ID
    GNTRAN=IEGT,              TIME OUT TRANSACTION
    GRPLIST=VSELST2,          AUTOINST. TERMINALS. AND MRO

```

Figure 2-11 DFHSIT for second CICS TS (Part 1 of 4)

Figure 2-12 shows the DFHSIT for the second CICS TS.

ICP=COLD,	INTERVAL CONTROL PGM	*
ICV=1000,	INTERVAL CONTROL EXIT TIME-MS	*
ICVR=5000,	RUNAWAY TASK TIME	*
ICVTS=100,	TERMINAL SCAN DELAY	*
INTTR=ON,	INTERNAL TRACE	*
IRCSTRT=NO,	START IRC DURING INITIALIZATION*	
ISC=YES,	INTERSYSTEM COMMUNICATION	*
JCT=NO,	NO JOURNALLING	*
KEYFILE=CRYPTO.KEYRING,	KEY RESIDENCE FOR SSL	*
LEVSE=YES,	SUPPORT LE ON THIS CICS	*
LGNMSG=YES,	VTAM LOGON DATA	*
LOCALCCSID=037,	CCSID LOCAL REGION	*
MCT=NO,	NO MONITOR CONTROL TABLE	*
MN=OFF,	MONITORING OFF	*
MNCONV=NO,	NO MONITORING OF CONVERSATIONAL*	
MNEXC=OFF,	MONITORING EXCEPTION CLASS	*
MNFREQ=0,	MONITORING FREQUENCY	*
MNPER=OFF,	MONITORING PERFORMANCE CLASS	*
MNSYNC=NO,	MONITORING SYNCPOINT	*
MNTIME=LOCAL,	MONITORING TIME GMT	*
MROBTCH=1,	MRO BATCHING EVENTS	*
MROFSE=NO,	MRO LIFETIME LONG-RUNNING EXT.	*
MROLRM=YES,	MRO LONG RUNNING MIRROR TASK	*
MSGCASE=MIXED,	MESSAGES IN MIXED CASE	*
MSGLVL=1,	ALL MESSAGES	*
MXT=50,	MAX NO. OF ALL CONCURRENT TASKS*	
NATLANG=E,	(E,X) X = S,G,... NLS	*
OPERTIM=120,	WTO TIMEOUT	*
PARMERR=INTERACT,	INTERACT IF OF PARAMETERS WRONG*	
PGAICTLG=ALL,	UPDATE AUTOINSTALL PGM DEFINING*	
PGAEXIT=DFHPGADX,	PGM AUTOINSTALL EXIT	*
PGAIPGM=ACTIVE,	PGM AUTOINSTALL ACTIVE	*
PGCHAIN=X/,	BMS CHAINING COMMAND	*
PGCOPY=COPY/,	BMS COPY COMMAND	*
PGPURGE=T/,	BMS PURGE COMMAND	*
PGRET=P/,	BMS RETRIEVAL COMMAND	*
PLTPI=P2,	POST-INITIALIZATION PLT	*
PLTPISEC=CMDSEC,	POST-INITIALIZATION PLT SECURE	*
PLTPIUSR=CICSUSER,	POST-INITIALIZATION PLT USER	*
PLTSD=S2,	SHUTDOWN PLT	*
PRGDLAY=100,	ONE HOUR PURGE DELAY	*
PRINT=PA1,	PRINT WITH PA1 AND TCP PRINT	*
PRTYAGE=5000,	PRIORITY AGING 5 SECONDS	*
RAMAX=256,	SIZE OF I/O AREA FOR RA	*
RAPOOL=10,	NUMBER OF FIXED RPLS	*
RENTPGM=PROTECT,	READ DSA FROM KEY 0 PROTECTED	*
RESP=FME,	FUNCTION MANAGEMENT END	*
RESSEC=ASIS,	NONOR RESSEC	*
RMTRAN=(CSGM,CSGM),	SYSTEM RECOVERY RESTART	*
RUWAPOL=YES,	RESERVE STORAGE RUN UNIT	*

Figure 2-12 DFHSIT for second CICS TS (Part 2 of 4)

Figure 2-13 shows the DFHSIT for the second CICS TS.

SEC=YES,	SIGNON SECURITY, DON'T CHANGE	*
SECPRFX=NO,	NO SECURITY PREFIX	*
SNSCOPE=NONE,	SIGNON MORE THAN ONCE	*
SPCTR=1,	SPECTRUM OF TRACE	*
SPOOL=(YES,B,A),	CICS SPOOLER ACTIVE	*
SRT=1\$,	DEFAULT SRT	*
SRVERCP=037,	CODEPAGE SERVER	*
SSLDELAY=600,	DELAY FOR SSL CONNECTION	*
START=AUTO,	LET CICS DETERMINE STARTUP	*
STATRCD=ON,	STATISTICS RECORDING	*
STGPROT=YES,	STORAGE PROTECTION	*
STGRVCY=YES,	RECOVER FROM STORAGE VIOLATION	*
STNTR=1,	STANDARD TRACING	*
SUFFIX=C2,	FOR SECOND CICS	*
SVA=YES,	SVA LOADING, EXCLUDE LIST	*
SYDUMAX=1,	ONLY ONE DUMP PER TABLE ENTRY	*
SYSIDNT=CIC2,	IDENTIFIER OF THIS CICS	*
SYSTR=ON,	ALLOW SYSTR CODING	*
TCP=YES,	TERMINAL CONTROL PROGRAM	*
TCPIP=YES,	CWS OVER TCP/IP	*
TCSACTN=NONE,	TERMINAL CONTROL SHUTDOWN	*
TCSWAIT=4,	TERMINAL CONTROL SHUTDOWN WAIT	*
TCT=NO,	FOR AUTOINSTALLED TERMINALS	*
TCTUAKEY=USER,	TCTUA STORAGE KEY	*
TCTUALOC=ANY,	ABOVE ALSO ALLOWED	*
TD=(3,3),	THREE BUFFERS & THREE STRINGS	*
TRAP=OFF,	FE TRAP OFF	*
TRDUMAX=1,	1 XACTION DUMP PER TABLE ENTRY	*
TRTABSZ=4096,	SIZE OF INTERNAL TRACE TABLE	*
TRTRANSZ=4096,	TRANSACTION DUMP TRACE SIZE	*
TRTRANZY=TRAN,	TRANSACTION DUMP TRACE TYPE	*
TS=(COLD,8,8),	COLD, EIGHT BUFFERS & STRINGS	*
TSMGSET=20,	20 MESSAGE SET ENTRIES	*
TST=NO,	NO TEMP STORAGE TABLE INCLUDED	*
USERTR=ON,	ALLOW USER TO SET MASTER TRACE	*
USRDELAY=30,	USER TABLE TIME OUT	*
VTAM=YES,	VTAM ACCESS METHOD	*
VTPREFIX=C,	COMMON CLIENT TERM NAME PREFIX	*
WEBDELAY=(5,60),	CWS NETWORK DELAY	*
WRKAREA=512,	COMMON WORK AREA OF THE CSA	*
XAPPC=NO,	APPC SECURITY	*
XCMD=NO,	COMMAND SECURITY	*
XDCT=NO,	DCT SECURITY	*
XFCT=NO,	FCT SECURITY	*
XJCT=NO,	JCT SECURITY	*
XLT=SP,	SUPPLIED WITH Z/VSE	*
XPCT=NO,	PCT SECURITY	*
XPPT=NO,	PPT SECURITY	*
XPSB=NO,	PSB SECURITY	*
XRF=NO,	NO XRF SUPPORT INCLUDED	*

Figure 2-13 DFHSIT for second CICS TS (Part 3 of 4)

Figure 2-14 shows the DFHSIT for the second CICS TS.

XTRAN=YES,	XACTION ATTACH SECURITY	*
XTST=NO,	TST SECURITY	*
XUSER=NO,	SURROGATE USER	*
DUMMY=DUMMY	TO END MACRO	
END DFHSITBA		

Figure 2-14 DFHSIT for second CICS TS (Part 4 of 4)

2.3.5 Customizing the DFHPLT tables

This section describes customizing the DFHPLT tables.

Customizing PLTPI

The sequence of events during initialization changed. CICS initialization processing features three phases and program list table (PLT) execution includes two phases.

These two phases are separated by including the following entry:

```
DFHPLT TYPE=ENTRY,PROGRAM=DFHDELIM
```

Note: Programs can be written to execute during the second and third stages of initialization, but not during the first.

Programs that are listed before the PROGRAM=DFHDELIM entry are executed during the second stage of initialization. The purpose of this stage is to enable user exit programs that are needed during recovery. The user exit program should be defined in an RDO group in the CICS startup GRPLIST.

Programs that are listed after the PROGRAM=DFHDELIM are executed during the third stage of initialization. If these programs are used to enable user exits, the user exit program must also be defined in an RDO group, or it must be autoinstalled.

This phase corresponds to the single-phase PLTPI processing of previous releases and does not require program resource definition.

Figure 2-15 shows the DFHPLTP2 source that we used.

```
// EXEC ASMA90,SIZE=(ASMA90,64K),PARM='EXIT(LIBEXIT(EDECKXIT)),SIZE(MAXC
-200K,ABOVE)'
*****
*
* 5686-066 (C) COPYRIGHT IBM CORP. 1984, 1998
*
*****
      PUNCH ' CATALOG DFHPLTP2.OBJ REP=YES'
      DFHPLT TYPE=INITIAL,SUFFIX=P2
      DFHPLT TYPE=ENTRY,PROGRAM=DFHDELIM
      SPACE 3
*-----*
*
*          VSE/ESA ENTRIES FOR PLTP2 FOLLOW HERE
*
*-----*
*          COPY IESZPLP2          COPY VSE/ESA ENTRIES
*-----*
*          LOCAL ENTRIES SHOULD BE MADE AFTER THIS POINT
*-----*
*          FOLLOWING ENTRY WILL STARTUP CICS EXPLORER SERVER TRANSACTION
*          REMOVE ASTERISK TO ACTIVATE. BE SURE YOU RAN CEXPLCSD
*          SUCESSFULLY.
*-----*
*          DFHPLT TYPE=ENTRY,PROGRAM=EYU9NXRM
      SPACE 3
      DFHPLT TYPE=FINAL
      END
```

Figure 2-15 DFHPLTP2 Source: Initialization

For more information, see 7.2.2, “Program list table programs” on page 184.

Customizing PLTSD

A shutdown includes two phases. These phases are separated by including the following entry:

```
DFHPLT TYPE=ENTRY,PROGRAM=DFHDELIM
```

Programs that are listed before the PROGRAM=DFHDELIM entry are executed during the first quiesce stage of shutdown.

Programs that are listed after the PROGRAM=DFHDELIM entry are executed during the second quiesce stage of shutdown. The second quiesce stage does not require program resource definitions because they are autoinstalled by CICS.

Figure 2-16 shows the DFHPLTS2 source that we used.

```
// EXEC ASMA90,SIZE=(ASMA90,64K),PARM='EXIT(LIBEXIT(EDECKXIT)),SIZE(MAXC
-200K,ABOVE)'
*****
*
* 5686-028 (C) COPYRIGHT IBM CORP. 1984, 1992
*
*****
PUNCH ' CATALOG DFHPLTS2.OBJ REP=YES'
DFHPLT TYPE=INITIAL,SUFFIX=S2
SPACE 3

*-----*
*
* VSE/ESA ENTRIES FOR PLTS2 FOLLOW HERE
*
*-----*
COPY IESZPLS2 COPY VSE/ESA ENTRIES
*-----*
* LOCAL ENTRIES TO BE EXECUTED DURING FIRST QUIESCE PHASE
* SHOULD BE MADE AFTER THIS POINT
*-----*
*
* FOLLOWING ENTRY FOR PROGRAM DFHOSTAT WILL CAUSE PRINTING
* OF STATISTICS DURING SHUT DOWN OF CICS TS. PLEASE ACTIVATE
* THE ENTRY ONLY AFTER HAVING
* 1. DEFINED FOLLOWING PROGRAMS IN THE CSD FILE:
* DFHOSTAT, DFH$STAS AND DFH$STCN.
* ALL THREE PROGRAMS HAVE TO BE DEFINED WITH EXECKEY=CICS.
* 2. COMPILE PROGRAMS DFHOSTAT AND DFHOSTM AND LINKEDIT THEM
* INTO PRD2.CONFIG.
* IF YOU DON'T HAVE A COBOL/VSE COMPILER INSTALLED, YOU MAY
* USE SKELETON DFHOSTAT IN ICCF LIBRARY 59 TO ESTABLISH
* THE PROGRAM WITHOUT COMPILING IT.
* ENTRY FOR PROGRAM DFH$SDAP WILL SHUTDOWN ALL APPLICATIONS IN A
* PROPER WAY. REFER TO ICCF MEMBER DFH$SDAP IN LIBRARY 59 FOR
* DETAILS ON PROGRAM DEFINITION.
*-----*
* DFHPLT TYPE=ENTRY,PROGRAM=DFHOSTAT
* DFHPLT TYPE=ENTRY,PROGRAM=DFH$SDAP SHUTDOWN APPLICATIONS
* DFHPLT TYPE=ENTRY,PROGRAM=DFHDELIM
*-----*
* LOCAL ENTRIES TO BE EXECUTED DURING SECOND QUIESCE PHASE
* SHOULD BE MADE AFTER THIS POINT
*-----*
DFHPLT TYPE=FINAL
END
```

Figure 2-16 DFHPLTS2 source: Shutdown

2.3.6 Tailoring the CICS start-up job stream

We used the job SKCICS2 that is supplied in ICCF library 59 to start our second CICS system in partition F8. Figure 2-17 shows the console log of CICS TS startup.

```
F8 0001 1Q4DI  JOB CICS3 35429 FINISHED PROCESSING IN PARTITION F8
F8 0001 1Q34I  F8 WAITING FOR WORK
F8 0001 1Q47I  F8 CICS2 35430 FROM (SYSA) , TIME=11:57:12 , TKN=00001310
F8 0008 // JOB CICS2          STARTUP OF SECOND CICS WITHOUT ICCF
      DATE 11/17/2016, CLOCK 11/57/12
F8 0001 1QFOI  DATA FILE 096% FULL - QUEUE FILE 034% FULL
F8 0008 LOG
F8 0008 ID (PARAMETERS SUPPRESSED)
F8 0008 NOLOG
F8 0008 *   WAITING FOR VTAM TO COME UP
F8 0008 *   WAITING FOR TCP/IP TO COME UP
F8 0008 * // EXEC REXX=IESWAITR,PARM='TCPIP00'
F8 0008 1T20I  SYS020 HAS BEEN ASSIGNED TO X'FEE' (TEMP)
F8 0008 * // ASSGN SYS023,DISK,VOL=DOSRES,SHR  IF JOURNALLING
F8 0008 DFHPA1101 PRODCICS DFHSITC2 IS BEING LOADED.
F8 0008 DFHPA1108 PRODCICS DFHSITC2 HAS BEEN LOADED. (GENERATED AT: MM/DD=
      11/03 HH:MM= 13:57).
F8 0008 DFHPA1100 PRODCICS OVERRIDE PARAMETERS FROM JCL EXEC STATEMENT:
F8 0008 DFHPA1927 PRODCICS APPLID=PRODCICS,START=AUTO,EDSALIM=450M,SI
F8 0008 DFHPA1102 PRODCICS OVERRIDE PARAMETERS FROM SYSIPT:
F8 0008 DFHPA1927 PRODCICS SIT=C2,STATRCD=OFF,NEWSIT=YES,
F8 0008 DFHPA1927 PRODCICS PRVMOD=(DFHALP,DFHAMP,DFHAPCR,DFHCCNV,DFHCHS,
      DFHCMP,DFHCRNP,
F8 0008 DFHPA1927 PRODCICS DFHCRQ,DFHCRR,DFHCRS,DFHCRSP,DFHCRT,DFHCXPA,
      DFHCXPB,DFHDCP,DFHDFCDU,
...
F8 0008 DFHPA1927 PRODCICS DFHXSWM,DFHZCUT,DFHZCXR,DFHZGAI,DFHZGBM,DFHZGCA,
      DFHZGCC,DFHZGCN,
F8 0008 DFHPA1927 PRODCICS DFHZGPR,DFHTRAQ)

F8 0008 DFHPA1103 PRODCICS END OF FILE ON SYSIPT.
F8 0008 DFHTR0103 TRACE TABLE SIZE IS 4096K
F8 0008 DFHSM0122I PRODCICS Limit of DSA storage below 16MB is 5,120K.
F8 0008 DFHSM0123I PRODCICS Limit of DSA storage above 16MB is 450M.
F8 0008 DFHSM0115I PRODCICS Storage protection is active.
F8 0159 DFHDM0101I PRODCICS CICS is initializing.
F8 0160 DFHXS1100I PRODCICS Security initialization has started.
F8 0160 DFHWB0109I PRODCICS Web domain initialization has started.
F8 0160 DFHSO0100I PRODCICS Sockets domain initialization has started.
F8 0160 DFHDH0100I PRODCICS Document domain initialization has started.
F8 0160 DFHSI1500 PRODCICS CICS startup is in progress for CICS Transaction
      Server Version 2.1.0
F8 0160 DFHSI1501I PRODCICS Loading CICS nucleus.
F8 0160 DFHXS1105 PRODCICS Resource profiles for class TCICSTRN have been
      built.
F8 0160 DFHXS1103I PRODCICS Default security for userid CICSUSER has been
      established.
F8 0160 DFHXS1101I PRODCICS Security initialization has ended.
F8 0160 DFHWB0110I PRODCICS Web domain initialization has ended.
F8 0160 DFHMN0105I PRODCICS Using default Monitoring Control Table.
F8 0160 DFHMN0110I PRODCICS CICS Monitoring is inactive.
```

Figure 2-17 Output from CICS startup (Part 1 of 2)

Figure 2-18 shows the console log of CICS TS startup.

```
F8 0160 DFHLD0107I PRODCICS
      DFHLDDMI is unable to locate module EZASOH00 in the SVA. LIBDEF
      search chain version of module will be used.
F8 0160 DFHHD0101I PRODCICS Document domain initialization has ended.
F8 0160 DFHSI1502I PRODCICS CICS startup is Warm.
F8 0160 DFHSI1503I PRODCICS Terminal data sets are being opened.
F8 0160 DFHDU0304I PRODCICS Transaction Dump Data set DFHDMPA opened.
F8 0160 DFHLD0107I PRODCICS
      DFHLDL1 is unable to locate module DFHAPATT in the SVA. LIBDEF
      search chain version of module will be used.
F8 0160 DFHCP0101I PRODCICS CPI initialization has started.
F8 0160 DFHPR0104I PRODCICS Partner resource manager initialization has
      started.
F8 0160 DFHLD0107I PRODCICS
      DFHLDL1 is unable to locate module DFHTDRP in the SVA. LIBDEF
      search chain version of module will be used.
F8 0160 DFHTS0100I PRODCICS Temporary Storage initialization has started.
F8 0160 DFHFC0100I PRODCICS File Control initialization has started.
F8 0160 DFHFC0101I PRODCICS File Control initialization has ended.
F8 0160 DFHTD0100I PRODCICS Transient Data initialization has started.
F8 0160 DFHTD0101I PRODCICS Transient Data initialization has ended.
F8 0160 DFHTS0101I PRODCICS Temporary Storage initialization has ended.
F8 0160 DFHAI0101I PRODCICS AITM initialization has started.
F8 0160 DFHCP0102I PRODCICS CPI initialization has ended.
F8 0160 DFHPR0105I PRODCICS Partner resource manager initialization has ended.
F8 0160 DFHAI0102I PRODCICS AITM initialization has ended.
F8 0160 DFHAPI203I PRODCICS Language Environment for VSE/ESA is being
      initialized.
F8 0160 CEE3550I LE/VSE C/VSE Run-Time Initialized
F8 0160 CEE3551I LE/VSE COBOL Run-Time Initialized
F8 0160 CEE3552I LE/VSE PL/I Run-Time Initialized
F8 0160 DFHS00120 PRODCICS 1 TCBS are initialized for SSL processing.
F8 0160 DFHS00101I PRODCICS Sockets domain initialization has ended.
F8 0160 DFHWB1007 PRODCICS Initializing CICS Web environment.
F8 0160 DFHWB1008 PRODCICS CICS Web environment initialization is complete.
F8 0160 DFHSI8430I PRODCICS About to link to PLT programs during the third
      stage of initialization.
F8 0178 BSD100I IPNRBSDC 02.01.04 20150629 17.40 0390B000 00B2
F8 0178 DFHS00117 PRODCICS
      Unable to determine the TCP/IP host name. Language Environment return
      code X'0000045D', reason code X'000000FF'. TCP/IP services are
      unavailable.
F8 0178 DFHS00102 PRODCICS
      11/17/16 11:57:14 PRODCICS A Language Environment Callable Service
      error (code X'0000') has occurred on receipt of a severe TCP/IP
      return code; the TCIPSERVICE CMCIT on port 27283 at IP address
      ANY will be closed.
F8 0160 DFHSI8434I PRODCICS Control returned from PLT programs during the
      third stage of initialization.
F8 0160 DFHSI1517 PRODCICS Control is being given to CICS.
```

Figure 2-18 Output from CICS startup (Part 2 of 2)

System initialization parameter descriptions

The following CICS system initialization parameters cannot be defined in the DFHSIT macro:

- ▶ CDSASZE={0K|number}
- ▶ CHKSTRM={CURRENT|NONE}
- ▶ CHKSTSK={ALL|CURRENT|NONE}
- ▶ ECDSASZE={0K|number}
- ▶ ERDSASZE={0K|number}
- ▶ ESDSASZE={0K|number}
- ▶ EUDSASZE={0K|number}
- ▶ JSTATUS=RESET
- ▶ NEWSIT={YES|NO}
- ▶ PRVMOD={name|(name,name...name)}
- ▶ RDSASZE={0K|number}
- ▶ SDSASZE={0K|number}
- ▶ SIT=xx
- ▶ SPCTRxx={{1[,2][,3]}|ALL|OFF}
- ▶ START=LOGTERM
- ▶ START=(option,ALL)
- ▶ STNTRxx={{1[,2][,3]}|ALL|OFF}
- ▶ UDSASZE={0K|number}

These parameters can be supplied only at CICS startup time by using any of the following methods:

- ▶ The PARM parameter of the EXEC DFHSIP statement.
- ▶ The SYSIPT data set defined in the startup job stream.
- ▶ Through the system operator's console.

For more information about coding system initialization parameters in PARM, SYSIPT, or at the console, see *CICS Transaction Server for VSE/ESA System Definition Guide*.

2.3.7 Autoinstall of resources

Autoinstall can be used for the following items:

- ▶ IBM VTAM® terminals
- ▶ APPC (LU6.2) connections
- ▶ Programs
- ▶ Mapsets
- ▶ Partitionsets

By using autoinstall, you do not need to define and install every resource that you intend to use. Instead, CICS dynamically creates and installs a definition for you when a resource is requested. You must provide at least one model resource definition for each type of resource to be autoinstalled.

When a resource is requested that does not have an installed definition, CICS creates a definition based on what you specified in the model.

Program autoinstall

Autoinstall for programs, mapsets, and partitionsets brings the following benefits:

- ▶ Reduced system administration costs
- ▶ Saving in virtual storage within the CICS address space
- ▶ Faster COLD restart
- ▶ Possible improvements on WARM and EMER restart

Complete the following steps to program autoinstall:

1. Decide whether your programs are eligible for autoinstall. The following programs cannot be autoinstalled:
 - Program autoinstall control program
 - Terminal autoinstall control program
 - Connection autoinstall control program
2. Decide which programs to autoinstall and what to do with existing program and mapset definitions in the CSD. You can use a mixture of RDO and autoinstall.
3. Decide whether you want to use program autoinstall with or without cataloging. If you decide to have autoinstalled definitions recorded in the CICS catalog, specify this configuration by using the PGAICTLG SI, SET SYSTEM SPI, or CEMT parameter.
4. Enable autoinstall for programs. The following system initialization parameters are related to program autoinstall:
 - PGAICTLG: Whether an autoinstalled program definition is cataloged.
 - PGAIPGM: Whether the program autoinstall function is active or inactive.
 - PGAEXIT: The name of the program autoinstall exit.

You can also enable autoinstall for programs by using the EXEC CICS or CEMT INQUIRE|SET SYSTEM command.

Figure 2-19 shows part of the DFHSITC2 table.

...	PARMERR=INTERACT,	INTERACT IF OF PARAMETERS WRONG*
	PGAICTLG=ALL,	UPDATE AUTOINSTALL PGM DEFINING*
	PGAEXIT=DFHPGADX,	PGM AUTOINSTALL EXIT *
	PGAIPGM=ACTIVE,	PGM AUTOINSTALL ACTIVE *
	PGCHAIN=X/,	BMS CHAINING COMMAND *
...		

Figure 2-19 Sample of DFHSIT with program autoinstall parameters

5. Define the CSPL transient data queue if you want to log messages that are associated with autoinstall for programs. In copybook IESZDCT, we include the definition that is shown in Figure 2-20.

CSPL	DFHDCT TYPE=INDIRECT,	MESSAGES FROM PROGRAM MANAGER *
	DESTID=CSPL,	DEFINE THE DESTINATION ID *
	INDEST=IESL	SEND IT THROUGH THE VSE/ESA QUEUE

Figure 2-20 CSPL queue in the IESZDCT copybook

6. Create your model program definitions.

A model definition provides CICS with one definition that can be used for all programs with the same properties. If you do not want to use your own definitions, you can use the CICS-supplied model definitions in group DFHPGAIP. DFHPGAPG for programs (see Figure 2-21).

OBJECT CHARACTERISTICS		CICS RELEASE = 0420
CEDA View PROGram(DFHPGAPG)		
PROGram	: DFHPGAPG	
Group	: DFHPGAIP	
DEscription	: DEFAULT PROGRAM FOR PROGRAM AUTOINSTALL	
Language	:	CObol Assembler C Pli
RELoad	: No	No Yes
RESident	: No	No Yes
USAge	: Normal	Normal Transient
USEsvacopy	: No	No Yes
Status	: Enabled	Enabled Disabled
RSI	: 00	0-24 Public
Cedf	: Yes	Yes No
DAtalocation	: Below	Below Any
EXECKey	: User	User Cics
REMOTE ATTRIBUTES		
REMOTESystem	:	
REMOTENAME	:	
+ Transid	:	
		SYSID=CIC1 APPLID=DBDCCICS
PF 1 HELP 2 COM 3 END 6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL		

Figure 2-21 RDO VIEW PROGRAM(DFHPGAPG): CEDA transaction

DFHPGAMP for mapsets is shown in Figure 2-22.

OBJECT CHARACTERISTICS		CICS RELEASE = 0420
CEDA View Mapset(DFHPGAMP)		
Mapset	: DFHPGAMP	
Group	: DFHPGAIP	
DEscription	: DEFAULT MAPSET FOR PROGRAM AUTOINSTALL	
RESident	: No	No Yes
USAge	: Normal	Normal Transient
USEsvacopy	: No	No Yes
Status	: Enabled	Enabled Disabled
RSI	: 00	0-24 Public
		SYSID=CIC1 APPLID=DBDCCICS
PF 1 HELP 2 COM 3 END 6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL		

Figure 2-22 RDO VIEW MAPSET(DFHPGAMP): CEDA transaction

DFHPGAPT for partitionsets is shown in Figure 2-23.

OBJECT CHARACTERISTICS		CICS RELEASE = 0420
CEDA View PARTITIONset(DFHPGAPT)		
PARTITIONset	: DFHPGAPT	
Group	: DFHPGAIP	
Description	: DEFAULT PARTITIONSET FOR PROGRAM AUTOINSTALL	
REsident	: No	No Yes
USAge	: Normal	Normal Transient
USEsvacopy	: No	No Yes
Status	: Enabled	Enabled Disabled
RS1	: 00	0-24 Public
SYSID=CIC1 APPLID=DBDCCICS		
PF 1 HELP 2 COM 3 END 6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL		

Figure 2-23 RDO VIEW PARTITIONSET(DFHPGAPT): CEDA transaction

7. Design and write an autoinstall control program.

The purpose of the autoinstall control program is to provide CICS with the extra information it needs to complete an autoinstall request, such as the autoinstall model name.

You can write your autoinstall program in any language that is supported by CICS, with full access to the CICS application programming interface. The following sample programs are supplied by CICS:

- ▶ DFHPGADX, in assembler language
- ▶ DFHPGAHX, in C language
- ▶ DFHPGALX, in PL/I language
- ▶ DFHPGAOX, in COBOL language

The source for these programs and the executable form of the assembler version are supplied in the VSE/ESA sublibrary PRD1.BASE.

VTAM autoinstall

Complete the following steps to set up VTAM autoinstall:

Note: For more information, see *CICS Transaction Server for VSE/ESA Resource Definition Guide*.

1. Decide whether your terminals are eligible for autoinstall. The following terminals cannot be autoinstalled:
 - Pipeline terminals
 - Automatic teller machines (3614 and 3624)
 - Non-VTAM resources
 - VTAM logical unit type 6.1 ISC and MRO sessions

2. Decide whether to use autoinstall.

You are likely to benefit from autoinstall if any of the following considerations apply to your system:

- A significant number of VTAM terminals
- Frequent changes to your network
- Many VTAM terminals logged off much of the time
- Many VTAM terminals using other applications much of the time
- Many VTAM terminals that need access to multiple, but unconnected
- CICS systems

3. Decide which devices to autoinstall.

This decision depends on how you use your VTAM terminals. An autoinstall logon is slower than a logon to a terminal that is individually defined to CICS; therefore, if you switch continually between applications and must log on to CICS frequently, you might require individual definitions for some terminals. Consider the following special issues:

- Automatic transactions initiation
- The TCT user area (TCTUA)
- The terminal list table (TLT)
- Transaction routing
- Autoinstall and output-only devices

For more information, see Chapter 9 of *CICS Transaction Server for VSE/ESA Resource Definition Guide*.

4. Create your TYPETERM and model TERMINAL definitions.

Define an autoinstall model for each type of terminal to be autoinstalled. The number of definitions should be kept to a minimum so that the autoinstall control program can be simple.

CICS supplies some TERMINAL and TYPETERM definitions in CSD group DFHTERM and DFHTYPE. For more information, see Appendix B of *CICS Transaction Server for VSE/ESA Resource Definition Guide*.

Note: The most important parameter in terminal definition is AUTINSTMODEL.

5. Redefine DFHZCQ.

The Terminal control installation interface program (DFHZCQ) should be redefined as RESIDENT(YES). If you use the DFHZCQ from the CICS-supplied group DFHSPI, it is defined with RESIDENT(YES) parameter, as shown in Figure 2-24.

OBJECT CHARACTERISTICS		CICS RELEASE = 0420
CEDA View PROGram(DFHZCQ)		
PROGram	: DFHZCQ	
Group	: DFHSPI	
DEscription	:	
Language	: Assembler	CObol Assembler C Pl i
RELoad	: No	No Yes
RESident	: Yes	No Yes
USAge	: Normal	Normal Transient
USEsvacopy	: No	No Yes
Status	: Enabled	Enabled Disabled
RSI	: 00	0-24 Public
Cedf	: No	Yes No
DAtallocation	: Any	Below Any
EXECKey	: Cics	User Cics
REMOTE ATTRIBUTES		
REMOTESystem	:	
REMOTENAME	:	
+ Transid	:	
SYSID=CIC1 APPLID=DBDCCICS		
PF 1 HELP 2 COM 3 END 6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL		

Figure 2-24 RDO VIEW PROGRAM(DFHZCQ): CEDA transaction

6. Ensure that your VTAM LOGMODE table entries are correct.

When a VTAM terminal logs on to CICS, the VTAM LOGMODE is used by autoinstall to find matching CICS terminal type definitions. The following items must correspond to each other:

- Physical terminal characteristics
- VTAM session parameters
- CICS terminal type definition

7. Design and write an autoinstall control program.

The autoinstall control program is invoked by CICS when a valid request for a TCT entry to be autoinstalled is received and when an autoinstalled TCT entry is deleted. The following sample programs are supplied:

- DFHZATDX, in assembler language
- DFHZATDY, in assembler language
- IESZATDX, in assembler language
- DFHZCTDX, in COBOL language
- DFHZDTDX, in C language
- DFHZPTDX, in PL/I language

The source for the IESZATDX program is supplied in VSE/ICCF library 59 and the others programs are supplied in the VSE/ESA sublibrary PRD1.BASE.

You can have only one autoinstall control program active at one time for terminals and connections. We suggest the use of IESZATDX. The DFHZATDY and IESZATDX programs provide the same function for terminal autoinstall as DFHZATDX and functions to autoinstall APPC connections. IESZATDX also provides support for LU6.2 APPC sessions.

8. Enable terminal autoinstall.

The following system initialization parameters relate to terminal autoinstall:

- AIEXIT: The name of the autoinstall program exit.
- AIQMAX: The maximum number of terminals that can be queued concurrently.
- AILDELAY: The time interval that elapses after an autoinstall terminal logs off before its TCTTE is deleted.
- AIRDELAY: The time interval that elapses after emergency restart before terminal entries are deleted if they are not in session.

You can also enable autoinstall for terminals by using EXEC CICS or CEMT INQUIRE|SET AUTOINSTALL commands.

9. Define the CADL transient data queue.

Define the CADL transient data queue if you want to record each installation and each deletion of TCT entries. Message DFHZC6987 is useful for indicating which model came closest to the model that was chosen when a null list of models is passed to the autoinstall control program.

In copybook IESZDCT, we include the definition that is shown in Figure 2-25.

CADL	DFHDCT TYPE=INDIRECT,	CEDA COMMAND LOGGING COMES HERE	*
	DESTID=CADL,	DEFINE THE DESTINATION ID	*
	INDEST=IESL	SEND IT THROUGH THE VSE/ESA QUEUE	

Figure 2-25 CADL queue in the IESZDCT copybook

APPC autoinstall

Autoinstall for APPC connections brings the following benefits:

- ▶ The COLD, WARM and EMER restarts are faster.
- ▶ There are fewer CSD definitions to manage.
- ▶ Less storage is taken up by unused definitions.

Complete the following steps to configure APPC autoinstall:

1. Decide whether to use autoinstall for connections. Consider the following possible restrictions:
 - If security for connections is used.
 - If persistent sessions support is used.
 - If you must recover an autoinstalled connection after WARM or EMER restarts.
2. Decide which sessions to autoinstall. You can use autoinstall for CICS-to-CICS connections, but it is intended primarily for workstations.
3. Create your model connection definitions.

Any installed connection definition can be used as a “template” for an autoinstalled connection, but you should use an installed connection definition that is not in use because the definition is locked while CICS copies it.

4. Design and write an autoinstall control program.

CICS supplies a sample control program (DFHZATDY) for connections autoinstall and VSE/ESA supplies a sample control program (IESZATDX).

5. Enable autoinstall for connections.

Autoinstall for connections are enabled when you enable autoinstall for terminals (for more information, see section “VTAM autoinstall” on page 43). If you want to disable autoinstall for connections, set the model connection out of service by using the **CEMT** or **EXEC CICS SET CONNECTION OUTSERVICE** command.

2.4 Migration from CICS/VSE 2.3

This section describes other tailoring tasks that are applicable when migrating from CICS/VSE 2.3.

2.4.1 Migrating the DFHPCT and DFHPPT tables

A number of functions are withdrawn in CICS TS, especially the ability to define transaction and program resource definitions using suffixed PCTs and PPTs.

You must define the following items in the CSD:

- ▶ Transactions
- ▶ Transaction profiles
- ▶ Programs
- ▶ Map sets
- ▶ Partition sets for basic mapping support

Migration steps

Complete the following steps to perform the migration:

1. Edit your DFHPPT source.
2. Assemble and link-edit the edited source.
3. Process the link-edited table by using the **MIGRATE** command of the DFHCSDUP utility.
4. Check the output that is produced by using the **MIGRATE** command (you might have to regroup and rename the migrated definitions).
5. Initialize CICS with the migrated definitions.

Editing your DFHPCT/DFHPPT source

You must reassemble your DFHPCT/DFHPPT source by using CICS Transaction Server macro libraries. The DFHPCT and DFHPPT macros are available only as input to the migration utility program in DFHCSDUP. Remove all IBM-supplied CICS and VSE Interactive Interface entries before assembling.

If your source is large, you might find it more convenient to assemble small segments of the source individually because each resource definition on the CSD must belong to a CSD group. It is better to create many small groups for your files rather than one large group. Groups should be limited to 100 resource definitions.

The definitions within a group often have something in common. For example, keep all the resource definitions that belong to one application in one group; for example, one CSD group for payroll application and one CSD group for inventory application. You can specify the names to be given to groups of definitions that are generated from your DFHPCT macros or DFHPPT macros by adding TYPE and GROUP macro statements (see Example 2-1).

Example 2-1 DFHPCT macros or DFHPPT macros

```
DFHPCT TYPE=GROUP, GROUP=GROUPNAME
or
DFHPPT TYPE=GROUP, GROUP=GROUPNAME
```

All definitions that follow a particular TYPE=GROUP macro statement are migrated into the named group in the CSD file. A new TYPE=GROUP statement overrides all previous statements.

Figure 2-26 shows the DFHPPTOL source that we used.

```
// EXEC ASMA90,SIZE=(ASMA90,64K),PARM='EXIT(LIBEXIT(EDECKXIT)),SIZE(MAXC
-200K,ABOVE)'
*****
*                                     *
*   5686-028 (C) COPYRIGHT IBM CORP. 1984, 1990   *
*                                     *
*****
      TITLE 'DFHPPTOL -- OLD PPT MIGRATION'
      PUNCH ' CATALOG DFHPPTOL.OBJ REP=YES'
      DFHPPT TYPE=INITIAL,SUFFIX=OL
*
      DFHPPT TYPE=GROUP, GROUP=MYAPPL
*
*-----*
      DFHPPT TYPE=ENTRY, PROGRAM=MYAPINIT, RES=YES, PGMLANG=ASSEMBLER
      DFHPPT TYPE=ENTRY, PROGRAM=MYAPPROG, RES=YES, PGMLANG=ASSEMBLER
      DFHPPT TYPE=ENTRY, PROGRAM=MYAPSTOP, RES=YES, PGMLANG=ASSEMBLER
*-----*
      DFHPPT TYPE=FINAL
      END
/*
```

Figure 2-26 DFHPPT source

Most DFHPCT and DFHPPT macros have CSD equivalents, with the following exceptions:

- ▶ DFHPCT macro operands without RDO equivalents:
 - ANTICPG
 - DTB=NO
 - FDUMP
 - PRMSIZE

► DFHPPT macro operands without RDO equivalents:

- FN
- LENABLE
- RES=PGOUT
- RES=FIX
- RES=ALIGN
- RSL

Assembling and link-editing the source

You must assemble your table by using the CICS TS for z/VSE 2.1 macro library (PRD1.BASE) and link-edit it into your CICS load library. If you receive a return code that is greater than 4, remove the cause of the error and reassemble.

Using the DFHCSDUP utility

To convert the table entries into resource definitions, use the **MIGRATE** command of the DFHCSDUP offline utility program (see Figure 2-27).

```
// LIBDEF *,SEARCH=(PRD2.CONFIG,PRD1.BASE,PRD2.SCEEBASE)
// EXEC DFHCSDUP,SIZE=600K          INIT AND LOAD CICS
//   MIGRATE TABLE(DFHPPTOL) TOGROUP(MYAPPL)
/*
```

Figure 2-27 Using the MIGRATE command to transfer a PPT to the CSD

Checking the output from MIGRATE

The use of the **MIGRATE** process might produce some messages. Read these warning messages carefully. Figure 2-28 shows the output from the **MIGRATE** command.

```
// EXEC DFHCSDUP,SIZE=600K          INIT AND LOAD CICS
1S54I  PHASE DFHCSDUP IS TO BE FETCHED FROM PRD1.BASE

MIGRATE TABLE(DFHPPTOL) TOGROUP(MYAPPL)

DFHCA5120 I PRIMARY CSD OPENED;  FILENAME: DFHCSD
DFHCA5144 I MIGRATION OF TABLE DFHPPTOL IN PROGRESS. DEFAULT GROUP IS MYAPPL
DFHCA5143 I GROUP MYAPPL  CREATED.
DFHCA5159 I PROGRAM MYAPINIT DEFINED IN GROUP MYAPPL
DFHCA5159 I PROGRAM MYAPPROG DEFINED IN GROUP MYAPPL
DFHCA5159 I PROGRAM MYAPSTOP DEFINED IN GROUP MYAPPL
DFHCA5140 I TOTAL PROGRAM      DEFINITIONS CREATED:      3
DFHCA5101 I MIGRATE COMMAND EXECUTED SUCCESSFULLY.
DFHCA5123 I PRIMARY CSD CLOSED;  FILENAME: DFHCSD

DFHCA5107 I COMMANDS EXECUTED SUCCESSFULLY: 1      COMMANDS GIVING WARNING(S): 0
DFHCA5108 I COMMANDS NOT EXECUTED AFTER ERROR(S): 0
DFHCA5109 I END OF DFHCSDUP UTILITY JOB. HIGHEST RETURN CODE WAS: 0
```

Figure 2-28 Output from PPT MIGRATE command

From the output listing, you can see the names of groups, transactions, profiles, and programs that are defined in the CSD.

Initializing CICS after migration

Every resource definition is a member of a group. To install these groups at initialization, complete the following steps:

1. Use DFHCSDUP or resource definition online (RDO) to ADD each group to a list.

Figure 2-29 shows the sample JCL for the **ADD** command in the DFHCSDUP offline utility.

```
// LIBDEF *,SEARCH=(PRD2.CONFIG,PRD1.BASE,PRD2.SCEEBASE)
// EXEC DFHCSDUP,SIZE=600K          INIT AND LOAD CICS
    ADD GROUP(MYAPPL) LIST(VSELST2)
/*
```

Figure 2-29 ADD command from DFHCSDUP

Figure 2-30 shows the sample **ADD** command in the CEDA transaction.

```
OVERTYPE TO MODIFY
CEDA Add
  Group      ==> MYAPPL
  List       ==> VSELST2
  Before     ==>
  After      ==>

                                SYSID=CIC1 APPLID=DBDCCICS
ADD SUCCESSFUL                  TIME: 14.17.00 DATE: 16.323
PF 1 HELP      3 END          6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

Figure 2-30 RDO ADD command: CEDA transaction

2. Use the GRPLIST parameter in the DFHSIT table or in the PARM field in CICS initialization to tell CICS which groups are to be installed. Figure 2-31 shows part of the CICS SIT with the GRPLIST parameter.

ENCRYPTION=NORMAL,	SSL ENCRYPTION	*
EODI=EO,	END OF DATA SEQUENTIAL DEVICES	*
ESMEXITS=INSTLN,	INSTLN PARAMETER IN RACROUTE	*
FCT=NO,	FOR SECOND CICS	*
FEPI=NO,	NO FRONT END PROGRAMMING I/F	*
GMTEXT='z/VSE CICS2',	GOOD MORNING MESSAGE TEXT	*
GMTRAN=IEGM,	LOGON TRANSACTION ID	*
GNTRAN=IEGT,	TIME OUT TRANSACTION	*
GRPLIST=VSELST2,	AUTOINST. TERMINALS. AND MRO	*
ICP=COLD,	INTERVAL CONTROL PGM	*
ICV=1000,	INTERVAL CONTROL EXIT TIME-MS	*
ICVR=5000,	RUNAWAY TASK TIME	*
ICVTS=100,	TERMINAL SCAN DELAY	*
INTTR=ON,	INTERNAL TRACE	*
IRCSTRT=NO,	START IRC DURING INITIALIZATION*	
ISC=YES,	INTERSYSTEM COMMUNICATION	*

Figure 2-31 Sample of DFHSIT with GRPLIST parameter

3. Initialize the CICS partition.

Alternatively, you can wait until CICS initializes and then use RDO to INSTALL the groups that you want CICS to use, as shown in Figure 2-32.

```
OVERTYPE TO MODIFY
CEDA Install
All
Connection ==>
Doctemplate ==>
File ==>
Lsrpool ==>
Mapset ==>
PARTitionset ==>
PARTner ==>
PROFile ==>
PROGram ==>
TCpipservice ==>
TERminal ==>
TRANClass ==>
TRANSaction ==>
TYpeterm ==>
Group ==> MYAPPL

                                SYSID=CIC1 APPLID=DBDCCICS
INSTALL SUCCESSFUL              TIME: 14.24.51 DATE: 16.323
PF 1 HELP      3 END            6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
```

Figure 2-32 RDO INSTALL command - CEDA transaction

The installed resources are active if CICS is WARM or EMERgency restarted. Add the resources to the GRPLIST to ensure that they are activated after a COLD start.

To create a program definition in the CSD, you use the **DEFINE PROGRAM** command in the following places:

- Resource definition online (RDO): CEDA or CEDB transactions
- DFHCSDUP offline utility
- Autoinstall (see “Autoinstall of resources” on page 40)

Figure 2-33 shows the RDO Program Definition.

OVERTYPE TO MODIFY		CICS RELEASE =
0420		
CEDA DEFINE PROGRAM(MYAPPR2)		
PROGRAM	:	MYAPPR2
Group	:	MYAPPL
Description	==>	
Language	==>	Assembler CObol Assembler C Pl1
RELoad	==>	No Yes
RESident	==>	No Yes
USAge	==>	Normal Transient
USEsvacopy	==>	No Yes
Status	==>	Enabled Disabled
RS1	:	00 0-24 Public
Cedf	==>	Yes No
DAtalocation	==>	Below Any
EXEKey	==>	User Cics
REMOTE ATTRIBUTES		
REMOTESystem	==>	
REMOTENAME	==>	
+ Transid	==>	
SYSID=CIC1 APPLID=DBDCCICS		
DEFINE SUCCESSFUL TIME: 14.28.55 DATE: 16.323		
PF 1 HELP 2 COM 3 END 6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL		

Figure 2-33 RDO DEF PROGRAM: CEDA transaction

2.4.2 Customizing and migrating the DFHFCT table

You can define the following data sets by using RDO or file control table (FCT) macros:

- ▶ VSAM files
- ▶ Remote VSAM files
- ▶ Remote DAM files
- ▶ VSAM local shared resource (LSR) pools
- ▶ Shared data tables

Migrate these definitions to the CSD to take advantage of the benefits of online definition. Any future releases of CICS will support online definition for these resources only.

Whatever your choices, the result is an FCT built from one or both of the following entries:

- ▶ Entries that are defined by using DFHFCT macros
- ▶ Entries that are defined and installed by using RDO and the GRPLIST parameter

Entries of both kinds can coexist so that you can keep your existing tables and try the new function in parallel.

If you continue using your existing DFHFCT macro, you must remove the CSD entry from the FCT source statements and then reassemble the table. The CSD is defined by system initialization parameters. For more information, see *CICS Transaction Server for VSE/ESA System Definition Guide*, SC33-1652.

If you define the same file by using the RDO and macro methods, be aware that the macro definitions are installed during cold start first. Then, the RDO-defined file entry overrides the equivalent macro-defined file entry when they are installed (by using CEDA install or during group list install on a cold start).

If you use data tables, see “Shared data tables” on page 63.

Migration steps

To migrate at least some of your table entries to RDO, complete the following steps:

1. Edit your existing DFHFCT source and remove any definition for the CSD.
2. Assemble and link-edit the edited source.
3. Process the link-edited table by using the DFHCSDUP utility.
4. Check the output that is produced by the DFHCSDUP utility.
5. Use the **DFHCSDUP ADD** command to add the new groups to a list of groups on your CSD.
6. Remove from your macro source all the table entries that were migrated and reassemble it.
7. Start your system with the non-VSAM FCT (if any) and the list you created.

Editing your existing DFHFCT source

You must reassemble your existing DFHFCT source by using the CICS TS macro library. If your source is large, you might find it convenient to assemble small segments of the source individually because each resource definition on the CSD must belong to a CSD group. It is better to create many small groups for your files rather than one large group. Groups should be limited to 100 resource definitions.

The definitions within a group often have something in common. For example, keep all the resource definitions belonging to one application in one group (one CSD group for payroll application, one CSD group for inventory application) or according to the department or function of the users who are using them.

You can specify the names to be given to groups of definitions that are generated from your DFHFCT macros by adding the following line:

```
DFHFCT TYPE=GROUP, GROUP=GROUPNAME
```

All definitions that follow a particular TYPE=GROUP macro statement are migrated into the named group in the CSD file.

A new TYPE=GROUP statement defines a new group, as shown in Figure 2-34.

```
// EXEC ASMA90,SIZE=(ASMA90,64K),PARM='EXIT(LIBEXIT(EDECKXIT)),SIZE(MAXC
-200K,ABOVE)'
    TITLE 'DFHFCTOL -- OLD FCT MIGRATION'
    PUNCH ' CATALOG DFHFCTOL.OBJ REP=YES'
    DFHFCT TYPE=INITIAL,SUFFIX=OL
*
    DFHFCT TYPE=GROUP,GROUP=VSEFILES
    COPY IESZFCTO -- DATASET ENTRIES FOR Z/VSE - OTH PARTITION
*
*-----*
    DFHFCT TYPE=GROUP,GROUP=MYFILES
*
FILEA  DFHFCT TYPE=FILE,FILE=FILEA,
        ACCMETH=VSAM,
        RECFORM=(FIXED,BLOCKED),
        LSRPOOL=01,
        JID=02,
        JREQ=ALL,
        LOG=YES,
        SERVREQ=(UPDATE,ADD,BROWSE,DELETE),
        STRNO=3
*
FILEB  DFHFCT TYPE=FILE,FILE=FILEB,
        ACCMETH=VSAM,
        RECFORM=(FIXED,BLOCKED),
        LSRPOOL=01,
        JID=02,
        JREQ=ALL,
        LOG=YES,
        SERVREQ=(READ,BROWSE),
        STRNO=3
*
FILEC  DFHFCT TYPE=FILE,FILE=FILEC,
        ACCMETH=DAM,
        RECFORM=(UNDEFINED),
        SERVREQ=(READ,BROWSE),
        BLKSIZE=88,
        LRECL=60,
        EXTENT=01
*
    DFHFCT TYPE=SHRCTL,
        LSRPOOL=01,
        BUFFERS=(512(12),1024(12)),
        KEYLEN=10,
        STRNO=3
*-----*
    DFHFCT TYPE=FINAL
    END
/*
```

Figure 2-34 DFHFCT source

Assembling and link-editing the source

You must assemble your table by using the CICS Transaction Server macro library and link-edit it into your CICS load library. If you receive a return code that is greater than 4, remove the cause of the error and reassemble.

Using the DFHCSDUP utility

To convert the table entries into resource definitions, use the **MIGRATE** command of the DFHCSDUP offline utility program, as shown in Figure 2-35.

```
// LIBDEF *,SEARCH=(PRD2.CONFIG,PRD1.BASE,PRD2.SCEEBASE)
// EXEC DFHCSDUP,SIZE=600K          INIT AND LOAD CICS
    MIGRATE TABLE(DFHFCTOL)
/*
```

Figure 2-35 Using the MIGRATE command to transfer the FCT to the CSD

Checking the output from MIGRATE

The use of the **MIGRATE** command might produce some messages. Read these warning messages carefully. Figure 2-36 shows part of the **MIGRATE** command output.

```
// EXEC DFHCSDUP,SIZE=600K          INIT AND LOAD CICS
1S54I  PHASE DFHCSDUP IS TO BE FETCHED FROM PRD1.BASE

    MIGRATE TABLE(DFHFCTOL)

DFHCA5120 I PRIMARY CSD OPENED;  FILENAME: DFHCSD
DFHCA5143 I GROUP VSEFILES CREATED.
DFHCA5159 I FILE IESTRFL  DEFINED IN GROUP VSEFILES
DFHCA5159 I FILE IESPRB   DEFINED IN GROUP VSEFILES
DFHCA5159 I FILE IESCNTL  DEFINED IN GROUP VSEFILES
DFHCA5159 I FILE BSTCNTL  DEFINED IN GROUP VSEFILES
DFHCA5159 I FILE IESROUT  DEFINED IN GROUP VSEFILES
DFHCA5143 I GROUP MYFILES  CREATED.
DFHCA5159 I FILE FILEA    DEFINED IN GROUP MYFILES
DFHCA5159 I FILE FILEB    DEFINED IN GROUP MYFILES
DFHCA5172 W NO DEFINITION FOR FILE FILEC    CREATED. DAM  FILES ARE NOT
SUPPORTED BY RDO.
DFHCA5159 I LSRPOOL LSRPOOL1 DEFINED IN GROUP MYFILES
DFHCA5140 I TOTAL FILE      DEFINITIONS CREATED:      7
DFHCA5140 I TOTAL LSRPOOL   DEFINITIONS CREATED:      1
DFHCA5102 I WARNING MESSAGE(S) ISSUED WHILE PROCESSING MIGRATE COMMAND.
DFHCA5123 I PRIMARY CSD CLOSED;  FILENAME: DFHCSD

DFHCA5107 I COMMANDS EXECUTED SUCCESSFULLY: 0      COMMANDS GIVING WARNING(S): 1
DFHCA5108 I COMMANDS NOT EXECUTED AFTER ERROR(S): 0
DFHCA5109 I END OF DFHCSDUP UTILITY JOB. HIGHEST RETURN CODE WAS: 4
```

Figure 2-36 Output from FCT MIGRATE command

From the output listing, you can see the names of groups, files, and LSRPOOLS that are defined in the CSD.

Reassembling the DFHFCT source with MIGRATE=COMPLETE

If you want to load tables from the DFHFCT load library, you must code FCT=YES or FCT=xx for a suffixed table in your system initialization parameters.

You do not need to edit your macro source to remove all migrated definitions immediately after migrating. However, you must assemble your macro source with MIGRATE=COMPLETE to retain an FCT to manage resources that you cannot define by using RDO. You should eventually remove from your FCT source all the definitions that were successfully migrated to the CSD.

The MIGRATE=COMPLETE option suppresses the assembly of RDO-eligible FCT entries whether or they were migrated to the CSD; therefore, use caution if you are intending to use the resulting FCT table for RDO-eligible entries.

Figure 2-37 shows the DFHFCTOL job with MIGRATE=COMPLETE that we used.

```
// EXEC ASMA90,SIZE=(ASMA90,64K),PARM='EXIT(LIBEXIT(EDECKXIT)),SIZE(MAXC
      -200K,ABOVE)'
      TITLE 'DFHFCTOL -- OLD FCT MIGRATION'
      PUNCH ' CATALOG DFHFCTOL.OBJ REP=YES'
      DFHFCT TYPE=INITIAL,SUFFIX=OL,MIGRATE=COMPLETE
*
      DFHFCT TYPE=GROUP,GROUP=VSEFILES
      COPY IESZFCTO -- DATASET ENTRIES FOR Z/VSE - OTH PARTITION
*
*-----*
      DFHFCT TYPE=GROUP,GROUP=MYFILES
*
FILEA  DFHFCT TYPE=FILE,FILE=FILEA,*
...
FILEC  DFHFCT TYPE=FILE,FILE=FILEC,*
      ACCMETH=DAM,*
      RECFORM=(UNDEFINED),*
      SERVREQ=(READ,BROWSE),*
      BLKSIZE=88,*
      LRECL=60,*
      EXTENT=01
*
      DFHFCT TYPE=SHRCTL,*
      LSRPOOL=01,*
      BUFFERS=(512(12),1024(12)),*
      KEYLEN=10,*
      STRNO=3
*-----*
      DFHFCT TYPE=FINAL
      END
/*
```

Figure 2-37 DFHFCT source with MIGRATE=COMPLETE

Initializing CICS after migration

Complete the following steps to initialize CICS after the migration process is complete:

1. Use DFHCSDUP or RDO to ADD each group to a list.
2. Use the GRPLIST parameter and the non-VSAM FCT (if any) in the DFHSIT table or in the PARM field in CICS initialization.
3. Initialize the CICS partition.

Alternatively, you can wait until CICS initializes. Then, use RDO to INSTALL the groups that you want CICS to use. For an example of the install command, see Figure 2-32 on page 51.

A new file definition can be installed only if the existing file is closed and disabled or is closed and disenabled.

2.4.3 Migrating the DFHTCT table

The DFHTCT macro must be used exclusively to define resources that are not supported by RDO, including the following examples:

- ▶ Sequential devices
- ▶ Remote BTAM terminals for transaction routing
- ▶ Logical device codes

You must define the following items in the CSD:

- ▶ VTAM-connected terminals
- ▶ VSE consoles for use as CICS terminals
- ▶ Multiregion operation (MRO) connections and sessions
- ▶ Intersystem communication (ISC) links and sessions that use LU6.1 and APPC
- ▶ (LU6.2) protocols
- ▶ Indirect connections

A special facility that is called autoinstall can remove the need to migrate specific resources. For more information, see 2.3.7, “Autoinstall of resources” on page 40.

Migration steps

To perform the migration of your RDO-eligible table entries to RDO, complete the following steps:

1. Edit your DFHTCT source.
2. Assemble and link-edit the edited source.
3. Process the link-edited table by using the DFHCSDUP utility.
4. Check the output that is produced by the DFHCSDUP utility.
5. Use the **DFHCSDUP ADD** command to create a list of groups on your CSD.
6. Remove from your source all the table entries that were migrated and reassemble it.
7. Start your system by using the non-VTAM TCT (if any) and the list you created.

Editing your DFHTCT source

You must reassemble your DFHTCT source by using CICS TS macro libraries. If your source is large, you might find it more convenient to assemble small segments of the source individually because each resource definition on the CSD must belong to a CSD group. It is better to create many small groups for your files rather than one large group. Groups should be limited to 100 resource definitions.

The definitions within a group often have something in common, as shown in the following examples:

- ▶ Keep all of your TYPETERM definitions in one group.
- ▶ Keep all of your TERMINAL definitions by departmental function or geographical location.
- ▶ Keep AUTOINSTMODEL terminal definitions separately in a group.

You can specify the names to be given to groups of definitions that are generated from your DFHTCT macros by adding the following command:

```
DFHTCT TYPE=GROUP, GROUP=GROUPNAME
```

All definitions that follow a particular TYPE=GROUP macro statement are migrated into the named group in the CSD file. A new TYPE=GROUP statement overrides all previous statements, as shown in Figure 2-38 on page 59.

```

// EXEC ASMA90,SIZE=(ASMA90,64K),PARM='EXIT(LIBEXIT(EDECKXIT)),SIZE(MAXC
-200K,ABOVE)'
TITLE 'DFHTCTOL -- TCT'
PUNCH ' CATALOG DFHTCTOL.OBJ REP=YES'
DFHTCT TYPE=INITIAL,SUFFIX=OL,
ACCETH=(VTAM,NOVTAM),MIGRATE=YES
*
DFHTCT TYPE=SDSCI,
DEVADDR=SYSIPT,
DEVICE=2540,
DSCNAME=READER
*
DFHTCT TYPE=SDSCI,
DEVADDR=SYSLST,
DEVICE=1403,
DSCNAME=PRINTER
*
DFHTCT TYPE=LINE,
ACCETH=BSAM,
INAREAL=80,
TRMTYPE=CRLP,
ISADSCN=READER,
OSADSCN=PRINTER
*
DFHTCT TYPE=TERMINAL,
TRMIDNT=SAMA,
TRMTYPE=CRLP,
TRMSTAT=TRANSCIVE
*-----*
* L77A - VTAM TRMTYPE LUTYPE2 NETNAME - D72L301.
* L77B - VTAM TRMTYPE 3270 NETNAME - D72L302.
* L86P - VTAM NON-SNA 3270 PRINTER NETNAME - P42L303.
DFHTCT TYPE=GROUP,GROUP=MYTERM
*
VTRM1 DFHTCT TYPE=TERMINAL,TRMIDNT=L77A,TRMTYPE=LUTYPE2,TRMMODL=2,
TIOAL=(1500,1920),
BUFFER=1536,
RELREQ=(NO,YES),
FEATURE=(SELCTPEN,AUDALARM,DCKYBD),
CHNASSY=YES,
NETNAME=D72L301,
PRINTTO=(VTRM3),
GMMSG=YES,
ACCETH=VTAM,TCTUAL=32,
TRMSTAT=(TRANSCIVE)
*

```

Figure 2-38 DFHTCT source (Part 1 of 2)

Figure 2-39 shows part 2 of the DFHTCT source example.

```
VTRM2  DFHTCT TYPE=TERMINAL,TRMIDNT=L77B,TRMTYPE=3270,TRMMODL=2,      *
        CLASS=(CONV,VIDEO),TIOAL=1500,RELREQ=(NO,YES),                *
        FEATURE=(SELCTPEN,AUDALARM,DCKYBD),                            *
        NETNAME=D72L302,                                              *
        PRINTTO=(VTRM3),                                             *
        GMSG=YES,                                                    *
        ACCMETH=VTAM,TCTUAL=32,                                       *
        TRMSTAT=(TRANSCIVE)
*
VTRM3  DFHTCT TYPE=TERMINAL,TRMIDNT=L86P,TRMTYPE=3270P,TRMMODL=2,      *
        NETNAME=P42L303,                                             *
        RELREQ=(YES,YES),                                           *
        CLASS=(CONV,VIDEO),TIOAL=1500,TCTUAL=32,                    *
        ACCMETH=VTAM,TRMSTAT=(TRANSCIVE)
*
*-----*
        DFHTCT TYPE=FINAL
        END
/*
```

Figure 2-39 DFHTCT source (Part 2 of 2)

For more information about different types of macros, see Appendix G of *CICS Transaction Server for VSE/ESA Resource Definition Guide*, SC33-1653.

Assembling and link-editing the source

You must assemble your table by using the CICS TS macro library and link-edit it into your CICS load library.

If you receive a return code that is greater than 4, remove the cause of the error and reassemble.

Using the DFHCSDUP utility

To convert the table entries into resource definitions, use the **MIGRATE** command of the DFHCSDUP offline utility program, as shown in Figure 2-40.

```
// LIBDEF *,SEARCH=(PRD2.CONFIG,PRD1.BASE,PRD2.SCEEBASE)
// EXEC DFHCSDUP,SIZE=600K          INIT AND LOAD CICS
MIGRATE TABLE(DFHTCTOL)  TYPESGROUP(MYTYPE)
/*
```

Figure 2-40 Using the MIGRATE command to transfer the TCT to the CSD

The TYPETERM definitions are put into the CSD group that is named in the TYPESGROUP parameter.

The TYPETERM attributes of each DFHTCT TYPE=TERMINAL macro are checked with existing TYPETERM definitions. If they do not match with any of these definitions, a new TYPETERM is added to the CSD.

The following checks are done for the TYPETERMs:

- ▶ TYPETERMs in the group currently being created.
- ▶ TYPETERMs in the group that is specified in the TYPESGROUP parameter of the **MIGRATE** command.

Checking the output of the MIGRATE command

The use of the **MIGRATE** command might produce some messages. Read these warning messages carefully. The output is shown in Figure 2-41.

```
// EXEC DFHCSDUP,SIZE=600K          INIT AND LOAD CICS
1S54I  PHASE DFHCSDUP IS TO BE FETCHED FROM PRD1.BASE

      MIGRATE TABLE(DFHTCTOL)  TYPESGROUP(MYTYPE)

DFHCA5120 I PRIMARY CSD OPENED;  FILENAME: DFHCSD
DFHCA5280 I PROCESSING DEFINITIONS FROM LIBRARY MEMBER DFHRDTOL
DFHCA5143 I GROUP MYTERM   CREATED.
DFHCA5159 I TERMINAL L77A DEFINED IN GROUP MYTERM
DFHCA5143 I GROUP MYTYPE   CREATED.
DFHCA5159 I TYPETERM LU2000   DEFINED IN GROUP MYTYPE
DFHCA5159 I TERMINAL L77B DEFINED IN GROUP MYTERM
DFHCA5159 I TYPETERM 3270000   DEFINED IN GROUP MYTYPE
DFHCA5159 I TERMINAL L86P DEFINED IN GROUP MYTERM
DFHCA5159 I TYPETERM 3270P000 DEFINED IN GROUP MYTYPE
DFHCA5140 I TOTAL TERMINAL   DEFINITIONS CREATED:      3
DFHCA5140 I TOTAL TYPETERM   DEFINITIONS CREATED:      3
DFHCA5101 I MIGRATE COMMAND EXECUTED SUCCESSFULLY.
DFHCA5123 I PRIMARY CSD CLOSED;  FILENAME: DFHCSD

DFHCA5107 I COMMANDS EXECUTED SUCCESSFULLY: 1      COMMANDS GIVING WARNING(S): 0
DFHCA5108 I COMMANDS NOT EXECUTED AFTER ERROR(S): 0
DFHCA5109 I END OF DFHCSDUP UTILITY JOB. HIGHEST RETURN CODE WAS: 0
```

Figure 2-41 Output from TCT MIGRATE command

From the output listing, you can see the names of the groups, terminals, and typeterms that are defined from library member DFHRDTOL (which contains the RDO-eligible definitions in command format).

Reassembling the DFHTCT source with MIGRATE=COMPLETE

If you need to use a Translation Communication Tool (TCT), you must code TCT=xx (where xx is the suffix of your table) in the DFHSIT macro.

You do not need to edit your macro source to remove all migrated definitions immediately after migrating, but you must assemble your macro source with MIGRATE=COMPLETE to retain a TCT to manage resources that you cannot define by using RDO. You should eventually remove from your TCT all the definitions that were successfully migrated to the CSD.

Initializing CICS after migration

To initialize CICS after migration, complete the following steps:

1. Use DFHCSDUP or RDO to ADD each group to a list.
2. Use the GRPLIST parameter and the non-VTAM TCT (if any) in your system initialization parameters or in the PARM field in CICS initialization.
3. Initialize the CICS partition.

You must install some migrated definitions in the system when you initialize it.

If you can use RDO to manage all of your resources, you can abandon your DFHTCT macro source and code TCT=NO in your system initialization parameters.

To create a Virtual Telecommunications Access Method (VTAM) terminal definition in the CSD, you use the **DEFINE TERMINAL** command in the following ways:

- ▶ Resource definition online (RDO): CEDA or CEDB transactions
- ▶ DFHCSDUP offline utility
- ▶ Autoinstall (see 2.3.7, “Autoinstall of resources” on page 40)

To create a typeterm definition in the CSD, you use the **DEFINE TYPETERM** command in the following ways:

- ▶ Resource definition online (RDO): CEDA or CEDB transactions
- ▶ DFHCSDUP offline utility

2.4.4 Other considerations

The following section describes other changes that were introduced with CICS TS.

Obsolete tables

Table 2-5 lists the tables that are obsolete in the CICS TS.

Table 2-5 Obsolete tables

Table name	Comments
ALT	In the CICS TS, the program management component is reengineered to such an extent that the virtual storage benefits that were offered by the Application Load Table are superseded.
NLT	There is no longer any need to optimize the working set because of the internal restructuring of CICS.
SNT	The Sign-on Table is obsolete because of the removal of CICS internal security in CICS Transaction Server.
PCT and PPT	These resources are installed only by using the CSD. However, the DFHPCT and DFHPPT table macros are retained for migrating PCT and PPT definitions to the CSD.

TRANCLASS definition

TRANCLASS is a resource object with which you can associate scheduling and dispatching priorities with TRANSACTION definitions. These definitions replace and extend the CMXT parameter.

By putting your transactions into transaction classes, you can control how CICS dispatches tasks. Transaction classes become RDO-defined resources in CICS.

Transaction Server allows you to have as many transaction classes as you need. Transaction classes are now identified by eight-character names instead of the numbering scheme that was used on the now-obsolete CMXT system initialization parameter.

Group DFHTCL contains definitions for the 10 CICS-defined TClasses (named DFHTCL01 - DFHTCL10) provided as replacements for the 10 numbered TClasses that were defined on the CMXT parameter.

To ensure that the default TRANCLASS names have the same effect as your current TCLASS members, you should copy and modify the DFHTCLxx definitions and specify values that correspond to your old system values.

Figure 2-42 shows the supplied RDO TRANCLASS definitions.

OBJECT CHARACTERISTICS		CICS RELEASE = 0420	
CEDA View TRANClass(DFHTCL01)			
TRANClass	:	DFHTCL01	
Group	:	DFHTCL	
Description	:	Replacement for CMXT class 1	
CLASS LIMITS			
Maxactive	:	001	0-999
Purgethresh	:	No	No 1-1000000
SYSID=CIC1 APPLID=DBDCCICS			
PF 1 HELP 2 COM 3 END 6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL			

Figure 2-42 RDO VIEW TRANCLASS: CEDA transaction

Shared data tables

CICS Transaction Server extends CICS file control by introducing shared data tables.

Shared data tables make the data tables facility that was introduced in CICS/VSE 2.2 obsolete. By using ESA/390 data spaces and cross memory services, the tables bring several advantages.

Regardless of how many partitions share the data table, CICS stores records for a data table in a z/VSE data space. The data space is created when the first file to be defined as a data table is opened in the partition. The data space is used by all CICS data tables that are owned by that partition. It is retained until CICS is shut down in that partition.

No changes are required to the file definitions for data tables. You might need to increase the size of the data space that is available in the VSE system. The data space for shared data tables is acquired in units of 16 MB. For more information about the use shared data tables, see *CICS Transaction Server for VSE/ESA Shared Data Tables Guide*.



Security

Security in IBM CICS Transaction Server (CICS TS) is provided through the services of an external security manager (ESM) that conforms to the z/VSE system authorization facility (SAF) interface. CICS TS does not provide any internal security mechanisms. To provide the necessary security for CICS resources, CICS TS issues RACROUTE calls by using the z/VSE SAF interface to route authorization requests to an ESM within CICS transaction processing.

The following levels of Security Management functionality are available:

- ▶ Basic security manager (BSM)

The BSM is provided with the z/VSE system package. The BSM provides considerable ESM functionality and sign-on and resource security checking. Since its introduction in VSE/ESA 2.4, it was considerably extended in later VSE versions.

- ▶ External security manager (ESM)

An ESM is provided by an independent software vendor (ISV) that conforms to the SAF interface. The ISVs support all of the various security checks that are necessary to protect your application programs and CICS system components from misuse. The full-function ESM must be installed separately.

In this chapter, we describe the security implications of migrating to the IBM CICS Transaction Server for z/VSE 2.1. We also describe the basic security support that is provided by the BSM in z/VSE Version 6 Release 1 and how the CICS TS makes use of this support. This chapter references only items of function and support as they relate to an ESM. Also, only security items in the z/VSE system that relate to the CICS TS are described in this document.

This chapter includes the following topics:

- ▶ 3.1, “z/VSE Version 6 Release 1 security” on page 66
- ▶ 3.2, “CICS Transaction Server for z/VSE 2.1 security” on page 83
- ▶ 3.3, “Security Migration Aid” on page 105
- ▶ 3.4, “Summary” on page 106

3.1 z/VSE Version 6 Release 1 security

Security is implemented with an ESM; either the BSM or a vendor product.

3.1.1 z/VSE V6R1 security options

This section describes the z/VSE V6R1 security options.

IPL SYS command

The **IPL SYS** command provides an ESM = option, SERVPART= option and the SEC=RECOVER option. The z/VSE Function Selection dialog for tailoring the IPL procedure was updated to reflect these options. An example of these options is shown in Figure 3-1.

TAS\$ICM1		TAILOR IPL PROCEDURE: SYS COMMAND	
Enter the required data and press PF5=PROCESS			
BUFLD.....	1	Load printer buffers?	1=yes, 2=no
CHANQ.....	_____	Number of channel queue entries	
DASDFP.....	1	DASD file protection?	1=yes, 2=no
SUBLIB.....	_____	Number of sublibraries	
VMCF.....	_	CMS-VSE console interface?	1=yes, 2=no, or blank for system default
SEC.....	3	access control security?	1=yes, 2=no, 3=NOTAPE, 4=RECOVER, 5=JCL, 6=JCL,NOTAPE (1)
ESM.....	_____	Name of the ESM initial. phase	(2)
SERVPART.....	FB	Security server partition (F1,F2, ... FB)	(3)
TRKHLD.....	12	Number of hold requests	
PF1=HELP	2=REDISPLAY	3=END	5=PROCESS
	8=FORWARD		

Figure 3-1 Tailor IPL SYS parameter

The following options are shown in Figure 3-1:

- ▶ Option SEC=RECOVER switches off all security completely.
Use this option only for problem determination. Option SEC=JCL activates access control checking and JCL security. With JCL security, you can protect operands of specific JCL statements.
- ▶ Option to specify the ESM initialization phase name.
This option is found in the ESM installation documentation. If nothing is specified, BSM is activated.
- ▶ Option to specify the partition that the Security Server is running.
If an ESM is specified, this option specifies the partition of the Security Server that is required by the ESM. For more information, see the ESM installation documentation.

DTSECTAB

The DTSECTAB table is required when security checking for batch operations is wanted. Although the DTSECTAB included user information for these checks in older VSE releases, it is no longer the case for z/VSE 6.1.

Security-related user profiles from z/VSE Interactive Interface are not stored in DTSECTAB, but are in VSE.CONTROL.FILE. You must define your current z/VSE Interactive Interface users to the VSE.CONTROL.FILE.

The user IDs FORSEC and DUMMY are the only user IDs that are stored in the DTSECTAB and the VSE.CONTROL.FILE. These IDs are used at system startup and should never be removed.

If you still have user profiles in DTSECTAB, migrate these user profiles by using the Maintain User Profiles dialog as described in “Maintaining Users with the Interactive Interface” on page 84 or the batch utility IESBLDUP. (The IESBLDUP utility is described in the *z/VSE Installation* documentation.) It can also be used to create a status report of your current system's users.

Note: The access control concept for resources that are defined in the DTSECTAB did not change with z/VSE V6R1.

VSE.CONTROL.FILE

All user profiles are stored in the VSE.CONTROL.FILE (a VSAM-KSDS file). User profiles from z/VSE Interactive Interface and from the old CICS SNT table are stored in the VSE.CONTROL.FILE. The Maintain User Profiles dialogs were updated to handle defining security-related information in the user profiles.

The user IDs FORSEC and DUMMY are the only user IDs that are stored in the DTSECTAB and VSE.CONTROL.FILE. These IDs are used at system start and should never be removed.

Do not copy an old VSE.CONTROL.FILE into a z/VSE V6R1 VSE.CONTROL.FILE as the file changed. Use the IESBLDUP utility to migrate.

VSE/POWER

VSE/POWER security provides spool access protection for the POWER queues. This support applies to the BSM and ESM. To activate the VSE/POWER spool access protection, enable VSE for batch security (IPL SYS SEC=YES statement) and include the statement SET SECAC=SYS into the POWER startup deck.

3.1.2 Basic security manager

The BSM is part of z/VSE V6R1. It provides sign-on security and resource security for CICS TS. The BSM security functionality is sufficient for most VSE customers. The BSM is always active during the start of your system. After start, control can be passed to an ESM.

The BSM requires the user profiles in the VSE.CONTROL.FILE, the resource profiles for online resources in the VSE.BSTCNTRL.FILE, and the resource control information for files, libraries, sublibraries, and members in the DTSECTAB. The BSM uses a Security Server to perform security checking from these files.

BSM can control the access to the following items:

- ▶ Online resources checking, including sign-on checking is enabled by using the CICS TS DFHSIT option “SEC=YES”.
- ▶ Batch resource checking is enabled by using IPL “SYS SEC=YES”.

Each option can be enabled independently.

Figure 3-2 shows a schematic overview of the z/VSE BSM and its profile repositories. For more information about the administration and operation of BSM, see 3.1.3, “Security Server” on page 69, and 3.1.7, “Maintaining BSM security profiles” on page 77.

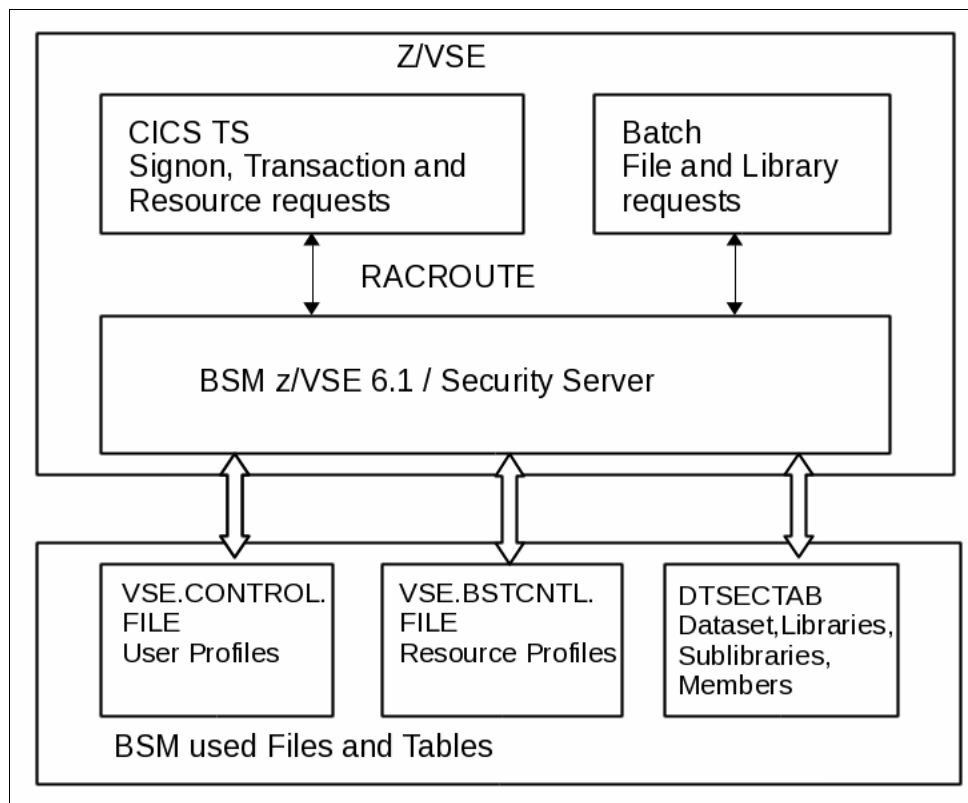


Figure 3-2 z/VSE BSM schematic overview

The BSM supports many resource classes that are used by CICS TS, as listed in Table 3-1.

Table 3-1 BSM Resource classes used by CICS TS

Resource class	Description
TCICSTRN	Transaction Security
ACICSPCT	Started Transaction
DCICSDCT	Transient Data Queue
FCICSFCT	File
JCICSJCT	Journals
MCICSPPT	Application Programs
SCICSTST	Temporary Storage Queues

Resource class	Description
APPL	Connect Security
FACILITY	Misc. Resources (Report Controller)
SURROGAT	Surrogate User Checking

Note: Transaction security is provided in z/VSE 6.1 only by profiles of the TCICSTRN resource class. The method to protect transactions by using table DTSECTXN is no longer supported. If you still use DTSECTXN, perform the migration on your existing VSE system, as described in *z/VSE Administration*, SC34-2692.

For more information about BSM, see *Security on IBM z/VSE*, SG24-7691.

3.1.3 Security Server

The Security Server is part of the BSM. It can run in any static partition (the default is FB partition). The Security Server controls the access to the VSE.CONTROL.FILE and to VSE.BSTCNTL.FILE.

Initial z/VSE V6R1 installation and the Security Server

On the initial installation of the z/VSE V6R1, you are prompted if you want to run with security ON (see Figure 3-3). Reply YES now in the installation process because it is easier to have the system set up the security facilities and turn it off than to reply NO and set up the security facilities later.

```
BG 0000 IESIO079D DO YOU WANT TO RUN YOUR SYSTEM WITH SECURITY ON? YES/NO
0 yes
```

Figure 3-3 Initial Install Security option

The Security Server runs in partition FB by default in the installation. Before POWER is started, you see the SECserv job start (see Figure 3-4).

```
FB 0011 // JOB SECserv
        DATE 10/18/2016, CLOCK 18/11/21
FB 0095 1J022I CPU CRYPTOGRAPHIC ASSIST FEATURE AVAILABLE.
FB 0095 1J017I CRYPTO HARDWARE NOT INSTALLED OR NOT DEFINED.
FB 0094 4228I FILE IJSYSCT OPEN ERROR X'B4'(180) CAT=..N/A.. (1)
        (OPNCT-20) VOLUME 'DOSRES' NOT OWNED BY VSAM
FB 0094 4228I FILE IESCNTL OPEN ERROR X'B4'(180) CAT=VSESPUC ( 4,AD, 2)
        (OPNHC-10) RC X'00000004' FROM CATLG
FB 0094 BST300E THE II CONTROL FILE COULD NOT BE OPENED.
```

Figure 3-4 Install SECserv startup messages

You can ignore the 4228I message. The installation process continues.

Postinstallation and the Security Server

The BSM is always active during start, independent of the SEC setting on your IPL SYS command.

After the installation, the IPL SYS parameter might be changed to SYS SEC=NO. No resource security checking is done in VSE. This method did not change from VSE resource checking in the past.

The CICS TS sign-on and transaction-attach security are still active.

Normal Security Server startup

The Security Server must run in a static partition (ALLOC=2M; SIZE=512K) and it is started before any other VSE partition, including VSE/POWER. In the initial setup, it runs in FB, but the default partition FB can be changed to any other static partition. This partition is specified in the SERVPART parameter option in the **IPL SYS** command.

If you require FB for production, F4 might be a good candidate for running the BSM. F4 cannot support a CICS TS system with subsystem storage protection (STGPROT=YES specified as system initialization parameter).

In the ASI procedure for partition BG startup, // EXEC BSSINIT initializes the BSM. Then, BG start continues.

Before VSE/POWER starts, the FB start procedure is run to start the Security Server with // EXEC BSTPSTS.

If you are running an ESM, the start of the ESM replaces the start of the BSM. Figure 3-5 shows the normal SECSERV startup message.

```
FB 0011 // JOB SECSERV
        DATE 10/27/2016, CLOCK 08/53/55
FB 0011 ID (PARAMETERS SUPPRESSED)
BG 0000 BST001I BASIC SECURITY MANAGER INITIALIZED
BG 0000 STOP
F1 0001 // JOB POWSTART
```

Figure 3-5 Normal SECSERV startup message

Security Server priority

The Security Server partition runs after VSE/POWER and before the CICS partitions, as shown in Example 3-1.

Example 3-1 CICS partitions

```
PRTY BG,FA,F9,F8,F6,F5,F4,F2,F7,FB,F3,F1
```

Security Server shutdown

When you shut down the system and VSE/POWER ends, the messages that are shown in Figure 3-6 on page 71 are displayed.

```

F1 0001 1Q21I  VSE/POWER HAS BEEN TERMINATED
F1 0001 ID (PARAMETERS SUPPRESSED)
F1 0001 * -----
F1 0001 * SECURITY SERVER PARTITION WILL BE STOPPED
F1 0001 * -----
MSG FB,DATA=STOPNOREP
AR 0015 1I40I  READY
FB 0011 BST212I STOP COMMAND ACCEPTED.
FB 0011 1N90I  EOP WAS FORCED BY EOJ
FB 0011 EOJ SECSERV  MAX.RETURN CODE=0099

```

Figure 3-6 Normal shutdown message

Communicating with the Security Server

There are commands that can be entered on the console that use the MSG command to communicate with the Security Server. The following message format is used:

MSG FB,DATA=command

If you issue the MSG command without the data operand, a list of command options display. Figure 3-7 shows the Security Server, which also supports the available hardware crypto facilities.

```

msg fb
AR 0015 1I40I  READY
FB 0011 BST221I POSSIBLE SECURITY SERVER COMMANDS ARE:
FB 0011  DBSTARTCACHE .....: STARTS DATABASE CACHING
FB 0011  DBSTOPCACHE .....: STOPS DATABASE CACHING
FB 0011  STATUS .....: SHOWS TOTAL SERVER STATUS
FB 0011  STATUS=ALL .....: SHOWS TOTAL SERVER STATUS
FB 0011  STATUS=MAIN|PS|DB ...: SHOWS SELECTED SECSERV STATUS
FB 0011  STATUS=CR|CPACF .....: SHOWS SELECTED CRYPTO STATUS
FB 0011  LOGTIME=N .....: SETS LOGTIME TO N MINUTES (1..9)
FB 0011  RESET .....: CLEANUP EVERYTHING
FB 0011  STOP | SHUTDOWN .....: STOPS THE SERVER (USE WITH CAUTION)
FB 0011  SHUTDOWN NOPROMPT ...: STOPS THE SERVER WITHOUT CONFIRM.
FB 0011  OPENCNTL .....: OPENS THE II CONTROL FILE
FB 0011  CLOSECNTL .....: CLOSSES THE II CONTROL FILE
FB 0011  OPENBST .....: OPENS THE BSM CONTROL FILE
FB 0011  CLOSEBST .....: CLOSSES THE BSM CONTROL FILE
FB 0011  HARDWARE CRYPTO COMMANDS:
FB 0011  APBUSY=NN .....: SET AP CRYPTO WAIT ON BUSY (0..99)
FB 001I ....

```

Figure 3-7 Security Server command available

Stopping the Security Server

Note: Use the **STOP** command carefully.

The **STOP** command stops the Security Server. Any new user that attempts to log on hangs instead, with no informational message. Previously active users continue to work, but with unpredictable results if they are using ICCF.

If you issue the **STOP** command, the warning messages that are shown in Figure 3-8 are displayed.

```
msg fb,data=stop
AR 0015 1140I  READY
FB-0011 BST226W DO YOU REALLY WANT TO STOP THE SECURITY SERVER? (Y/N)
11 y
FB 0011 BST212I STOP COMMAND ACCEPTED.
FB-0011 // PAUSE TO RESTART THE SECURITY SERVER ENTER '// EXEC PROC=RESTASEC'
```

Figure 3-8 Messages after STOP issued to the Security Server

The Security Server can be restarted by issuing `// EXEC PROC=RESTASEC`.

If you pressed the Enter key to the FB partition and see the message: “Please assign sysrdr”, you must restart VSE to get Security Server up and running.

Note: In a system without batch-security (SYS SEC=NO), you can restart by running `EXEC PROC=$BJCL` or your own FB startup procedure.

When Security Server is restarted, you do not receive any confirmation message that Security Server is initialized.

Canceling the Security Server

If FB is canceled, a warning message is issued (see Figure 3-9). If the warning message is ignored and the Security Server is canceled, you must restart VSE to get Security Server up and running.

```
replid
AR 0015 1188I NO REPLIES OUTSTANDING
cancel fb
AR 0015 1155D CANCEL SEC.SRVR. ?  REPLY YES OR NO
AR+0015
15 yes
AR 0015 1140I  READY
FB 0011 BST231I SERVER WAS CANCELLED. CLEANING UP AND EXITING.
FB 0011 1N90I  EOP WAS FORCED BY EOJ
FB 0011 1S78I  JOB TERMINATED DUE TO  CANCEL COMMAND
FB 0011 EOJ SECSERV  MAX.RETURN CODE=0099
          DATE 10/27/2016, CLOCK 13/58/27, DURATION  00/20/15
FB-0011 1C10D  PLEASE ASSIGN SYSRDR.
```

Figure 3-9 Canceling the FB partition

A warning message is shown after the Security Server is cancelled.

Note: If you stop or cancel FB, the security on your system is off. Users cannot log on to the system. Any user that is logged on continues working. ICCF users that are signed on might experience unpredictable results (including hanging) as the ICCF waits for responses from the Security Server.

Security Server commands

To communicate with the Security Server, you must issue the following MSG command on the console: << is the period part of the command?>>

MSG FB,DATA=command.

Figure 3-7 on page 71 lists the available commands that include the following definitions:

- Server database caching

With database caching, recently accessed VSE Control records are loaded into a data space, which avoids VSAM I/O. Use DBSTARTCACHE (see Figure 3-10) and DBSTOPCACHE (see Figure 3-11) to control this function. On start, caching is OFF.

```
msg fb,data=dbstartcache
AR 0015 1I40I  READY
FB 0154 BST310I DATABASE CACHING STARTED.
```

Figure 3-10 Starting Security Server database caching

Figure 3-11 shows an example of the **DBSTOPCACHE** command.

```
msg fb,data=dbstopcache
AR 0015 1I40I  READY
FB 0154 BST309I DATABASE CACHING ENDED.
```

Figure 3-11 Stopping Security Server database caching

- Server log time

The LOGTIME is the frequency with which the server updates its database with the latest in user profiles. The default setting is 5 minutes.

- Opening and closing the VSE Control File

The VSE.CONTROL.FILE can be closed to the Security Server. This closure does not affect users that are logging on to the system, as the cached file is used. However, it does affect any new user that is added to the VSE.CONTROL.FILE; they receive the following message: SIGN-ON FAILED. INFORMATION LOGGED.

► Status

The status of the Security Server can be displayed, as shown in Figure 3-12).

```
msg fb,data=status=all
AR 0015 1I40I  READY
FB 0011 BST223I CURRENT STATUS OF THE SECURITY TRANSACTION SERVER:
FB 0011 SERVER GENERAL STATUS:
FB 0011  SERVER WAS STARTED AT ..... : 10/27/2016 10:58:18
FB 0011  TIME ELAPSED (DDDD::HH:MM:SS) ..... : 0000::01:43:46
FB 0011  NO. OF REQUESTS IN XPCC QUEUE ..... : 0
FB 0011  NO. OF REQ. IN INTERNAL REPLY QUEUE : 0
FB 0011  NO. OF FREE (REUSABLE) REQ. BLOCKS . : 9
FB 0011  NO. OF CURRENTLY ALLOCATED BLOCKS .. : 10
FB 0011  SIZE OF ONE REQUEST BLOCK (BYTES) .. : 740
FB 0011  TOTAL NO. OF PS/DB REQUESTS SO FAR . : 163
FB 0011  TOTAL NO. OF AP REQUESTS SO FAR .... : -
FB 0011  HIGHEST NO. OF PARALLEL PS/DB REQ .. : 1
FB 0011  SERVER DEBUG LEVEL ..... : SMALL,CONSOFF
FB 0011  BSM DEBUG LEVEL ..... : CONSOFF
FB 0011  NUMBER OF SERVER RESETTINGS ..... : 0
FB 0011 PROFILE SERVICE SUBTASK STATUS:
FB 0011  PS SUBTASK STARTED ..... : YES
FB 0011  NO. OF ITEMS IN PS REQUEST QUEUE ... : 0
FB 0011 DATABASE SUBTASK STATUS:
FB 0011  DB SUBTASK STARTED ..... : YES
FB 0011  II CONTROL FILE OPEN ..... : YES
FB 0011  BSM CONTROL FILE OPEN ..... : YES
FB 0011  DATABASE CACHE INITIALIZED OK ..... : YES
FB 0011  DATABASE CACHING ..... : OFF
FB 0011  LOGTIME INTERVAL IN MINUTES ..... : 5
FB 0011  NO. OF ITEMS IN DB REQUEST QUEUE ... : 0
FB 0011 CRYPTO DEVICE DRIVER STATUS:
FB 0011  AP CRYPTO SUBTASK STARTED ..... : NO
FB 0011 END OF CRYPTO DEVICE DRIVER STATUS
FB 0011 CPU CRYPTOGRAPHIC ASSIST FEATURE:
FB 0011  CPACF AVAILABLE ..... : YES
FB 0011  INSTALLED CPACF FUNCTIONS:
FB 0011  DES, TDES-128, TDES-192
FB 0011  AES-128, AES-192, AES-256, PRNG
FB 0011  SHA-1, SHA-256, SHA-512
FB 0011  KMAC_DES, KMAC_TDES128, KMAC_TDES192
FB 0011 PROTECTED KEY CPACF FUNCTIONS:
FB 0011  ENCR_DES, ENCR_TDES128, ENCR_TDES192
FB 0011  ENCR_AES128, ENCR_AES192, ENCR_AES256
FB 0011  KMAC_ENCR_DES, KMAC_ENCR_TDES128, KMAC_ENCR_TDES192
FB 0011 ENCRYPTION MODES:
FB 0011  ECB, CBC
FB 0011 END OF CPACF STATUS
```

Figure 3-12 Security Server - STATUS=ALL - function

3.1.4 Batch security with DTSECTAB

The z/VSE batch environment is secured with the DTSECTAB. Resources to be protected must be defined in DTSECTAB. The following resources can be protected:

- ▶ All libraries, sublibraries, and all their members.
- ▶ Files:
 - All VSE/VSAM KSDS, RRDS, VRDS, and ESDS that are accessed directly by using an ACB macro. Also, all VSE/VSAM-managed SAM files that are accessed by using a DTFSD or an ACB macro, or by appropriate file definition statements of the IBM compilers that are used at your installation. (The file's VOLSER and catalog are not checked.)

When VSE/VSAM data is accessed by using a path, the path name is used for access checking.

- All non-VSAM disk files and standard-labeled tape files that are defined by a file description macro (DTFxx) or by appropriate file definition statements of the IBM compilers that are used at your installation.

For each resource to be protected, the security administrator defines one or more access control classes in the corresponding resource profile. In general, resources without an entry in DTSECTAB are not protected.

Access control classes are 1 - 32, which are assigned to the resource. Access Control by using classes establishes an individual access right for the user. The ACC parameter defines access control classes together with associated access rights. A user's access control class can have one of the following access rights:

- ▶ ALT (Alter)
- ▶ UPD (Update)
- ▶ READ (Read-only)
- ▶ CON (Connect)

These access rights are ordered hierarchically; that is, ALTER implies UPDATE, UPDATE implies READ, and READ implies CONNECT.

A user can access a resource by specifying the access control class in their user profile. The dialog "Maintain User Profiles" (fastpath 2-1-1) is used.

The DTSECTAB that is included with z/VSE is stored in source form in ICCF library 59 in member DTSECTRC.

3.1.5 BSM logging and reporting

A z/VSE optional program, called VSE/Access Control-Logging and Reporting (ACLR), is available for logging and reporting on the BSM. It logs accesses to the system and resources that are defined in the DTSECTAB. It does not log access to CICS transactions.

For more information about this facility, see *VSE/Access Control-Logging and Reporting, Program Reference and Operations Guide*.

With z/VSE 4.3 (and later), you can use Data Management Facility (DMF) instead of ACLR for this auditing.

To prepare your system to create SMF records and send them to DMF, you must catalog only the \$SVALOG.PHASE into IJSYSRS.SYSLIB. You can use skeleton SKSECLOG from ICCF library 59. The phase \$SVALOG ensures that the logger program is loaded in the system when the batch security is active. If DMF is not active, you do not receive any audit record from the logger program.

Logging of invalid signon attempts or transaction security violations are logged to the CICS TS message log for the CICS TS that is being accessed and the z/VSE console. A sample message log is shown in Figure 3-13.

```
DFHSN1100 10/28/16 08:20:45 DBDCCICS Signon at netname D0600001 by user TEST1
in group * is complete.
DFHXS1111 10/28/16 08:20:50 DBDCCICS TECC Security violation by user TEST1 at
netname D0600001 for resource TECC in
          class TCICSTRN. SAF codes are (X'00000008',X'00000000'). ESM codes
are (X'00000008',X'00000000').
DFHAC2003 10/28/16 08:20:50 DBDCCICS Security violation has been detected term
id = A000, trans id = TECC, userid =
          TEST1.
DFHSN1200 10/28/16 08:21:03 DBDCCICS Signoff at netname D0600001 by user TEST1
is complete. 5 transactions entered with
          1 errors.
```

Figure 3-13 Security Server: CICS TS message log (security violation messages)

A sample console log is shown in Figure 3-14.

```
F2 0162 BST120I USER(TEST1  )
      BST120I  TECC CL(TCICSTRN)
      BST120I  INSUFFICIENT ACCESS AUTHORITY
      BST120I  FROM TECC
      BST120I  ACCESS INTENT(READ  ) ACCESS ALLOWED(NONE  )
```

Figure 3-14 Security Server: VSE/ESA console log message

3.1.6 System Authorization Facility

The SAF provides centralized control over security processing by using a system service that is called the z/VSE router. The z/VSE router features a common interface for all products that request access control checking for resources or authorization-related requests. A resource manager, such as CICS, issues a RACROUTE macro call to the z/VSE SAF router. The z/VSE router then calls the security manager (BSM or ESM).

The z/VSE router provides an optional installation exit. This exit can be used to pass control to your own ESM or to get control before going to a vendor-supplied ESM.

The exit must be named ICHRTX00 and must be in the shared virtual area (SVA). This exit also replaces the user Multiregion operation (MRO) security identification program DFHACEE.

For more information about the z/VSE Router exit, see *CICS Transaction Server for VSE/ESA Customization Guide*, SC33-1652.

3.1.7 Maintaining BSM security profiles

BSM resource profiles are stored in the VSE.BSTCNTL.FILE (BSTCNTL), which is a VSAM/KSDS file that is allocated during z/VSE 6.1 installation process. The required resource profiles for system-provided resources are defined during z/VSE installation.

To ease the administration effort for permitting access to resources, the BSM provides a GROUP concept. z/VSE predefines groups GROUPXY, which represents the previously (in CICS z/VSE 2.3) security access keys. You can establish any groups of your own, mapping your organizational structure.

z/VSE offers the following options to administer these profiles:

- ▶ Batch utility BSTADMIN
- ▶ Interactive Interface dialogs (IUI)

Batch utility BSTADMIN

The BSTADMIN utility is used to perform the following actions:

- ▶ Add, change, and delete resource profiles. Certain resource profiles can be specified in a generic way. With the DATA('MYPACKAGE') parameter, you can supply an installation-defined tag for this resource.
- ▶ Add, change, and delete group profiles. The group concept greatly simplifies the access to certain resources for a group of people.
- ▶ Connect and remove user IDs to and from group profiles.
- ▶ Permit access of a group or user ID to a specific resource.
- ▶ List resources profiles and groups.
- ▶ Perform other actions, such as activating classes or settings for password.

After maintaining any BSM profiles, the BSM data space must be refreshed by using the BSTADMIN command **PERFORM DATASPACE REFRESH**. For more information about BSTADMIN, see *z/VSE Administration*, SC34-2692.

Figure 3-15 shows a job that is used to add resource profiles.

```
* $$ JOB JNM=BSTADD,CLASS=0,DISP=K,NTFY=YES
* $$ LST CLASS=A,DISP=H
// JOB BSTADD          ADD RESOURCE PROFILES
// EXEC BSTADMIN
* DISCRETE TRANSACTION PROFILE BTM1
AD TCICSTRN BTM1      DATA('BALTIMORE TRANS')
* GENERIC TRANSACTION PROFILES STARTING WITH DAF*
AD TCICSTRN DAF GEN   DATA('MY APPL RES')
/*
/&
* $$ E0J
```

Figure 3-15 BSTADMIN - ADD resource sample

Figure 3-16 shows a job that is used to add a group and connect user IDs to it.

```
* $$ JOB JNM=BSTGRP,CLASS=0,DISP=K,NTFY=YES
* $$ LST CLASS=A,DISP=H
// JOB BSTGRP          ADD GROUP AND CONNECT USERS
// EXEC BSTADMIN
* DEFINE GROUP DEPT SALEDEP
AG SALEDEP             DATA('SALES DEPT')
* CONNECT USERS TO SALES DEPT
* USERIDS ARE NOT CHECKED FOR EXISTENCE IN CONTROL FILE
CO SALEDEP USER1
CO SALEDEP USER2
CO SALEDEP CLERKY
/*
/&
* $$ EOJ
```

Figure 3-16 BSTADMIN - ADD group and CONNECT user IDs

Figure 3-17 shows a job that is used to permit access of groups and user IDs to resources.

```
* $$ JOB JNM=BSTPER,CLASS=0,DISP=K,NTFY=YES
* $$ LST CLASS=A,DISP=H
// JOB BSTPER          PERMIT USER/GROUP ACCESS TO RESOURCES
// EXEC BSTADMIN
* PERMIT TO BTM1
PE TCICSTRN BTM1       ID(SYSA) ACCESS(UPDATE)
* PERMIT TO DAF*
PE TCICSTRN DAF GEN    ID(SALEDEP) ACCESS(READ)
/*
/&
* $$ EOJ
```

Figure 3-17 BSTADMIN - PERMIT access to resources

IUI Dialogs

The Interactive Interface supports BSM resource administration for the following items:

- ▶ Resource profiles: Use function “Unified BSM Resource Profile Maintenance” (Fastpath 2-8-6) to maintain the profiles in the BSTCNTL/VSAM file directly.
- ▶ Group Maintenance: Use function “BSM Group Maintenance” (2-8-2) to maintain the group profiles.

Figure 3-18 shows the Unified BSM Resource Profile Maintenance panel, in which all supported BSM classes for profile maintenance are listed.

IESADMBSLC		BSM RESOURCE CLASSES	
START....			
OPTIONS:		6=PROFILE LIST	
OPT	RESOURCE CLASS NAME	STATUS	
—	ACICSPCT	ACTIVE	
—	APPL	ACTIVE	
—	DCICSDCT	ACTIVE	
—	FACILITY	ACTIVE	
—	FCICSFCT	ACTIVE	
—	JCICSJCT	ACTIVE	
—	MCICSPPT	ACTIVE	
—	MQADMIN	INACTIVE	
—	MQCMDS	INACTIVE	
—	MQCONN	INACTIVE	
—	MQNLIST	INACTIVE	
—	MQQUEUE	INACTIVE	
PF1=HELP	2=REFRESH	3=END	
	8=FORWARD	9=PRINT	

Figure 3-18 IUI dialog - BSM Resource classes

In the BSM Resource class window, select the class with option **6** to maintain the profile list.

Figure 3-19 shows the profile list for class TCICSTRN.

IESADMBSLE				MAINTAIN SECURITY PROFILES	
BSM RESOURCE CLASS: TCICSTRN		(START is Case Sensitive)		STATUS: ACTIVE	
START....					
OPTIONS: 1 = ADD 2 = CHANGE 5 = DELETE 6 = ACCESS LIST					
OPT	PROFILE NAME	DESCRIPTION	UNIVERSAL AUDIT	ACCESS	VALUE
-	ftp	IBM SUPPLIED			12
-	iccf	IBM SUPPLIED			12
-	lpr	IBM SUPPLIED			12
-	newc	IBM SUPPLIED			12
-	ping	IBM SUPPLIED			12
-	ropc	IBM SUPPLIED			12
-	teln	IBM SUPPLIED			12
-	AADD	IBM SUPPLIED			12
-	ABRW	IBM SUPPLIED			12
-	ACCT	IBM SUPPLIED			12
-	ACEL	IBM SUPPLIED			12
-	ACLG	IBM SUPPLIED			12
PF1=HELP 3=END					
8=FORWARD 9=PRINT					

Figure 3-19 Maintain security profiles - TCICSTRN

Note: Do not delete profiles that are supplied by IBM.

Figure 3-20 shows the BSM Group Maintenance panel, in which all defined groups are listed.

```

IESADMBSLG                                MAINTAIN SECURITY PROFILES
BSM RESOURCE CLASS:  GROUP
START.... GROUP56
OPTIONS:  1 = ADD          2 = CHANGE          5 = DELETE          6 = USER LIST
          8 = CONNECT      9 = REMOVE          USERID              MODEL USERID
          OPT  GROUP NAME  DESCRIPTION          CONNECTED?          CONNECTED?

      -  GROUP56          TRANSEC CLASS MIGRAT          _____          _____
      -  GROUP57          TRANSEC CLASS MIGRAT
      -  GROUP58          TRANSEC CLASS MIGRAT
      -  GROUP59          TRANSEC CLASS MIGRAT
      -  GROUP60          TRANSEC CLASS MIGRAT
      -  GROUP61          TRANSEC CLASS MIGRAT
      -  GROUP62          TRANSEC CLASS MIGRAT
      -  GROUP63          TRANSEC CLASS MIGRAT
      -  GROUP64          TRANSEC CLASS MIGRAT
      -  SALEDEP          SALES DEPT

PF1=HELP          3=END
PF7=BACKWARD      9=PRINT  10=REMOVE ALL

```

Figure 3-20 Maintain security profiles - Class group

Also shown in this window are the system-provided groups GROUPXY (XY=01-64) and the previously defined SALEDEP group.

3.1.8 Migrating BSM profiles

The BSM control file VSE.BSTCNTL.FILE (BSTCNTL) contains information about resource classes. The information is stored in resource profiles, which are used to control access to these resources; for example, TCICSTRN. The transaction profiles in this class control the access to CICS transactions. After initial installation, you must migrate your BSM control file to your target system.

Do *not* migrate the BSTCNTL file by using a VSE/VSAM function, as this process might back-level the system profiles of the new installation. Complete the following steps:

1. On your source system, run the batch program BSTSAVER to build BSTADMIN commands from the contents of the BSM control file. These commands are saved in a librarian member.
2. Move the librarian member to the target system.
3. Run the BSTADMIN program to process the BSTADMIN commands of the librarian member (created by BSTSAVER) and to rebuild the security data space.

You can use skeleton SKBSTSAV in ICCF library 59 to build the jobs that start the BSTSAVER and BSTADMIN programs.

3.1.9 BSM cross-reference reports

In a complex and large user environment that features many user IDs, groups, and profiles, you might need information to ensure that all the different definitions provide a consistent security setup.

Therefore, the BSM provides cross-reference reports with information about the following items:

- ▶ User IDs
- ▶ Groups
- ▶ Access control classes (ACCs) that are used with DTSECTAB
- ▶ Resources that can be accessed by using the UACC definition in the profile
- ▶ Undefined user IDs that are found in groups and access lists of resource profiles

By using these reports, you can manage the profile definitions that are stored in the following locations:

- ▶ VSE control file (IESCNTL)
- ▶ Table DTSECTAB
- ▶ BSM control file (BSTCNTL).

Figure 3-21 shows the input on the IUI dialog BSM Cross Reference report (2-8-5).

IESADMBST	BSM CROSS REFERENCE REPORT
OPTIONS: 1 = REPORT 2 = DETAILED REPORT	
OPT	REPORT NAME
2	Information about user-ID HUGO_____
-	Information about group *_____
2	Information about access control class 6_ (1..32)
-	Information about all user-ID inconsistencies
-	Information about UACC that allow resource access
* = ALL	
PF1=HELP	3=END

Figure 3-21 Input for BSM cross-reference report

Figure 3-22 shows the generated JCL.

// JOB BSTXREF
// EXEC BSTXREF,PARM='USERID=HUGO,L'
/*
// EXEC BSTXREF,PARM='ACC=6'
/*
/&

Figure 3-22 Generated BSTXREF JCL

3.1.10 External security manager installation

ESMs are available from third-party vendors. For more information about vendor products, see [the List of z/VSE Vendors website](#).

ESMs require SEC=YES in your IPL SYS command. By using this command, the DTSECTAB security is activated (as in previous releases of VSE). You might experience initial problems on installing your ESM if it was never run with SEC=YES. The DTSECTAB is controlling your access to VSE files, libraries, sublibraries, and members.

When SEC=YES, the DTSECTAB protects the resources at IPL until the ESM takes over. If a BSM is used, the DTSECTAB stays active. The member DTSECTRC in ICCF library 59 can be used to add the required VSE security access for starting the ESM.

Other SYS parameters that are required for the ESM are the SYS ESM = and SYS SERVPART= parameters. For more information about the ESM initialization phase name, see the ESM installation documentation.

Some ESMs require their own partition in which to run. Because CA-Top Secret requires LE/VSE, the partition must be large enough for LE/VSE to run.

z/VSE V6R1 checks for the ESM setting first. If the parameters are set on the IPL SYS, the ESM is activated; otherwise, the BSM is used. System initialization parameters also must be considered when you are running with an ESM.

Note: Full ESM installation documentation should be acquired from the vendor because IBM cannot provide more information about how to install, administer, or operate a third-party vendor's product.

3.2 CICS Transaction Server for z/VSE 2.1 security

With the removal of CICS internal security, you must decide what level of security your CICS systems require. The following levels of security are available:

- ▶ No security

This level runs the CICS system with no security. If you specify SEC=NO (DFHSIT) as a system initialization parameter, every user can run every transaction and access every resource.

- ▶ Basic security

This level uses the BSM that is supplied with z/VSE Version 6 Release 1. This level provides sign-on, resource security and report security. The BSM functionality is sufficient for most VSE customers.

- ▶ Full-function security

This level uses an ESM. This level provides resource, command, MRO/ISC link, and report security for the CICS Report Control Facility.

3.2.1 CICS TS Sign-on security provided by the BSM

Sign-on security for your CICS regions must be considered as a system-wide security issue, not just a CICS region issue. Only one user profile per user ID is used system-wide. The VSE.CONTROL.FILE contains this profile.

Sign-on security is always performed independently of the IPL SYS SEC= setting except for the RECOVER option (see Figure 3-1 on page 66).

Migrating your System Sign on Table (DFHSNT) from CICS/VSE

The SNT sign-on table macro for defining CICS users and user security is obsolete. Also, the DFHSNT program definition was removed from the group DFHSIGN in DFHLIST.

All security-related information is in the VSE.CONTROL.FILE. Users are now defined by using the z/VSE Interactive Interface. Every user has a user profile that includes the unique user ID and password.

You define users by using the Maintain User Profiles dialog on the z/VSE Interactive Interface. For more information, see “Maintaining Users with the Interactive Interface” on page 84.” For many user updates, the batch utility program IESUPDCF is available. For more information, see “Running batch program IESUPDCF” on page 90”.

If you have multiple DFHSNT tables for different CICS/VSE regions, these tables must be combined and users defined to the VSE.CONTROL.FILE.

Maintaining Users with the Interactive Interface

User profiles can be defined, updates, or deleted by using the Maintain User Profiles dialog on the z/VSE Interactive Interface. To access the dialog, start with VSE/ESA Function Selection panel and select the following choices:

- ▶ 2 (Resource Definition)
- ▶ 1 (User Interface Tailoring)
- ▶ 1 (Maintain User Profiles)

Note: The Fastpath to the “Maintain User Profiles” dialog is 211.

On the Maintain User Profiles panel (see Figure 3-23), enter 1 to add a user or 2 to change a user in the OPT column that is to the left of the user ID.

IESADMUPL2		MAINTAIN USER PROFILES				
VSE CONTROL FILE						
START.... NEWB						
OPTIONS: 1 = ADD 2 = CHANGE 5 = DELETE						
OPT	USERID	PASSWORD VALID UNTIL	REVOKE DATE	USER TYPE	INITIAL NAME	NAME TYPE
—	NEWB			2	IESEPROG	2
—	NEWF	10/30/16 *		1	IESEADM	2
—	OPER			2	IESEOPER	2
—	POST			1	IESA\$FST	1
—	PRODCICS			1	DUMMY	1
1	PROG			2	IESEPROG	2
—	SYSA			1	IESEADM	2
—	VCSRVS			1	DUMMY	1
—	WACK	12/26/16		1	IESEADM	2
—	WACKA	10/30/16 *		1	IESEADM	2
—	WACKPRO			3	IESEADM	2
—	WALK	01/29/17		1	IESEADM	2

PF1=HELP 3=END
PF7=BACKWARD 8=FORWARD 9=PRINT

Figure 3-23 Security Server - Maintain User Profiles

There are four panels that are to set up a user. Panel 1 and Panel 2 are used for defining z/VSE profile information. Panel 3 and Panel 4 are used for the CICS TS profile information. Use PF8 to scroll through these panels.

Panel 1 (Add or Change User Profile) is shown in Figure 3-24.

IESADMUPBA		ADD OR CHANGE USER PROFILE	
Base	II	CICS	ResClass ICCF
To ADD, specify information for all of the entries.			
USERID.....	TESTID__	4 - 8 characters (4 characters for ICCF users) .1.	
INITIAL PASSWORD...	_____	3 - 8 characters .2.	
DAYS.....	000	0-365 Number of days before password expires	
REVOKE DATE.....	_____	Date when Userid will be revoked (mm/dd/yy) .3.	
USER TYPE.....	2	1=Administrator, 2=Programmer, 3=General	
AUDITOR.....	2	1=yes, 2=no	
INITIAL NAME.....	IESEPROG	Initial function performed at signon	
NAME TYPE.....	2	1=Application, 2=Selection Panel	
SYNONYM MODEL.....		Userid to be used as model for synonyms	
PROGRAMMER NAME....		Supplementary user name	
PF1=HELP		3=END	5=UPDATE
8=FORWARD			

Figure 3-24 Add or Change User Profile panel

The following settings apply to the fields that are highlighted by numbers in Figure 3-24:

1. Only eight characters are available for the USERID field, no matter which sign-on procedure is used (z/VSE Interactive Interface or CESN). ICCF did not change. An ICCF user can have only a 4-character user ID.
2. This password must be changed by the user at initial sign-on.
3. After this date, a sign-on attempt is rejected. This field can be used for temporary users. The BSM also revokes a user ID after the number of sign-on attempts exceeds 5. This value is specified in IESELOGO setup. A sample job is found in ICCF library 59.

The second panel (User Authentication) is shown in Figure 3-25. In this panel, the user's authorization to perform certain tasks is set.

IESADMUPII		USER AUTHORIZATION	
Base	II	CICS	ResClass ICCF
Answer yes or no to the following questions for userid TESTID			
Enter 1 for yes, 2 for no			
NEWS.....	1	Should user receive news items?	
ESCAPE.....	2	Can user escape to CICS?	
CONFIRM DELETE.....	2	Does user want a confirmation message?	
VSE PRIMARY SUBLIBRARY.....	1	Does user want a PRIMARY sublibrary?	
SUBMIT TO BATCH.....	1	Can user submit to Batch?	
VSAM FILES.....	1	Can user define VSAM files?	
VSAM CATALOGS.....	2	Can user manage VSAM catalogs?	
OLPD.....	1	Can user delete OLPD incidents?	
CONSOLE COMMANDS.....	1	Can user enter all commands?	
CONSOLE OUTPUT.....	1	Can user see all messages?	
BATCH QUEUES.....	2	Can user manage all POWER jobs?	
DEFAULT USER VSAM CATALOG.. IJSYSCT			
PF1=HELP		3=END	5=UPDATE
PF7=BACKWARD		8=FORWARD	

Figure 3-25 User Authentication panel

The third panel (Add or Change CICS Segment) is shown in Figure 3-26 and the fourth panel (Add or Change Resource Access Rights) is shown in Figure 3-27 on page 89. These panels are used for CICS users.

IESADMUPCI		ADD OR CHANGE CICS SEGMENT						
Base	II	CICS	ResClass	ICCF				
OPERATOR ID.....		PRO	Enter 3 character id for user TESTID					
OPERATOR PRIORITY.....		000	Operator priority 0 - 255					
XRF SIGNOFF.....		2	Sign off after XRF takeover (1=yes,2=no)					
TIMEOUT.....		00	Minutes until sign off 0 - 60					
PRIMARY LANGUAGE.....		_	National language for CICS messages					
Place an 'X' next to the operator classes for this user								
01 X	02 _	03 X	04 _	05 _	06 _	07 _	08 _	
09 _	10 _	11 _	12 _	13 _	14 _	15 _	16 _	
17 _	18 _	19 _	20 _	21 _	22 _	23 _	24 _	
PF1=HELP			3=END			5=UPDATE		
PF7=BACKWARD		8=FORWARD						

Figure 3-26 Add or Change CICS Segment panel

The operator classes define the user to CICS TS (1 is the default class), as shown in Figure 3-27.

IESADMUPR1		ADD OR CHANGE RESOURCE ACCESS RIGHTS									
Base	II	CICS	ResClass ICCF								
Userid: TESTID											
Specify the access rights for 1-32 DTSECTAB access control classes (_=No access, 1=Connect, 2=Read, 3=Update, 4=Alter)											
01 _	02 _	03 _	04 _	05 _	06 _	07 _	08 _	09 _	10 _	11 _	
12 _	13 _	14 _	15 _	16 _	17 _	18 _	19 _	20 _	21 _	22 _	
23 _	24 _	25 _	26 _	27 _	28 _	29 _	30 _	31 _	32 _		
READ DIRECTORY..... 1 User can read directory with Connect (1=yes, 2=no) B-TRANSIENTS..... 2 User can manipulate B-Transients (1=yes, 2=no)											
PF1=HELP				3=END				5=UPDATE			
PF7=BACKWARD		8=FORWARD									

Figure 3-27 Add or Change Resource Access Rights panel

These panels have not changed from z/VSE 5.1.

In previous VSE releases, transaction security keys were specified on this panel (IESADMUPR1). User access to transactions is done now by adding the user ID to the transaction profile access list.

When finished, press PF5 to update.

Adding a type 2 or 3 user enablement for the BSM resource profiles can be done by adding the user to existing groups. As shown in Figure 3-28, it is good practice to press PF6 and add the user to the same group as the selected model. When finished in panel IESADMBSLG, press PF3 to complete the action.

IESADMBSLG

MAINTAIN SECURITY PROFILES

BSM RESOURCE CLASS: GROUP

START....

OPTIONS:

1 = ADD

2 = CHANGE

5 = DELETE

6 = USER LIST

8 = CONNECT

9 = REMOVE

USERID

MODEL USERID

CONNECTED?

CONNECTED?

TESTID

PROG

OPT	GROUP NAME	DESCRIPTION	TESTID	PROG
-	GROUP01	TRANSEC CLASS MIGRAT		M
-	GROUP02	TRANSEC CLASS MIGRAT		
-	GROUP03	TRANSEC CLASS MIGRAT		
-	GROUP04	TRANSEC CLASS MIGRAT		
-	GROUP05	TRANSEC CLASS MIGRAT		
-	GROUP06	TRANSEC CLASS MIGRAT		
-	GROUP07	TRANSEC CLASS MIGRAT		
-	GROUP08	TRANSEC CLASS MIGRAT		
-	GROUP09	TRANSEC CLASS MIGRAT		
-	GROUP10	TRANSEC CLASS MIGRAT		
-	GROUP11	TRANSEC CLASS MIGRAT		
-	GROUP12	TRANSEC CLASS MIGRAT		

PF1=HELP

3=END

6=CONNECT MODEL

8=FORWARD

9=PRINT

10=REMOVE ALL

CHANGE THE SECURITY PROFILE OF USERID ACCORDING TO THE MODEL PROG .

Figure 3-28 Adding type 2 or 3 user to BSM group

In VSE, type 1 users are similar to Linux or UNIX “root” users that can access resources without having specific authorization.

The following message displays, which indicates that the profile was added to the VSE.CONTROL.FILE:

USER PROFILE INFORMATION HAS BEEN UPDATED.

Running batch program IESUPDCF

When migrating users from the DFHSNT table, a batch program can be used. A parameter card defines the user ID, password, and profile. If you set up a common profile for a group of users, the other fields are copied from the profile. A sample of this job is shown in Figure 3-29 on page 91 and provided in the ICCF library 59.

```

* $$ JOB JNM=IESUPDCF,CLASS=0,DISP=D
* $$ PUN DISP=I,CLASS=0,PRI=9
// JOB IESUPDCF
// OPTION NOLOG
*
* THIS SKELETON MAY BE USED BY THE ADMINISTRATOR TO GENERATE A
* JOB FOR BATCH USER PROFILE MAINTENANCE.
* 1. IF THE CONTROL FILE BELONGS TO A CICS WITHOUT ICCF AND THIS
* CICS DOES NOT SHARE THE CONTROL FILE WITH CICS/ICCF,
* ADJUST THE '// DLBL' STATEMENT TO MAINTAIN
* USER PROFILES IN THE RELATED CONTROL FILE.
* 2. SUPPLY AN OPERAND FOR THE ICCF PARAMETER, VALID OPERANDS ARE:
* Yes ... UPDATE USER PROFILES IN CONTROL FILE (CICS) AND
* IN THE DSTFILE (ICCF).
* No ... UPDATE USER PROFILES IN CONTROL FILE ONLY.
* INHIBIT CHANGES TO ICCF RELATED INFORMATION.
* Ignore ... UPDATE USER PROFILES IN CONTROL FILE ONLY.
* THIS VALUE MUST BE USED IF THE CONTROL FILE
* IS USED IN CICS SUBSYSTEMS WITHOUT ICCF ONLY.
* 3. INSERT THE ADD, ALTER AND DELETE STATEMENTS THAT YOU NEED TO
* MAINTAIN USER PROFILES.
* SAMPLE STATEMENTS:
* =====
* * TEXT ... A COMMENT LINE
* Add USERID,PASSWD,PROFILE(,OPTIONAL PARAMETERS)
* Alter USERID(,OPTIONAL PARAMETERS)
* Delete USERID
* EXPLANATION OF PARAMETERS:
* =====
* 1. REQUIRED AND POSITIONAL PARAMETERS:
* -----
* USERID ... THE ID OF THE USER ( ADD, ALTER, DELETE )
* ( 4-8 CHARACTER / 4 CHARACTER FOR ICCF USER )
* PASSWD ... THE PASSWORD OF THE USER ( ADD )
* ( 3-8 CHARACTERS )
* PROFILE ... THE ID OF THE USER USED AS PROFILE FOR
* THE NEW USER ( ADD )
* ( 4-8 CHARACTER / 4 CHARACTER FOR ICCF USER )
* 2. OPTIONAL PARAMETERS IN ADD/ALTER STATEMENT:
* -----
* Catalog= ... THE DEFAULT CATALOG OF THE USER
* EXAMPLE: CAT=VSESPUC
* Days= ... NUMBER OF DAYS IN EXPIRATION INTERVAL
* EXAMPLE: DAYS=20 ( RANGE: 0-365)
* Library= ... Primary ICCF library ( only ICCF users )
* EXAMPLE: LIB=20
* Initial= ... Initial function at SIGNON
* EXAMPLE: INIT=APPLNAME(A) ... FOR APPLICATION
* INIT=SELNAME(S) ... FOR SELECTION P.

```

Figure 3-29 Sample job IESUPDCF (Part 1 of 4)

Figure 3-30 shows Part 2 of the sample job IESUPDCF.

```

*      Natlang=    ... NATIONALLANGUAGE_INDICATOR
*                  EXAMPLE: NAT=E ( for english )
*
*      PGMID=      ... Programmer ID
*                  EXAMPLE: PGMID=HENRY_FORD (max. 20 characters )
*
*      Operator=   OPERATOR ID
*                  EXAMPLE: OPER=OPE
*
*      PWD=        ... USER PASSWORD
*
*      PAssword=   EXAMPLE: PWD=PASSWD ( 3 - 8 Characters )
*
*      Priority=    ... OPERATOR PRIORITY
*                  EXAMPLE: PRIOR=5 ( RANGE: 0-255 )
*
*      Revoke=     ...REVOKE_DATE
*                  EXAMPLE: R=01/31/97 ( Format mm/dd/yy )
*
*      Synonym=    ... SYNONYMS MODEL
*                  EXAMPLE: SYNONYM=SYNS ( 4 - 8 Characters )
*
*      Timeout=    ... TIMEOUT INTERVAL
*                  EXAMPLE: TIME=20 ( Values: 0,5,10,...,60 )
*
*      APM=Yes|No  ... APPLICATION PROFILE MAINTENANCE
*
*      AUD=Yes|No  ... USER IS AUDITOR
*
*      BQA=Yes|No  ... MANAGE BATCH QUEUES
*
*      CMD=Yes|No  ... ENTER ALL CONSOLE COMMANDS
*
*      COU=Yes|No  ... FULL OUTPUT ON SYSTEM SONSOLE
*
*      COD=Yes|No  ... CONFIRM ON DELETE
*
*      DVF=Yes|No  ... DEFINE VSAM FILES
*
*      ESC=Yes|No  ... ESCAPE TO CICS
*
*      MVC=Yes|No  ... MANAGE VSAM CATALOGS
*
*      NEWS=Yes|No ... DISPLAY NEWS TO USER
*
*      OLPD=Yes|No ... DELETE OLPD INCIDENTS
*
*      PSL=Yes|No  ... OWNS A PRIVATE SUBLIBRARY
*
*      BAT=Yes|No  ... SUBMIT TO BATCH
*
*      SPM=Yes|No  ... SELECTION PANEL MAINTENANCE
*
*      UPM=Yes|No  ... USER PROFILE MAINTENANCE
*
*      XRF=Yes|No  ... XRF SIGNOFF
*
*      -          ... CONTINUATION CHARACTER
*
*      4. DELETE BLOCK 'UPDPL', IF YOU DO NOT WANT TO MAINTAIN
*          THE 'PRIMARY' LIBRARY.
*
* *****
*
*      IESCNTRL MUST BE CLOSED IF UPDATES ARE DONE. PERFORM
*      CEMT SET FILE(IESCNTRL) CLOSE      IN EACH CICS WITH THE
*
*                                          INTERACTIVE INTERFACE ACTIVE
*
*      MSG FB,DATA=CLOSECNTRL             TO CLOSE THE FILE IN BSM
*

```

Figure 3-30 Sample job IESUPDCF (Part 2 of 4)

Figure 3-31 shows Part 3 of the sample job IESUPDCF.

```

* IMPORTANT:
*   IF NEW USERS ARE ADDED, IN ORDER TO DEFINE THIS USERS
*   TO THE BSTCNTL BASED SECURITY, YOU HAVE TO PRESS PF6 (GROUPS)
*   ON PANEL 'USER PROFILE MAINTENANCE' FASTPATH 211.
*   THE USERS CAN ALSO BE ADDED USING BSTADMIN:
*       // EXEC BSTADMIN
*       CONNECT GROUPxx  user
*   /*
*   IF USERS ARE DELETED, THIS USERS SHOULD BE REMOVED FROM THE
*   GROUP:
*       // EXEC BSTADMIN
*       REMOVE GROUPxx  user
*   /*
*
* *****
*
* ==> UPDATE NEXT LINE IF NECESSARY (SEE 1.)
// DLBL IESCNTL,'VSE.CONTROL.FILE',,VSAM,CAT=VSESPUC
*
// EXEC PROC=DTRICCF
// EXEC IESUPDCF,SIZE=64K
*
* ==> SUPPLY AN OPERAND FOR THE ICCF PARAMETER (SEE 2.)
ICCF=
*
* ==> INSERT STATEMENTS HERE (NO COMMENT '*' IN FIRST COLUMN, SEE 3.)

/*
// IF $RC=0 THEN
// GOTO STEP2
// IF $RC=4 THEN
// GOTO ERROR
// IF $RC>6 THEN
// GOTO END
// LOG
* ==> JOB 'DTRUPD' CREATED, ENSURE THAT THIS JOB IS EXECUTED NEXT
// NOLOG
// IF $RC=2 THEN
// GOTO STEP2
/. ERROR
// LOG
* ==> ERRORS IN INPUT DATA, STATEMENT(S) FLAGGED IN LISTING
// NOLOG
/. STEP2
*
* ==> DELETE BLOCK 'UPDPL', IF REQUIRED (SEE 4.)
* ***** BEGIN OF BLOCK 'UPDPL' *****
// EXEC PROC=IESUPDPL
/*
* ***** END OF BLOCK 'UPDPL' *****

```

Figure 3-31 Sample job IESUPDCF (Part 3 of 4)

Figure 3-32 shows Part 4 of the sample job IESUPDCF.

```
/. END
* *****
* IESCNTRL CAN BE REOPENED:
*   CEMT SET FILE(IESCNTRL) OPEN      IN EACH CICS
*   MSG FB,DATA=OPENCNTRL             TO OPEN THE FILE IN BSM
* *****
/&
* $$ E0J
```

Figure 3-32 Sample job IESUPDCF (Part 4 of 4)

Now, you can perform the following actions:

- ▶ Apply data to add user1 with password usr1 and profile prof1.
- ▶ Set optional parameters, such as operator ID, that are entered after the profile name.

3.2.2 User sign-on and sign-off

Users sign on to CICS TS by using the z/VSE Interactive Interface sign-on dialog (see Figure 3-33 on page 95) or the CICS CESN sign-on transaction (see Figure 3-34 on page 95).

The VSE.CONTROL.FILE is checked by the Security Server or the information that is passed on to the ESM. CSSN and CSSF transactions are obsolete. CSSF can still be issued by an application program by issuing the following command:

```
EXEC CICS START CSSF
```

There is a new system initialization parameter SNSCOPE= that is used to specify whether a user ID can be signed on to CICS more than once within a CICS region or a VSE system.

Figure 3-33 shows the z/VSE Interactive Interface sign-on panel.

```

IESADMS01                                z/VSE ONLINE
5609-VSE and Other Materials (C) Copyright IBM Corp. 2015 and other dates

                                ++
                                ++  VV  VV  SSSSS  EEEEEEE
                                ++  VV  VV  SSSSSS  EEEEEEE
zzzzzz      ++  VV  VV  SS  EE
zzzzz      ++  VV  VV  SSSSS  EEEEEEE
zz      ++  VV  VV  SSSSS  EEEEEEE
zz      ++  VV  VV  SS  EE
zzzzzz      ++  VVVV  SSSSSS  EEEEEEE
zzzzzz      ++  VV  SSSSS  EEEEEEE

Your terminal is A000 and its name in the network is D0600001
Today is 11/03/2016 To sign on to DBDCCICS -- enter your:

USER-ID..... _____ The name by which the system knows you.
PASSWORD..... Your personal access code.

PF1=HELP      2=TUTORIAL  3=TO VM      4=REMOTE APPLICATIONS
                                10=NEW PASSWORD

```

Figure 3-33 z/VSE Interactive Interface sign-on panel

Figure 3-34 shows the CICS CESN sign-on transaction.

```

Type your userid and password, then press ENTER:

  Userid . . . . 1.
  Password . . .
  Groupid . . . 2.
  Language . . .

  New Password . . .

DFHCE3520 Please type your userid.
F3=Exit

```

Figure 3-34 CICS CESN sign-on transaction

The following considerations apply:

- ▶ Only eight characters are available for the Userid field.
- ▶ Groupid applies only to a full-function ESM, not to the VSE BSM.

3.2.3 User password checking

The first time that users sign onto the system after being defined to the VSE.CONTROL.FILE, they are prompted for a new password (see Figure 3-35).

IESADMS02	z/VSE SIGN-ON WITH NEW PASSWORD
Enter your new password in both places below then enter your current password for sign-on verification. Then press the ENTER key.	
NEW PASSWORD ==>	3 - 8 characters
NEW PASSWORD ==>	Re-Enter new password for verification
OLD PASSWORD ==>	Current password
PF1=HELP	3=END
YOUR PASSWORD HAS EXPIRED. SPECIFY A NEW TO SIGN-ON TO z/VSE.	

Figure 3-35 Password change panel when signing onto z/VSE Interactive Interface

3.2.4 CICS TS security parameters

Security parameters can be defined within CICS in the following locations:

- The system initialization table DFHSIT

For more information about the new security system initialization parameters, see *CICS Transaction Server for VSE/ESA System Definition Guide*.

- Transaction definitions

For more information about the new Transaction security definitions, see *CICS Transaction Server for VSE/ESA Resource Definition Guide*.

- Link security

For more information about the new Link security options, see *CICS Transaction Server for VSE/ESA Resource Definition Guide*.

3.2.5 CICS TS transaction-attach security

Transaction security is part of the BSM. It is always active, regardless of the security setting in the IPL SYS SEC parameter. It is not active if SEC=NO is specified as a system initialization parameter.

For each transaction, a BSM security profile of class TCICSTRN must be defined. You can define the required profiles by using the batch utility BSTADMIN or through the IUI security dialogs, as described in 3.1.7, “Maintaining BSM security profiles” on page 77.

When new CICS products are installed, their transactions must be defined to the CICS TS CSD file and to the BSM. The user must be enabled for a transaction by specifying the user ID on the transaction profile access list or adding the user ID to a group that is enabled for the transaction. Unauthorized access attempts are logged to the CICS console file and the VSE console log. For more information, see 3.1.5, “BSM logging and reporting” on page 75”.

Note: A transaction for which no security profile exists cannot be accessed. All transactions must be access protected and in the BSTCNTL file.

Multiple CICS regions

If more than one CICS TS runs under z/VSE V6R1, a CICS region ID is used to identify the CICS system to the transaction security definition. This prefix is the ID statement in the CICS startup JCL.

The new system initialization parameter SECPRFX=YES must be used for the CICS system to be checked. Also, IPL SYS SEC=YES must be specified. If your z/VSE system includes IPL SYS SEC=NO, do not specify a CICS region on your transaction security definitions. Only transactions with no CICS Region specified will be checked. For more information about setting up multiple CICS TS systems for transaction security, see Figure 3-37 on page 98.

Maintaining Transaction profiles using the z/VSE Interactive Interface

From the z/VSE Function Selection panel, select the following values:

- ▶ **2** for Resource Definition
- ▶ **8** for Security Maintenance
- ▶ **6** for Unified BSM Resource Profile Maintenance

The Fastpath to the Define Transaction Security panel is 286

Select BSM class TCICSTRN by the specifying option **6=Profile List**, as shown in Figure 3-36.

IESADMBSLC		BSM RESOURCE CLASSES	
START.... MQQUEUE			
OPTIONS:		6=PROFILE LIST	
OPT	RESOURCE CLASS NAME	STATUS	
—	MQQUEUE	INACTIVE	
—	MXTOPIC	INACTIVE	
—	SCICSTST	ACTIVE	
—	SURROGAT	ACTIVE	
6	TCICSTRN	ACTIVE	
PF1=HELP		2=REFRESH	3=END
PF7=BACKWARD		9=PRINT	

Figure 3-36 Select resource class TCICSTRN

Figure 3-37 shows the list of BSM profiles for class TCICSTRN.

IESADMBSLE		MAINTAIN SECURITY PROFILES	
BSM RESOURCE CLASS: TCICSTRN		(START is Case Sensitive)	STATUS: ACTIVE
START....			
OPTIONS: 1 = ADD 2 = CHANGE 5 = DELETE 6 = ACCESS LIST			
OPT	PROFILE NAME	DESCRIPTION	UNIVERSAL AUDIT ACCESS VALUE
-	ftp	IBM SUPPLIED	12
-	iccf	IBM SUPPLIED	12
-	lpr	IBM SUPPLIED	12
-	newc	IBM SUPPLIED	12
-	ping	IBM SUPPLIED	12
-	ropc	IBM SUPPLIED	12
-	teln	IBM SUPPLIED	12
-	AADD	IBM SUPPLIED	12
-	ABRW	IBM SUPPLIED	12
-	ACCT	IBM SUPPLIED	12
-	ACEL	IBM SUPPLIED	12
-	ACLG	IBM SUPPLIED	12
PF1=HELP		3=END	
		8=FORWARD	9=PRINT

Figure 3-37 List BSM profiles for class TCICSTRN

Figure 3-38 shows the ADD transaction profile panel.

IESADMBSAE		MAINTAIN SECURITY PROFILES	
BSM RESOURCE CLASS:		TCICSTRN	
Add Profile:			
PREFIX.....	_____	CICS region	
RESOURCE NAME.....		Maximum length is 4 characters.	
..... TST2			
GENERIC.....	2	(1=yes, 2=no)	
UNIVERSAL ACCESS...	_	(_=None, 2=Read, 3=Update, 4=Alter)	
AUDIT-LEVEL 1	1	(_=None, 1=Failure, 2=Success, 3=All)	
ACCESS-LEVEL 1	2	(2=Read, 3=Update, 4=Alter, _=default)	
AUDIT-LEVEL 2		(_=None, 1=Failure, 2=Success, 3=All)	
ACCESS-LEVEL 2		(2=Read, 3=Update, 4=Alter, _=default)	
DESCRIPTION.....	MY PRODUCT PACK1	Optional remark	
PF1=HELP	3=END	5=UPDATE	
RESOURCE NAME FIELD IS CASE SENSITIVE. ENTER DATA AS REQUIRED.			

Figure 3-38 Add transaction profile panel

CICS Region (Prefix) is brought forward from the previous dialog where the ADD was selected. When finished, press PF5=UPDATE, which saves the transaction profile directly in the BSTCNTL file. No extra catalog step is required to make the change permanent.

After adding, changing, or deleting transactions, press PF5 while you are in the Define Transaction Security panel to PROCESS.

Maintaining transactions using BSTADMIN utility

Alternatively, you can use batch utility BSTADMIN to maintain the BSM profiles. For more information, see “Batch utility BSTADMIN” on page 77.

Activating transaction security changes

After changing any security definitions, the CICS region must reflect the change. This action must be done with the CEMT CICS-supplied transaction by using the following form:

CEMT PERFORM SECURITY or CEMT P SE

The new CICS transaction table is made available to the CICS region where the command was entered.

3.2.6 CICS default user ID

All CICS TS tasks feature an associated user ID. If a task is started that does not have an associated user ID, the default user ID from the system initialization parameter DFLTUSER= is used.

You must define a default CICS user ID to your security manager. If an ESM is used, the system initialization parameters SEC=YES and DFLTUSER=name must be specified. The default user ID name must also be defined to the ESM. If no default CICS user ID is specified, CICSUSER is the default ID.

CICS uses the security attributes of the default user ID to perform all security checks for any users that do not sign-on by using the CESN sign-on transaction.

These security checks that use the default user ID apply to transaction-attach security checking. Command security checks are not supported by the BSM.

Note: Because this default user ID is universally available, limit the resources that you intend to make available to the default ID.

On CICS initialization, the default user ID is signed-on, as shown in Figure 3-39.

```
F2 0002 DFHTR0103 TRACE TABLE SIZE IS 4096K
F2 0002 DFHSM0122I DBDCCICS Limit of DSA storage below 16MB is 5,120K.
F2 0002 DFHSM0123I DBDCCICS Limit of DSA storage above 16MB is 200M.
F2 0002 DFHSM0115I DBDCCICS Storage protection is active.
F2 0167 DFHDM0101I DBDCCICS CICS is initializing.
F2 0168 DFHXS1100I DBDCCICS Security initialization has started.
F2 0168 DFHXB0109I DBDCCICS Web domain initialization has started.
F2 0168 DFHS00100I DBDCCICS Sockets domain initialization has started.
F2 0168 DFHDH0100I DBDCCICS Document domain initialization has started.
F2 0168 DFHSI1500 DBDCCICS CICS startup is in progress for CICS Transaction
Server Version 2.1.0
F2 0168 DFHSI1501I DBDCCICS Loading CICS nucleus.
F2 0168 DFHXS1105 DBDCCICS Resource profiles for class TCICSTRN have been
built.
F2 0168 DFHXS1105 DBDCCICS Resource profiles for class SURROGAT have been
built.
F2 0168 DFHXS1103I DBDCCICS Default security for userid CICSUSER has been
established. .I.
F2 0168 DFHXS1101I DBDCCICS Security initialization has ended.
```

Figure 3-39 Initialization of CICS TS with default user ID of CICSUSER

The following parameters apply to our CICS TS initialization:

► Default user ID of CICSUSER

If the default user ID is changed, this default user ID must be defined to the BSM in VSE.CONTROL.FILE. If the default user ID is not found, CICS initialization ends (see Figure 3-40).

```
F8 0180 DFHXS1105 PRODCICS Resource profiles for class TCICSTRN have been
      built.
F8 0179 IRR012I  VERIFICATION FAILED. USER PROFILE NOT FOUND.  .0.
F8 0180 DFHXS1104 PRODCICS
      Default security could not be established for userid CICSPROU. The
      security domain cannot continue, so CICS is terminated. SAF codes are
      (X'00000004',X'00000000'). ESM codes are (X'00000004',X'00000000').
F8 0008 DFHKE1800 PRODCICS ABNORMAL TERMINATION OF CICS IS COMPLETE.
```

Figure 3-40 CICS startup failure when default user ID not defined

► During this startup, a non-existing user ID was specified

CICSPROU ID was defined as the default user ID but was not defined in VSE.CONTROL.FILE.

Figure 3-41 shows an example of defining the default user ID to the BSM.

```
IESADMUPBA          ADD OR CHANGE USER PROFILE
Base   II          CICS   ResClass ICCF
To ADD, specify information for all of the entries.

  USERID..... CICSPROU   4 - 8 characters (4 characters for ICCF
users)

  INITIAL PASSWORD... _____ 3 - 8 characters

  DAYS..... 000          0-365 Number of days before password expires
  REVOKE DATE..... _____ Date when Userid will be revoked (mm/dd/yy)

  USER TYPE..... 3        1=Administrator, 2=Programmer, 3=General
  AUDITOR..... 2          1=yes, 2=no
  INITIAL NAME..... DFLESEL Initial function performed at signon
  NAME TYPE..... 2         1=Application, 2=Selection Panel
  SYNONYM MODEL.....       Userid to be used as model for synonyms
  PROGRAMMER NAME.....       Supplementary user name

PF1=HELP              3=END              5=UPDATE
                      8=FORWARD
```

Figure 3-41 Definition of the new default user ID

Using the default user ID

Some resources include other keywords to specify a user ID.; for example, the DCT. In the DFHDCT TYPE=INITIAL macro, you can specify the new USERID parameter that is for trigger-level transactions that are not associated with a terminal. CICS uses this user ID for security checking. If omitted, the CICS default user ID that is specified on the system initialization parameter DFLTUSER is used. If you migrate destination control tables that do not specify a user ID for trigger-level transactions, CICS TS issues the transaction-attach security check against the CICS default user ID.

The **EXEC CICS START** command has a new user ID parameter, USERID. This parameter allows the transaction to run with the authority of the USERID specified.

The default USERID is also specified for multiregion security. For more information, see “Multiregion operation security not supported by the BSM” on page 103.

3.2.7 Security for program list table programs at startup

A new system initialization parameter that is named PLTPIUSR is available that specifies the user ID that CICS is to use for security checking for program list table (PLT) programs that run during CICS initialization. By using the new system initialization parameter PLTPISEC, you can specify the type of security checking to be performed (none, command, resource, or all). The PLT programs run under a new CICS-supplied transaction, CPLT.

3.2.8 Resource security checking

Another level of transaction security can be implemented by controlling access to the resources that the CICS transaction uses. This resource checking can be specified at the individual transaction level. The system initialization parameter that is required for transaction resource level checking is RESSEC=.

The transaction definition requires the RESSEC= parameter set (the RESSEC parameter is called RSLC in previous releases of CICS. RSLC(EXTERNAL) can be specified in compatibility mode only.

Other system initialization parameters and transaction definition parameters are required for full CICS transaction resource checking.

3.2.9 Command security checking not supported by BSM

You can use CICS command security to check system programming interface (SPI) commands that are issued from CICS application programs, and the equivalent commands that can be issued from CEMT commands. Command security operates in addition to any transaction or resource security that might be defined for a transaction. The commands that are subject to command security checking are the following commands that require the SP translator option:

- ▶ ACQUIRE
- ▶ COLLECT
- ▶ CREATE
- ▶ DISABLE and ENABLE
- ▶ DISCARD
- ▶ EXTRACT EXIT
- ▶ INQUIRE
- ▶ PERFORM

- ▶ RESYNC
- ▶ SET

3.2.10 Surrogate user security

Surrogate users allow users to start work on behalf of another user. The system initialization parameter XUSER=NOIYES specifies whether CICS is to perform surrogate user checking where such a check is permitted.

CICS uses the BSM or ESM to verify the authority of one user (the surrogate user) to submit a transaction on behalf of another user.

3.2.11 Security on intercommunication

If you have intercommunication definitions for Multiregion operation (MRO), Intersystem Communication (ISC), or Advanced Program-to-Program Communication (APPC) under CICS/VSE 2.3, you must review the CONNECTION and SESSION parameters in your CSD or your SYSTEM definitions in your DFHTCT.

During IPL of z/VSE 6.1, the intercommunication program DFHIRP is loaded into the SVA and is used for any communication session irrespective of the CICS level.

APPC (LU6.2) session security

This section provides recommendations for migrating from CICS/VSE 2.3 to CICS TS.

If you have an APPC connection and you defined the connection with a BINDPASSWORD, you must decide whether to continue using CICS bind security. The method you use for bind security on LU6.2 connections depends on the capabilities of your ESM.

If your ESM supports bind-time session security, you can use the facilities that are provided by the APPCLU general resource class and specify XAPPC=YES as a system initialization parameter. Defining BINDSECURITY=YES on your CONNECTION resource definition indicates that you want bind-time security checking on your connection.

If your ESM does not support bind-time security (for example, the BSM), you can continue to use the BINDPASSWORD option on the CONNECTION resource definition. You must specify XAPPC=NO as a system initialization parameter and BINDSECURITY=NO on the CONNECTION resource definition. CICS then uses the BINDPASSWORD.

Multiregion operation security not supported by the BSM

This section provides recommendations for migrating from CICS/VSE 2.3 to CICS TS.

If you have MRO connections and you coded the SECURITYNAME parameter on the CONNECTION definition in the CSD (or as part of your SYSTEM definition in a DFHTCT), be aware that this parameter is no longer used for bind-time security checking. It also is not used for any other security purpose on MRO links. The internal security check mechanisms that were performed on previous releases were replaced by calls to the ESM by using the SAF interface.

If you coded the OPERSECURITY or OPERRSL parameters on the SESSION definition, these parameters are obsolete under the CICS TS. Under CICS TS, the method of supplying a user ID for link security is the USERID parameter of the SESSIONS definition. If you do not specify a USERID in the SESSIONS definition, CICS uses the connecting region's user ID as the default link user ID.

To establish whether you want link security checking or it is to be bypassed, CICS compares its own region ID with the ID that is being passed in the USERID parameter of the SESSION definition, or the user ID of the connecting region if it was not supplied in the SESSION definition.

Consider the following basic rules:

- ▶ If the user IDs are the same, link security checking is bypassed.
- ▶ If the user IDs are different, CICS signs-on the link user ID as passed on the SESSION definition and uses this ID for all link security checks.

When link security is bypassed, security checking is determined by the ATTACHSEC parameter on the CONNECTION definition.

The relationship between ATTACHSEC and link security is listed in Table 3-2.

Table 3-2 MRO security options

Link security	ATTACHSEC (LOCAL)	ATTACHSEC (IDENTIFY)
Yes	CICS ignores the user ID that is associated with the transaction and issues all security checks against the user ID only.	CICS uses two security checks: One against the user ID associated with the transaction, and one against the link user ID. These checks ensure that the transaction cannot access resources that are not authorized for the link.
Bypassed	CICS issues all security checks against the CICS default user ID (the user ID specified on the DFLTUSER system initialization parameter).	CICS issues security checks against the user ID that is associated with the transaction.

3.2.12 CICS Report Controller

CICS TS offers report security and printer security. The BSM supports report security requirements since z/VSE 3.1.1. An ESM is not required.

CICS TS no longer provides internal security facilities; therefore, the authorization request for reports is passed by RACROUTE calls through the z/VSE SAF to the BSM or ESM.

To provide report controller security in an environment that is managed by external security, the report controller requires the following sets of resource names to be defined to the BSM or ESM:

- ▶ For REPORT security: DFHRCF.RSL01 through DFHRCF.RSL24 and DFHRCF.RSLPU
- ▶ For report BROWSE security: DFHRCF.BRSL01 through DFHRCF.BRSL24 and DFHRCF.BRSLPU

In each case, 01-24 and PU (public) correspond to an RSL specification on the SPOOLOPEN command. The BSM provides these profiles within the FACILITY class. Allow the users access by specifying them on the ACCESS LIST for subject profile.

For more information about security, see *CICS Transaction Server for z/VSE Security Guide* and *CICS Transaction Server for z/VSE Report Controller Planning Guide*.

3.2.13 Printer security

Printer security is also handled with RACROUTE calls to BSM or an ESM. A user ID is associated with the printer and used by the ESM. For printer security, the profiles DFHRCF.PRSL01 through DFHRCF.PRSL24 and DFHRCF.PRSLPU are used.

3.2.14 Terminal security

There are no OPERRSL or OPERSEC parameters in terminal definitions. There is a preset user ID used by the ESM. Every CICS task features an associated user ID.

3.3 Security Migration Aid

This section provides recommendations for migrating from CICS/VSE 2.3 to CICS TS.

The Security Migration Aid (SMA) utility is available to assist in the migration of CICS/VSE internal security to the z/VSE supplied BSM or a fully functional ESM.

This utility helps you convert your CICS internal security definitions into a format acceptable to a security manager. It is a two-phase utility that gathers data about your source system and processes the data about the target system to generate the proper format.

For more information about the SMA, see *CICS Transaction Server for VSE/ESA Security Guide*, SC33-1942.

The SMA utility is no longer included with CICS TS for z/VSE 2.1.

3.4 Summary

The BSM that is supplied with z/VSE Version 6 Release 1 provides considerable security functions for your CICS Transaction Server for z/VSE 2.1. For full system security, a vendor-supplied ESM should be used.

Consider the following points when you are reviewing the security on your system when migrating to the CICS Transaction Server for z/VSE 2.1 by using the BSM:

- ▶ Move your users that are defined in the DFHSNT table (CICS/VSE customers) or the DTSECTAB table to the VSE.CONTROL.FILE.
- ▶ BSM implements access checking by mapping resources to a profile. A user is granted access to the resource by specifying the user ID on the access list.
- ▶ Review the security on the CICS-supplied transactions and modify them, as required.
- ▶ Check any vendor or in-house programs that were using the DFHSEC macro.
- ▶ Check your use and definition of the Destination Control Table (DCT) for the default user ID checking.
- ▶ Check your use of the programs in the PLTPI for default user ID checking.
- ▶ Introduce your users to the new sign-on (CESN) and sign-off (CESF) transactions.
- ▶ Review any intercommunications on your CICS TS system and the security definitions that are required or removed. Security of the MRO, APPC, and ISC connections also must be reviewed.

For more information about any security issues with the z/VSE Version 6 Release 1 and the CICS Transaction Server for z/VSE 2.1, see *CICS Transaction Server for VSE/ESA Security Guide*, SC33-1942.



CICS Explorer

The IBM CICS Explorer is a system management tool for use by system administrators and operators. It is designed to provide a simple, easy way of managing IBM CICS Transaction Server (CICS TS) systems.

The CICS TS can run in a partition of z/VSE or a region of z/OS.

The CICS Explorer Client is based on the Eclipse Rich Client Platform (RCP), which is an application that uses the windowing and GUI features of the operating system (for example, drag) and is integrated with the operating system's component model. The same CICS Explorer Client is used for accessing CICS systems that are running under z/VSE and z/OS.

The CICS Explorer Client interacts with its server part that is running on z/VSE or z/OS. The CICS Explorer server part on z/VSE consists of the following components:

- ▶ A client interface that manages incoming requests to the CICS TS and outgoing responses from the CICS TS.
- ▶ A system interface that executes the incoming requests.

The CICS Explorer features the following benefits:

- ▶ You can manage your CICS systems, perform tasks, and present information in an integrated and common way instead of (as previously) having to use various stand-alone graphical and non-graphical user interfaces.
- ▶ You can use your own plug-ins and future CICS tools if they were integrated into the Eclipse RCP.

For more information about the CICS Explorer, see the following resources:

- ▶ [CICS Explorer YouTube video](#)
- ▶ IBM Redbooks publication, *IBM CICS Explorer*, SG24-7778.

This chapter includes the following topics:

- ▶ 4.1, “Overview of CICS Explorer Workbench” on page 109
- ▶ 4.2, “Supported Operations views” on page 111
- ▶ 4.3, “Supported CICS Management Interface resources” on page 113
- ▶ 4.4, “Restrictions when connecting to a CICS TS for z/VSE system” on page 117
- ▶ 4.5, “Installing and configuring the CICS Explorer” on page 117
- ▶ 4.6, “Configuring a new connection to a CICS system” on page 121
- ▶ 4.7, “Using a connection to a CICS system” on page 125
- ▶ 4.8, “Using Job EYUPARM to enter or modify debugging commands” on page 126
- ▶ 4.9, “Messages generated when using the CICS Explorer” on page 128

4.1 Overview of CICS Explorer Workbench

Figure 4-1 shows the menu bar, view toolbar, perspective switcher, online help, fast view toolbar, and other details of the CICS Explorer Workbench. Our description is based on CICS Explorer Version 5.3.

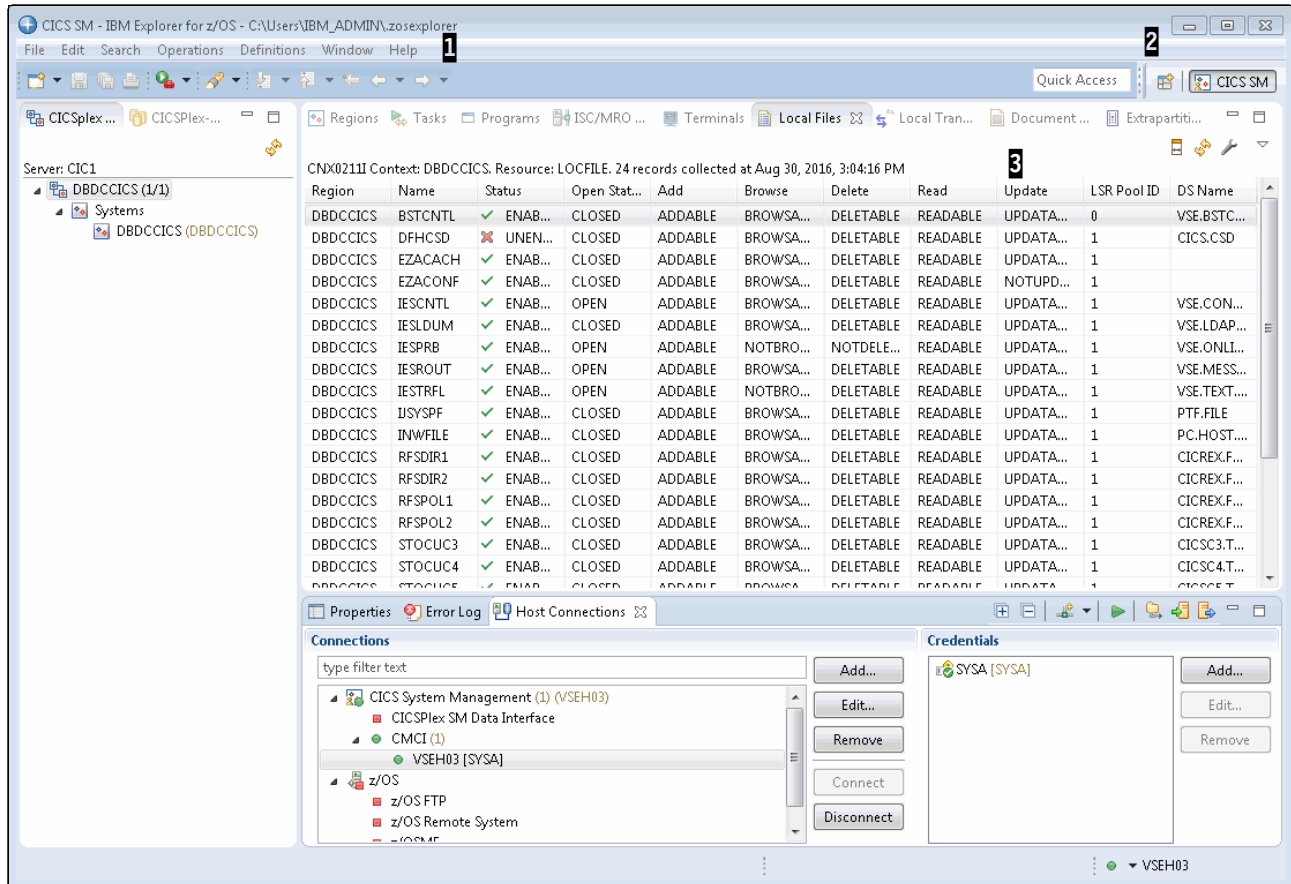


Figure 4-1 CICS Explorer Workbench

An Eclipse perspective is a set of related tools that are grouped for a specific task or role. For example, the perspective that is used by a CICS Systems Programmer is different from the perspective that is used by a Java™ Application Developer.

The CICS Explorer Client features the following built-in perspectives:

- ▶ System Management (CICS SM)

This perspective contains information about the currently connected CICS system, which includes the active regions, running tasks, and so on. If there is no active CICS system connection, all views in the perspective are empty.

- ▶ Resource

This perspective is primarily concerned with the manipulation of event bindings and bundle projects. This perspective requires the use of the CICS PA plug-in, which is not used in the z/VSE environment.

Figure 4-1 shows one perspective (SM Administration) used with the CICS Explorer.

If you click **Window** → **Open Perspective** → **Other**, you can see the list of perspectives that are supplied with this version of the CICS Explorer.

If you click **Operations** and then select a resource from the pull-down menu that is supported by the CICS Explorer (for example, Local Files), a resource view is created. In the example that is shown in Figure 4-1 on page 109, the Local Files view is created with a list of all local files that are defined to this CICS TS system.

For example, if file BSTCNTL is double-clicked, the attributes for this file are displayed, as shown in Figure 4-2.

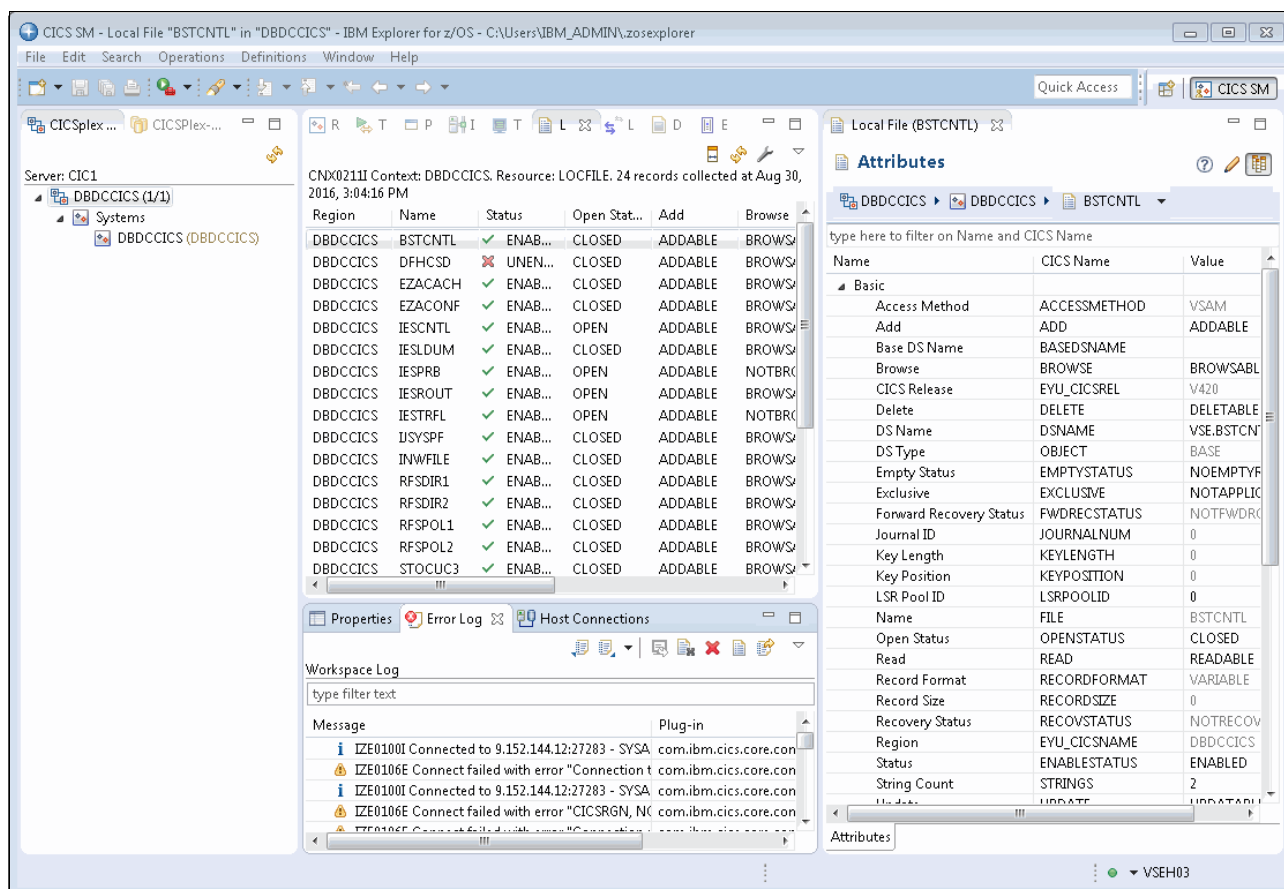


Figure 4-2 CICS Explorer Workbench illustrating display of a file's attribute

In Figure 4-2, you see that the attributes are displayed in tabular form and logically grouped. Consider the following points:

- ▶ Every attribute benefits from field-level verification, where the entry is validated in real time.
- ▶ Errors are identified by the Error icon, which identifies the field in error and the page on which the field appears.

If an entry in a resource view is right-clicked, the actions that are possible are displayed in bold font. For example, as shown in Figure 4-2, if file BSTCNTL is selected and right-clicked, a window is displayed in which the following options can be selected:

- ▶ Open
- ▶ Open File
- ▶ Close File

- ▶ Discard
- ▶ Enable
- ▶ Disable
- ▶ Copy
- ▶ Create System Event

Any functions that cannot be performed are shown in a pale font.

For more information about the CICS Explorer Workbench, see the online help that is provided with the CICS Explorer by clicking **Help** (see Figure 4-1 on page 109).

How to perform the following tasks are available in the online help:

- ▶ Reposition columns
- ▶ Resize columns
- ▶ Customize filters
- ▶ Create perspectives

4.1.1 Selecting CICS Explorer views

This example uses a selection of CICS Explorer views. It shows how you can switch between views to obtain useful information that otherwise requires having to use various CICS transactions and GUIs.

To display information about CICS resources, you might use the following views:

- ▶ The Program view displays the attributes of selected programs. If you use the equivalent CICS transaction CECI INQUIRE PROG(*program-name*), the information is distributed over several pages.
- ▶ The Regions view (Partitions view) shows more dynamic information, such as the maximum tasks that are allowed and number of times the maximum number of tasks was reached. This type of information, together with the “short on storage” indicators, provide valuable performance indicators.
- ▶ The Task view displays detailed information about tasks in the relevant CICS partition.
- ▶ The Queues view, which includes the Temporary Storage (TS) queue and Transient Data (TD) queue, determines how the temporary storage and transient data queues were used.

4.2 Supported Operations views

In this section, we describe the Operations menu item views that are supported when you use the CICS Explorer to connect to a CICS TS for z/VSE system.

If you select a resource from the Operations view, a resource view is opened that includes a list of all corresponding resource definitions. If you right-click an entry in this list, an action list for the entry is displayed. In this list, actions that can be performed are shown in bold font and actions that cannot be performed are shown in “pale” font.

The following operations views are supported for a CICS TS for z/VSE system:

- ▶ Document Templates

This view (DOCTEMP) view can be used to display or process information about document templates that are installed in the active CICS system.

- ▶ **Files**
The local Files (LOCFILE) and remote Files (REMFIL) view can be used to display or process information about local and remote CICS files in the current context and scope.
- ▶ **Interval Control Requests**
This view (REQID) can be used to display or process information about outstanding interval control requests in active CICS systems.
- ▶ **ISC/MRO Connections**
The InterSystems Communication (ISC)/Multiregion Operation (MRO) Connections (CONNECT) view can be used to display or process information about ISC over IBM Systems Network Architecture (SNA) and MRO connections.
- ▶ **Programs**
This view (PROGRAM) can be used to display or process information about currently installed programs.
- ▶ **Regions**
This view (CICSRGN) can be used to display or process information about the active CICS systems in the current context and scope.
- ▶ **Tasks**
This view (TASK) can be used to display or process information about tasks that are running in the current context and scope.
- ▶ **TCP/IP Services**
This view (TCPIPS) can be used to display or process information about the TCP/IP Services in an active CICS system.
- ▶ **Transient Data (TD) queue**
This queue includes the following types of views:
 - The extrapartition Transient Data Queues (EXTRATDQ) view can be used to display or process information about currently installed extrapartition transient data queues.
 - The indirect Transient Data Queues (INDTDQ) view can be used to display or process information about currently installed indirect transient data queues.
 - The intrapartition Transient Data Queues (INTRATDQ) view can be used to display or process information about currently installed intrapartition transient data queues.
 - The remote Transient Data Queues (REMTDQ) view can be used to display or process information about currently installed remote transient data queues.
- ▶ **Terminals**
This view (TERMNL) can be used to display or process information about tasks that are running in the current context and scope.
- ▶ **Transactions:**
 - The local Transactions (LOCTRAN) view can be used to display or process information about CICS and user-defined local transactions in the current context and scope.
 - The remote Transactions (REMTRAN) view can be used to display or process information about CICS and user-defined remote transactions in the current context and scope.
- ▶ **Transaction Classes**
This view (TRANCLAS) can be used to display or process information about the transaction classes for each CICS system.

► TS Queues

This view (TSQNAME) can be used to display information about temporary storage queues in an active CICS system.

4.3 Supported CICS Management Interface resources

In this section, we describe the CICS Management Interface resources that are supported when you use the CICS Explorer (or equivalent HTTP requests) to connect to a CICS TS for z/VSE system.

When the CICS Explorer is used, the following general program flow occurs:

1. The CICS Explorer uses the CICS Web Interface (HTTP format) to connect to a CICS TS and send a request.
2. The CICS TS receives the CICS Explorer request and converts this (HTTP) request into an EXEC CICS command or a Control Information Query.
3. The results from executing the EXEC CICS command or Control Information Query are converted back into HTTP format and are returned to the CICS Explorer.
4. The CICS Explorer Workbench displays the results in the views (see Figure 4-1 on page 109). In doing so, the CICS Explorer recognizes whether the connected CICS TS is running under z/VSE or z/OS and adjusts the views.

The connection process of the CMCI interface is shown in Figure 4-3.

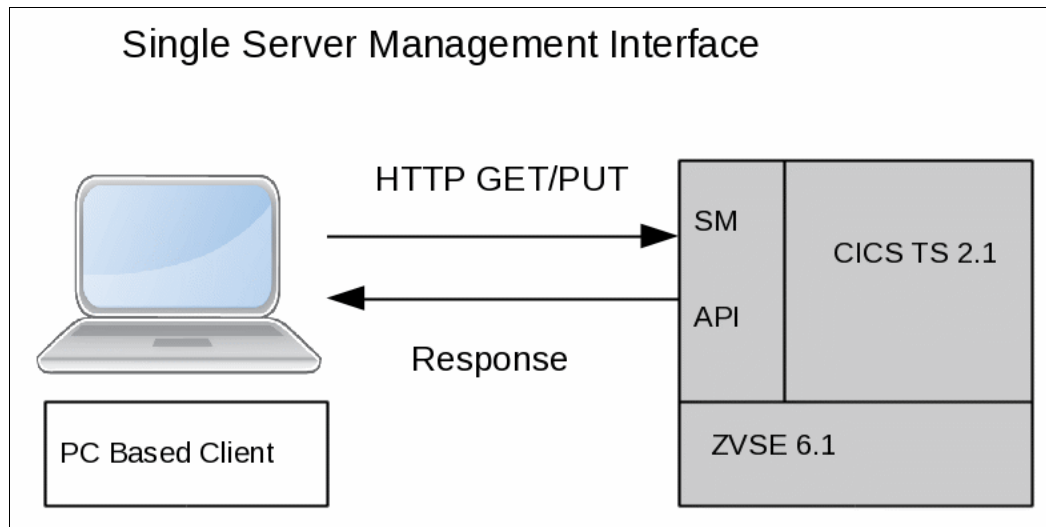


Figure 4-3 Schematic CMCI interface

You can also use the HTTP protocol to develop HTTP client applications that monitor and control CICS Management Interface resources.

The CICS Management Resources that are supported for CICS TS for z/VSE systems are listed in Table 4-1.

Table 4-1 CICS Management Resources supported for CICS TS for z/VSE systems

Resource name	Key name
CICSRegion	APPLID
CICSDocumentTemplate	NAME
CICSLocalTransaction	TRANID
CICSRemoteTransaction	TRANID
CICSTransactionClass	NAME
CICSProgram	PROGRAM
CICSTask CICSLocalFile	TASK
CICSLocalFile	FILE
CICSRemoteFile	FILE
CICISISCMROConnection	NAME
CICSTCPIPService	NAME
CICSTerminal	TERMID
CICSIntervalControlRequest	NAME
CICSExtrapartitionTDQueue	TDQUEUE
CICSIntrapartitionTDQueue	TDQUEUE
CICSRemoteTDQueue	TDQUEUE
CICSIndirectTDQueue	TDQUEUE
CICSTSQueue	NAME

4.3.1 Using an HTTP GET request for information about resources

To request the resource information, you can use the following HTTP GET request:

```
get http://hostname:port/CICSSystemManagement/resource/regionname
```

To request information about a selected resource item, you can use the following HTTP GET request:

```
get http://hostname:port/CICSSystemManagement/resource/regionname/regionname?CRITERIA=((keyname=='itemname'))
```

For example, to display information about program IESNEWS, the following parameters are used:

- ▶ keyname = PROGRAM
- ▶ itemname = IESNEWS

The following options are available:

- ▶ **hostname**
The TCP/IP host name of your stand-alone CICS region (partition).
- ▶ **port**
The port that is used to access the server. This value is specified in the PORTNUMBER attribute of the TCPIP SERVICE definition that is created when the CICS management client interface is configured.
- ▶ **resourcename**
The resource name as described in the list of resource names.
- ▶ **regionname**
The name of the CICS region (partition) from where you want to retrieve the resource information.

For more information, see 4.2, “Supported Operations views” on page 111.

4.3.2 Using an HTTP PUT request to process a single resource

To update or perform an action on a CICS Explorer resource << by using or that uses?>> using the HTTP protocol, an HTTP PUT request is required. PUT requests include an XML request body that contains either of the follow types of information:

- ▶ Details about the changes to be made to resource attributes.
- ▶ The action to be performed on the targeted resources.

The syntax of an HTTP PUT request is shown in Example 4-1.

Example 4-1 HTTP PUT syntax

```
put http://hostname:port/CICSSystemManagement/ResourceName/regionname/  
regionname?CRITERIA=((keyname=name))
```

The request body that is used to perform an action on a resource is shown in Example 4-2.

Example 4-2 Request body

```
<request>  
<action name='action' />  
</request>
```

Example 4-3 shows how to open a file.

Example 4-3 Open a file

```
http://hostname:port/CICSSystemManagement/  
CICSLocalFile/PRODCICS/PRODCICS?CRITERIA=((FILE=='FILE001'))
```

The request body for the open file example is shown in Example 4-4.

Example 4-4 Request body

```
<request>  
<action name="OPEN" />  
</request>
```

Some action requests require another parameter. Example 4-5 shows that a FILE is closed.

Example 4-5 File closed

```
http://hostname:port/CICSSystemManagement/  
CICSLocalFile/PRODCICS/PRODCICS?CRITERIA=((FILE=='FILE001'))
```

The request body for the file closed example is shown in Example 4-6.

Example 4-6 Request body for the file closed

```
<request>  
<action name='CLOSE' >  
<parameter name='BUSY' value='NOWAIT' />  
</action>  
</request>
```

The request body that is used to update one or more attributes is shown in Example 4-7.

Example 4-7 Update one or more attributes

```
<request>  
<update>  
<attributes attribute_name="new_value" [attribute_name = "new_value" ...] (/>  
</update>  
</request>
```

Example 4-8 shows updating the value of MAXTASKS value to 60.

Example 4-8 Update the value of MAXTASKS

```
put http://hostname:port/CICSSystemManagement/CICSRegion/PRODCICS/  
PRODCICS?CRITERIA=((APPLID=='PRODCICS'))
```

The request body for the update of the value of MAXTASKS is shown in Example 4-9.

Example 4-9 MAXTASK example request body

```
<REQUEST>  
<UPDATE>  
<ATTRIBUTES MAXTASKS="60" />  
</UPDATE>  
</REQUEST>
```

4.4 Restrictions when connecting to a CICS TS for z/VSE system

The following restrictions can occur when the CICS Explorer is used to connect to a CICS TS for z/VSE system:

- ▶ When defining a connection, you can select the CICS Management Interface only. Other types of connection (for example, the IBM CICSplex® SM Data Interface) are not supported.
- ▶ The Definitions item that is shown in the menu bar in Figure 4-1 on page 109 is not supported.
- ▶ Not all views are supported within the pull-down menu for the Operations item. For more information about the views that are supported, see 4.2, “Supported Operations views” on page 111. If you select a view (for example, Atom Services) that is not supported, the following message is displayed:

CNX0203I This resource is not available

- ▶ Not all CICS Management Interface resources are supported. For more information about the resources that are supported, see 4.3, “Supported CICS Management Interface resources” on page 113.
- ▶ Only the IPv4 protocol is supported. The IPv6 protocol is not supported.

4.5 Installing and configuring the CICS Explorer

In this section, we describe the tasks that you must complete to install and configure the CICS Explorer. This description differentiates between the CICS Explorer client-part (which is running on a local workstation, a remote network drive, or a shared Linux server) and the CICS Explorer server-part (which is running on z/VSE).

The z/VSE server offers the following functionality:

- ▶ z/VSE 5.2 with CICS TS for z/VSE 1.1.1: Monitoring CICS Resources
- ▶ z/VSE 6.1 with CICS TS for z/VSE 2.1: Monitoring and updating CICS Resources (enable/disable).

The CICS Explorer server support is integrated into CICS TS for z/VSE 2.1; for CICS TS for z/VSE 1.1.1, the support is available by using PTF.

4.5.1 Configuring the z/VSE host for use with CICS Explorer

Before you start installing the client part of the CICS Explorer, you must prepare your z/VSE host system so that the CICS Explorer can connect to a CICS Transaction Server that is running in a z/VSE partition. The following procedure assumes that the CICS Explorer is used with the second, predefined CICS TS (CICS2) (for more information about how to set up a second CICS, see *z/VSE Administration*, SC34-2692). Complete the following steps:

1. Skeleton CEXPLCSD is stored in VSE/ICCF Library 59 and contains the instructions that are described in this procedure. Change the entry for file EYUPARM in CEXPLCSD so that EYUPARM is defined as a Basic Access Method (BAM) file. Example 4-10 on page 118 shows the relevant statements in CEXPLCSD.

Example 4-10 Relevant CEXPLCSD statements

```
* --V001-- VALID WHERE THE FILE RESIDES
* --V002-- START TRACK/BLOCK
* --V003-- NUMBER OF TRACKS/BLOCKS (1 TRACK OR 64 BLOCKS)
* VALID INPUT COULD BE ANY TRACK, TRAP, OR TRACE COMMAND, AN
* EXAMPLE WOULD BE (YOU HAVE TO REPLACE THE BLANK LINE
* AFTER THE $$DITTO CSQ COMMAND):
* TRAP(NQCR,12)
* TRACK(XQPQ,FROM,NQLT,SPEC)
* DATTRACE(18)
* CACHECMXTND(1024)
* YOU WILL BE INSTRUCTED BY IBM PERSONNEL ON SPECIFIC
* COMMANDS TO BE USED.
```

2. Run the skeleton CEXPLCSD, which performs an EXEC DFHCSDUP to define groups SMSSEYU and CMCI. These groups contain the profiles, programs, and transactions that are used with the CICS Explorer. Example 4-11 shows the transaction CORM.

Example 4-11 CORM example

```
DEFINE TRANSACTION(CORM) GROUP(SMSSEYU)
PROGRAM(EYU9NXRM) TWASIZE(512) PROFILE(DFHICST) STATUS(ENABLED)
TASKDATALOC(ANY) TASKDATAKEY(CICS) STORAGECLEAR(NO)
RUNAWAY(50000) SHUTDOWN(DISABLED) DYNAMIC(NO) PRIORITY(255)
TRANCLASS(DFHTCLOO) DTIMOUT(NO) INDOUBT(BACKOUT) RESTART(NO)
SPURGE(NO) TPURGE(NO) DUMP(YES) TRACE(YES) CONFDATA(NO)
RESSEC(NO) CMDSEC(NO)
```

An EXEC BSTADMIN defines the BSM security for the CICS transactions (see Example 4-12) that defines the CORM.

Example 4-12 CORM to define BSM security

```
ADD TCICSTRN 'CORM' UACC(NONE) DATA('IBM SUPPLIED')
PERMIT TCICSTRN 'CORM' ID(GROUP01) ACCESS(READ)
```

An EXEC DITTO defines and initializes the EYUPARM file. This file is used in job EYUPARM, as described in 4.8, “Using Job EYUPARM to enter or modify debugging commands” on page 126.

3. Use skeleton DFHDCTC2 to activate EYUPARM and COPR. Example 4-13 shows the entries in DFHDCTC2.

Example 4-13 DFHDCTC2 entries

```
* EYUPARM DFHDCT TYPE=SDSCI, CICS EXPLORER PARAMETER
DSCNAME=EYUPARM,
BLKSIZE=80,
RECSIZE=80,
RECFORM=FIXUNB,
TYPEFLE=INPUT,
DEVADDR=SYS075,
DEVICE=DISK,
BUFNO=1
...
* COPR
DFHDCT TYPE=EXTRA, CICS EXPLORER INPUT PARAMETER
DESTID=COPR,
```

```
DSCNAME=EYUPARM,  
RECFORM=FIXBLK,  
TYPEFLE=INPUT,  
OPEN=INITIAL
```

4. Add the following DLBL and EXTENT statements to the job DTRCICS2 that defines the labels for a second CICS. Job DTRCICS2 is contained within skeleton SKPREPC2 (stored in VSE/ICCF library 59), as shown in Example 4-14.

Example 4-14 DLBL and EXTENT statements

```
// DLBL EYUPARM,'EYUPARM.FILE',0,SD  
// EXTENT SYS075,--V001--,1,0,--V002--,--V003--
```

5. Add the ASSGN statement for the EYUPARM file to the CICS startup job DTRCICS2, as shown in Example 4-15 on page 119.

Example 4-15 ASSGN statement

```
// ASSGN SYS075,DISK,VOL=--V001--,SHR
```

6. In your CICS System Initialization Table (for example DFHSITC2, which is stored in VSE/ICCF library 59) ensure that the following parameters are set:

- SEC=YES
- TCPIP=YES

7. In some environments, the CICS Explorer requires another 31-bit partition storage.

If you select Environment C, the amount of partition storage for PRODCICS is sufficient to run the CICS Explorer.

For other DBDCCICS and Environments A and B, you should increase the partition size (ALLOC procedure) and PASIZE (Tailor IPL). As a result, VSIZE often must be increased. For Environment A, you should also change the ELIM parameter in skeletons SKCICS2 or SKCICS, as shown in Example 4-16.

Example 4-16 ELIM parameters changed

```
// IF XENVNR = A THEN  
// SETPARM ELIM=25M
```

For Environments B and C, the ELIM parameter must not be modified.

8. Create a conversion table by using the skeleton DFHCNV (stored in VSE/ICCF library 59) and then, perform a CEMT SET PRO(DFHCNV)NEW in the CICS system you are using (DBDCCICS or PRODCICS).
9. By using the CEDA transaction, install groups SMSSEYU and CMCI in your PRODCICS.
10. Define a TCPIP SERVICE in z/VSE. Consider the following points:
 - As shown in Figure 4-4 on page 120, you must specify your own IP address.
 - If TCP/IP is running, you can open this TCPIP SERVICE by using the **CEMT SET TCPIPS** command.
 - You must use different ports if you want to use the CICS Explorer for multiple CICS Transaction Server for z/VSE systems.
 - Per default, the CICS Explorer attempts to connect to a CICS system by using the Security Sockets Layer (SSL) protocol. If the SSL connection is unsuccessful, the connection is reattempted by using basic authentication.

If you want to use SSL, you must select **YES** in the field SSL that is shown in Figure 4-4. To use SSL connections, you must enter a different port number and define valid certificates.

```

DEF TCPI
OVERTYPE TO MODIFY                                CICS RELEASE = 0420
CEDA DEFINE TCPIPService(                          )
  TCPIPService ==> CMCIT
  Group        ==> CMC
  Description   ==> CICS Explorer
  Urm          ==> dfhwbady
  Portnumber   ==> 27283                1-65535
  Certificate   ==>
  SStatus      ==> Open                Open | Closed
  SSl          ==> No                  Yes | No | Clientauth
  Attachsec    ==> Verify              Local | Verify
  TRansaction  ==> cwxn
  Backlog      ==> 00001                0-32767
  TSqprefix    ==>
  Ippaddress   ==> 9.
  S0cketclose  ==> No                  No | 0-240000

MESSAGES: 2 SEVERE                                SYSID=CIC1 APPLID=DBDCCICS

PF 1 HELP 2 COM 3 END                            6 CRSR 7 SBH 8 SFH 9 MSG 10 SB 11 SF 12 CNCL
MA A                                             16/024
Connected to remote server/host boeblingen.de.ibm.com using port 23
deboel43-140-01-71032-04-Boeblingen on det

```

Figure 4-4 Defining a TCPIP SERVICE for the CICS Explorer

11. Ensure that the TCP/IP Server is identified by a name for its IP address; otherwise, CICS WEB Support does not initialize successfully (and the TCPIP SERVICE cannot be opened). Use the TCP/IP command **DEFINE NAME** to set this identification requirement.
12. Ensure that the CICS resource definition group DFH\$WBSN is activated for the CICS Transaction Server for z/VSE system that is used with the CICS Explorer.

4.5.2 Starting and stopping the CICS Explorer server-part

After completing the installation of the CICS Explorer client-part as described in 4.5.3, "Obtaining a copy of the CICS Explorer Client" on page 120 and 4.5.4, "Installing CICS Explorer Client" on page 121, you can use the CORM transaction (entered at a 3270 terminal or the z/VSE Console) to start the server-part of the CICS Explorer.

You can use the COSH transaction to stop the server-part of the CICS Explorer.

4.5.3 Obtaining a copy of the CICS Explorer Client

You obtain a copy of the client-part of the CICS Explorer by downloading it from the CICS Explorer web page. Complete the following steps:

1. Browse to [the z/VSE Components web page](#).

From within the CICS Explorer section, click **IBM CICS Explorer download** and you are redirected to the IBM CICS Explorer Downloads web page.

2. Click **CICS Explorer Downloads** and you are redirected to the IBM CICS Explorer Family web page.
3. Select the **IBM Installation Manager (IM)** method.
4. When starting from scratch, download the IM installer for IBM Explorer for z/OS Aqua release by selecting one of the platforms (Windows, Linux, or Mac OS).

Note: Consider the following points:

- ▶ The CICS Explorer that you download from the IBM CICS Explorer Family web page is used for connecting to CICS TS for z/VSE and CICS TS for z/OS systems. The CICS Explorer recognizes whether the connected CICS TS is running under z/VSE or z/OS and adjusts the views.
- ▶ To use the VSE CICS Explorer Update capabilities that are provided with z/VSE 6.1, the CICS Explorer 5.2 (or later) client is required.

4.5.4 Installing CICS Explorer Client

You can install the CICS Explorer Client on a local workstation, a remote network drive, or a shared Linux server.

Complete the following steps to install the CICS Explorer on a local workstation:

1. Ensure that you completed the task that is described in *Obtaining a copy of the CICS Explorer* to download the CICS Explorer .zip file (or a .tar.gz file on Linux) from the download site to your local workstation.
2. Extract the contents of the .zip or .tar.gz file.
3. Run the IM installer as a user or an administrator. (The user installation is more commonly used.)
4. Select and install one or more available installation packages from within IM installer.
5. Find the eclipse.exe file (Eclipse on Linux) and create a shortcut on your desktop.

You can now double-click the shortcut icon to start the CICS Explorer.

4.6 Configuring a new connection to a CICS system

In this section, we describe how you can configure a new connection to a CICS system (a CICS Transaction Server for z/VSE or a CICS Transaction Server for z/OS system).

Before you attempt to connect to a CICS system, you might want to check that the server part of the CICS Explorer is started by using the CORM transaction. You can use the COSH transaction to stop the server part of the CICS Explorer.

4.6.1 Adding a set of CICS Explorer credentials

Before you can use the CICS Explorer to connect to a CICS system (a CICS Transaction Server for z/VSE or a CICS Transaction Server for z/OS), you must have at least one set of credentials. When you connect to a CICS system, your credentials (user ID and password or passphrase) are sent to the CICS system for authentication.

After you define a set of credentials, you can use them on all CICS systems that share the credentials (you are not required to reenter your details each time).

Note: Before starting this procedure, ensure that you have the correct level of authorization to connect to your CICS system.

To add a set of credentials, complete the following steps:

1. Click **Window** → **Manage Connections** on the menu bar (see Figure 4-1 on page 109). The Host Connections window that is shown in Figure 4-5 opens.

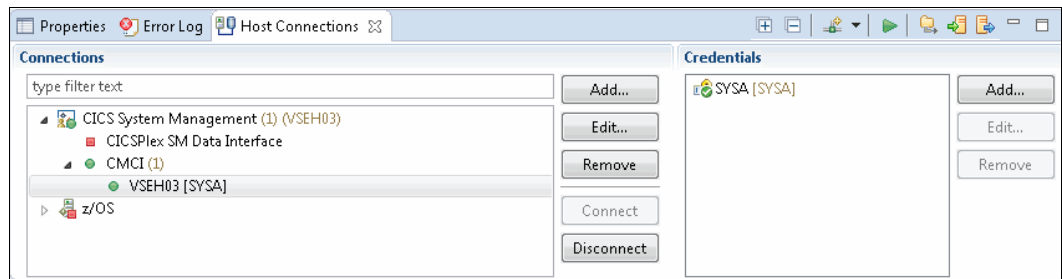


Figure 4-5 Host Connections window

2. Click **Add** in the Credentials section of the window that is shown in Figure 4-5 and a New Credentials window opens, as shown in Figure 4-6.

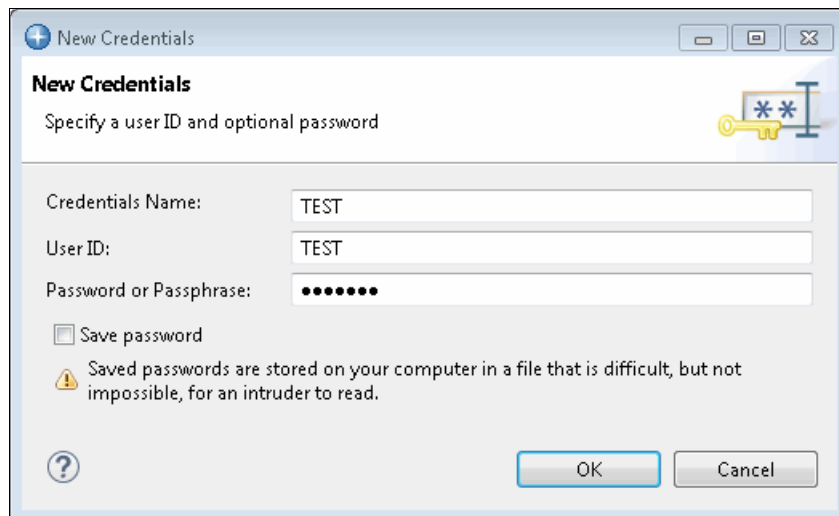


Figure 4-6 New Credentials window

3. Enter your credential details and a credential name.

The name is used only to help you distinguish between different credentials. If you do not enter a name, the default name that is used is the same as the User ID.

Enter a password or passphrase and select the **Save password** option to save the password.

Note: If you do not enter a password or passphrase, you are prompted to enter a password or passphrase with each connection request.

Click **OK**.

If you want to add a connection to a CICS system, proceed to the next section.

4.6.2 Adding a CICS Explorer connection

In this section, we describe how to add a CICS Explorer connection that can be used for connecting to a CICS system.

Before you attempt to connect to a CICS system, you might want to check that the server part of the CICS Explorer was started by using the CORM transaction. You can use the COSH transaction to stop the server part of the CICS Explorer.

Complete the following steps to add a connection:

1. If a Host Connections window is not open, click **Window** → **Manage Connections** in the menu bar that is shown in Figure 4-1 on page 109. A Host Connections window opens.
2. Click **Add** in the Connections section and a pull-down menu is displayed that includes the following CICS system choices:
 - CMCI - CICS Management Interface
 - CICSplex SM Data Interface
 - z/OS FTP
 - z/OSMF
 - z/OS Remote System
3. Your selection depends upon whether the connection is to a CICS Transaction Server for z/VSE system or to a CICS Transaction Server for z/OS system. Consider the following points:
 - If you want to connect to a CICS Transaction Server for z/VSE system, select **CMCI** (CICS Management Interface). All other options are not supported.
 - If you want to connect to a CICS Transaction Server for z/OS system, you can select any of the four options that are listed in Step 2.

The Add CICS Management Interface Connection window opens, as shown in Figure 4-7.

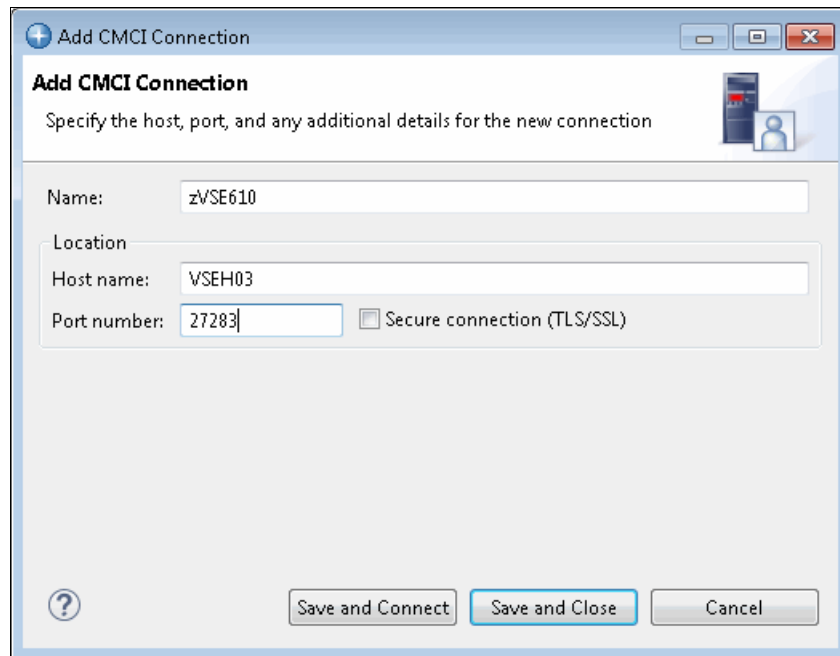


Figure 4-7 Add CICS Management Interface window

4. Enter the following information:
 - Name: The local name that is used to identify this connection. The name is used only to help you distinguish between different connections.
 - Host name: The TCP/IP host name or IP address of the host system that is running your CICS region (partition).
 - Port number: The port that is used to access the server. This value is the value that is specified in the PORTNUMBER attribute of the TCPIP SERVICE definition that was created during the configuration of the CICS management interface.
5. If you click **Save and Connect** (see Figure 4-7 on page 124), the CICS Explorer attempts to connect to the CICS system you configured. The connection also is stored for your future use. The Signon dialog is now displayed. Enter your User ID and password, and click **OK**.

If the connection is successful, the connection name appears in the connection status bar in the lower right corner of the Workbench window (as shown in Figure 4-1 on page 109), next to one of the following items:

- Green icon, which indicates a non-SSL connection
- Padlock icon, which indicates an SSL connection

If the connection is unsuccessful, a red icon is displayed in the connection status bar in the lower right corner of the Workbench window (as shown in Figure 4-1 on page 109), next to the connection name. An error message is displayed at the top of the Connections Preferences view, which indicates the reason for the failure. Check the values in the fields, correct any errors, and click **Connect** to test your corrections.

The first time that you use the CICS Explorer to connect to a CICS system, a user workspace is automatically created in a default location. For more information about changing this default location, see 4.7.1, "Changing a CICS Explorer user workspace" on page 126.

4.7 Using a connection to a CICS system

In this section, we describe how to use a CICS Explorer connection to connect to a CICS system (a CICS Transaction Server for z/VSE).

Before you attempt to connect to a CICS system, you might want to check that the server part of the CICS Explorer was started by using the CORM transaction. You can use the COSH transaction to stop the server part of the CICS Explorer.

Complete the following steps to use an CICS Explorer connection:

1. If a Host Connections window is not open, click **Window** → **Manage Connections** on the menu bar that is shown in Figure 4-1 on page 109. A Host Connections window then opens.
2. Your selection now depends upon whether the connection is to a CICS Transaction Server for z/VSE system or to a CICS Transaction Server for z/OS system. If you want to connect to a CICS Transaction Server for z/VSE system, complete the following steps:
 - a. Click the **CICS System Management** option and the CICS Management Interface is displayed.
 - b. Click the **CICS Management Interface** option and the CICS Transaction Server for z/VSE systems to which you can connect are displayed.
 - c. Select a CICS Transaction Server for z/VSE system together with a set of Credentials, as shown in the Figure 4-8 on page 125.

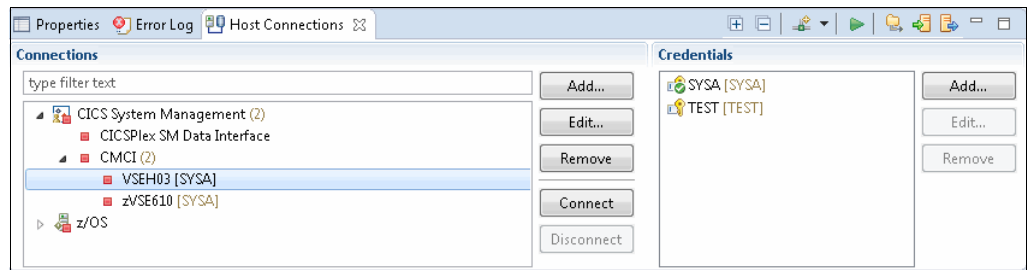


Figure 4-8 Selecting a Host Connection to a CICS TS for z/VSE system

- d. Click **Connect**.
3. The CICS Explorer attempts to connect to the CICS system you selected.

If the connection is successful, the connection name appears in the connection status bar that is in the lower right corner of the Workbench window (see Figure 4-8), next to one of the following icons:

- Green icon, which indicates a non-SSL connection
- Padlock icon, which indicates an SSL connection

If the connection is unsuccessful, a red icon is displayed in the connection status bar in the lower right corner of the Workbench window (see Figure 4-8 on page 125), next to the connection name. An error message is displayed at the top of the Connections Preferences view, which gives the reason for the failure. Check the values in the fields, correct any errors, and click **Connect** to test your corrections.

4.7.1 Changing a CICS Explorer user workspace

In this section, we describe how to change a user's CICS Explorer workspace from the default location that is created automatically by the CICS Explorer when a user starts the CICS Explorer for the first time.

The CICS Explorer workspace contains connection and configuration information. Because the workspace includes user IDs and passwords, you should ensure that the workspace can be accessed by the owning user only.

Per default, the CICS Explorer creates a workspace on the local file system.

The following names of the default workspace folder are used:

- ▶ CICS Explorer Version 5.3 or later: `.zosexplorer`
- ▶ CICS Explorer Version 5.2 and earlier: `.cicsexplorer`

Under Linux, the default user workspace is created in the following directory:

`<drive letter>/home/<username>/.zosexplorer`

Under Windows, the default user workspace is created in the following directory

`<drive letter>\Documents and Settings\<username>\.zosexplorer`

To change the location of a user's workspace, complete the following steps that are described in *IBM CICS Explorer User Guide*:

1. Click **Help** in the CICS Explorer Workbench window (see Figure 4-1 on page 109).
2. Select **Help Contents** from the pull-down menu. The IBM CICS Explorer User Guide is opened.

In the chapter "Installing" of the *IBM CICS Explorer User Guide*, you find section "Changing the CICS Explorer workspace location".

4.8 Using Job EYUPARM to enter or modify debugging commands

By using job EYUPARM, you can enter or modify the debugging commands and their parameters that are used for debugging the CICS Explorer.

You can then use CICS transaction COD0 to start debugging from the moment the CICS Explorer is started. Per default, debugging is not activated.

Note: You should use the CICS Explorer debugging transactions COD0 and CODB only when requested to do so by IBM Customer Support Personnel.

You can use the EYUPARM job to change or enter the **TRACE**, **TRAP**, and **TRACK** commands (see Example 4-17). The following values are used:

- ▶ The value of TRAP for NQCR is set to level 12.
- ▶ The TRACK statement activates trace flags for the XQPQ method only when it is called from the NQLT method.
- ▶ The value of DATTRACE is set to level 18.

The CACHEDMXTND(1024) statement (see Example 4-17) increases the size of the internal cache. This increase might be required, for example, if the Program view fails with following message EYUXC0020E:

Cache request exceeds extension size for compid cache

Example 4-17 CACHEDMXTND(1024) statement

```
// JOB EYUPARM
// DLBL EYUPARM,'EYUPARM.FILE',0,SD
// EXTENT SYS001,SYSWK1,1,0,1577060,64
// UPSI 1
// EXEC DITTO
$$DITTO SET EOD=@@@
$$DITTO CSQ BLKFACTOR=1,FILEOUT=EYUPARM,RECSIZE=80
TRAP(NQCR,12)
TRACK(XQPQ,FROM,NQLT,SPEC)
DATTRACE(18)
CACHEDMXTND(1024)
@@@
/*
/ &
```

The default settings for the EYUPARM job are empty (no settings) and are established with skeleton CEXPLCSD (see Example 4-18).

Example 4-18 EYUPARM job

```
// JOB EYUPARM
// DLBL EYUPARM,'EYUPARM.FILE',0,SD
// EXTENT SYS001,SYSWK1,1,0,1577060,64
// UPSI 1
// EXEC DITTO
$$DITTO SET EOD=@@@
$$DITTO CSQ BLKFACTOR=1,FILEOUT=EYUPARM,RECSIZE=80
@@@
/*
/ &
```

4.9 Messages generated when using the CICS Explorer

In this section, we summarize the following message prefixes that you might receive when the CICS Explorer is used and where you can obtain explanations of these messages:

- ▶ For CNX (for example, CNX0222W), start the CICS Explorer and then complete the following steps:
 - a. From the toolbar, select **Help** and then, select **Help Contents**. The Help - IBM CICS Explorer window opens.
 - b. Click the plus sign (+) that is to the left of the CICS Explorer User Guide.
 - c. Click the plus sign (+) that is to the left of Reference information.
 - d. Click the plus sign (+) that is to the left of Messages and then, click **CICS Explorer (CNX) Messages**. All current messages with prefix CNX are listed. You can now select the message for which you need more information.
- ▶ For EYU (for example, EYUAR0001E), most of the EYU prefix messages that you might receive with z/VSE 6.1 are described in *z/VSE Messages and Codes, Volume 2*, SC34-2683. If a message is missed, you can search for “CICSPllex SM Messages and Codes” at [the CICS TS Information Center at website](#).
- ▶ For DFHWU (for example, DFHWU4002), you can search for “CICS TS Messages and Codes Volume 2” at [the CICS TS Information Center at website](#).



Web support

This chapter describes how IBM CICS Transaction Server (CICS TS) applications can be accessed from the internet. The basic method is the CICS Web Support (CWS), which enables CICS TS to start programs in response to an HTTP request.

Readers should understand the following key concepts:

- ▶ **SOAP:** SOAP is a standard, XML-based, industry-wide protocol that allows applications to exchange information over the internet by using HTTP.
- ▶ **3270 Bridge:** The 3270 bridge facility runs 3270-based transactions with a browser without the need to change the associated application.
- ▶ **ECI/CICS Transaction Gateway (CTG):** The CTG provides a comprehensive set of Java based programs for access to CICS applications from a web browser. The CTG is a three-tier connector model.
- ▶ **CICS Listener:** The CICS Listener supports the establishment of TCP/IP socket applications, which are built on the client/server model.

This chapter includes the following topics:

- ▶ 5.1, “Connection security” on page 130
- ▶ 5.2, “Using SOAP for inter-program communication” on page 130
- ▶ 5.3, “3270 bridge” on page 138
- ▶ 5.4, “ECI/CICS Transaction Gateway” on page 156
- ▶ 5.5, “CICS Listener” on page 159
- ▶ 5.6, “Evaluation” on page 162

5.1 Connection security

Securing the data traffic within a connection is important to ensure privacy. Generally, the following methods are used to secure a connection:

- ▶ Transport-layer security
- ▶ Message-layer security

5.1.1 Transport-layer security versus message-layer security

Transport-layer security secures the network communication between the communication partners by encrypting the data that is being transmitted over the network. In addition, data integrity, authentication, and confidentiality can be achieved. Transport-layer security typically uses digital signatures, Public Key Infrastructure (PKI) certificates, and secure hash functions to prevent messages from being “camouflaged”, passwords from being hacked, and transactions from being denied.

In situations where an environment consists of several hops, the communication between each hop must be considered separately in terms of transport-layer security (see Figure 5-1).

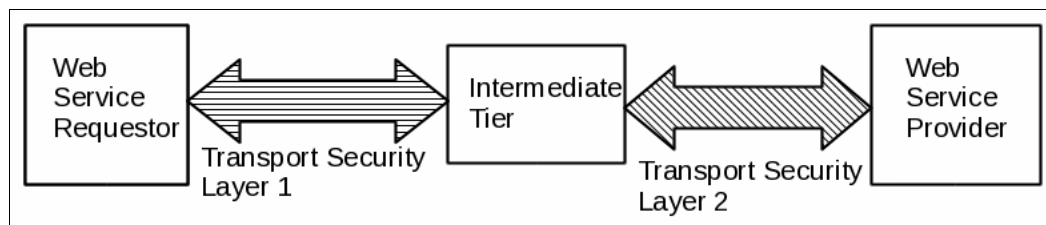


Figure 5-1 Connection Security for web processing

As shown in Figure 5-1, the connections between each hop might use different transport-layer security methods (or even no transport security for some connections). Transport-layer security does not “span” multiple hops, meaning an intermediate hop might be able to read the message.

To achieve end-to-end security, you must use message-layer security. Using message-layer security, the message is secure and does not change when sent over multiple hops. Transport-layer security can be implemented by using any of the following industry-wide protocols:

- ▶ Secure Sockets Layer (SSL), which is denoted by HTTPS.
- ▶ Virtual Private Network/Internet Protocol Security (VPN/IPSec), which is not apparent to applications.

Message-layer security includes security-related information in the message.

5.2 Using SOAP for inter-program communication

This chapter describes how you use the SOAP to send and receive information between CICS programs and other modules, over the internet.

For more information about SOAP, see the [Apache documentation website](#).

Consider the following points:

- ▶ The implementation of SOAP for z/VSE is for use with the CICS TS for VSE/ESA or CICS TS for z/VSE only.
- ▶ The implementation of SOAP for z/VSE does not require the use of either of the following components:
 - Universal Description, Discovery, and Integration (UDDI)
 - Web Services Description Language (WSDL)

5.2.1 z/VSE support for web services and SOAP overview

SOAP is a standard, XML-based, industry-wide protocol that allows applications to exchange information over the internet by using HTTP.

eXtensible Markup Language (XML) is a universal format that is used for structured documents and data on the web. It is independent of the web client's operating system platforms and the programming language that is used. HTTP is supported by all internet web browsers and servers.

SOAP combines the benefits of XML and HTTP into one standard application protocol. As a result, you can send and receive information to and from various platforms.

By using web browsers, you can view information that is contained on websites. However, by using SOAP you can combine the contents of different websites and services, and generate a complete view of all the relevant information.

z/VSE supports the SOAP protocol; therefore, it allows you to implement web services.

An example of the use of SOAP might be when a travel agent requires a combined view of the web services that cover hotel reservation, flight booking, and car rental. After the travel agent enters the required data, all three web services from the three different providers are processed in one transparent step. This approach is an example of how a "business-to-business" (B2B) relationship can be implemented.

5.2.2 SOAP syntax overview

You do not often need to concern yourself with the tagging that is described in this section because it is automatically generated by one of the following methods:

- ▶ SOAP client and converted to native data by the SOAP server.
- ▶ SOAP server and converted to native data by the SOAP client-processor.

However, for debugging purposes, you might need more information about SOAP tagging. For more information, see the [Apache documentation website](#).

In this section, we provide an overview of SOAP.

A SOAP message is a standard XML document that includes the following main parts:

- ▶ A SOAP envelope that defines the content of the message.
- ▶ A SOAP header (optional) that contains header information.
- ▶ A SOAP body that contains call and reply information.

The following types of elements are used for the SOAP message:

- ▶ The <Envelope> element is the root element of a SOAP message and defines the XML document to be a SOAP message.

- ▶ The <Header> element can be used to include more application-specific information about the SOAP message or security-specific information. The information here is user-defined. For example, it might be used to define the language that is used for the message.
- ▶ The <Body> element is used to define the message.
- ▶ The <Fault> element can be optionally used within the <Body> element and is used to supply information about any errors that might occur when the SOAP message was processed.

Figure 5-2 shows example of the SOAP syntax.

```
<soap:Envelope>
  <soap:Header>
    ...
  </soap:Header>
  <soap:Body>
    <GetStock>
      <Company>IBM</Company>
    </GetStock>
  </soap:Body>
</soap:Envelope>
```

Figure 5-2 Sample SOAP message

Figure 5-2 also shows a simplified version of a SOAP XML document that is used for requesting the IBM share price.

5.2.3 Web Service Security overview

Web Service Security includes the following types of security:

- ▶ Transport-layer security
- ▶ Message-layer security

Transport-layer and message-layer security provide security features for the following tasks:

- ▶ Authentication and authorization
- ▶ Data encryption and signatures

5.2.4 Using authentication with web Service Security

The use of authentication allows a service provider to check who is using the requested service. The service provider also might use this information to run the service under a specific user ID, with its associated access rights (authorization).

To fully understand authentication and authorization, it is important to understand the following concepts:

- ▶ Authentication

The process of identifying an individual by using the credentials of that individual.

- ▶ Authorization

The process of determining whether an authenticated client is allowed to access a resource or perform a task within a security domain. Authorization uses information about a client's identity and roles to determine the resources or tasks that a client can perform.

► **Credentials**

A set of claims that is used to prove the identity of a client. They contain an identifier for the client and proof of the client's identity, such as a password. They might also include information (such as a signature) to indicate that the issuer certifies the claims in the credential.

► **Identification**

The use of an identifier that allows a system to recognize a particular subject and distinguish it from other users of the system.

Authentication can be performed by using one of the following methods:

► **Transport-layer authentication**

The transport layer carries information about who is requesting the service. The following implementations can be used:

- HTTP Authentication Basic and Digest Access Authorization (RFC 2617). For more information, see the [HTTP basic authentication page](#) of the IBM Knowledge Center website.
- The use of SSL Client Authentication with SSL/HTTPS.

► **Message-layer authentication**

The SOAP message carries information about who is requesting the service. The following implementations can be used:

- Direct authentication by using plain text passwords or a password digest.
- Brokered Authentication by using an X.509 Certificate, Kerberos, Security Token Services, or SAML Assertion. Brokered Authentication by using an X.509 Certificate carries the X.509 Certificate as part of the SOAP header (see Example 5-1).

Example 5-1 SOAP example

```
<soap:Header>
<Security xmlns="...secect-1.0.xsd"
<BinarySecurityToken EncodigType= "wsse:Base64Binary"
ValueType= "wsse:X509v3">
MIICuzCCAiqCBF...
BinarySecurityToken>
</Security>
...
```

Direct authentication defines the following methods for transporting the password:

- **Plain text password** in which UsernameToken is used to transport the actual password. If you use plain-text password configuration, you must use a secure transport method, such as HTTPS (see Example 5-2).

Example 5-2 Plain text password

```
<soap:Header>
<Security xmlns="...secect-1.0.xsd"
<UsernameToken>
<Username>John Smith</Username>
<Password>Pass12wd</Password>
</UsernameToken>
</Security>
...
```

- Password digest, in which the communicating parties (the requester and the service) use an insecure transport channel. Steps must be taken to protect the passwords from being exposed to others. The requester creates a digest of the actual password that is concatenated with a set of random bytes (field nonce) and another value that depends on the creation-time (field created). This digest is computed by using the following equation:

$$\text{digest} = \text{Base64_encode}(\text{SHA-1}(\text{nonce} + \text{created} + \text{password}))$$
- To authenticate the request, the service computes the digest value by using the password that is bound to the received user name. It compares the received digest value with the computed digest value (see Example 5-3).

Example 5-3 Password that is bound to the received user name

```

<soap:Header>
<Security xmlns="...secect-1.0.xsd"
<UsernameToken>
<Username>John Smith</Username>
<Password Type="...#PasswordDigest">AFHHF23wger=</Password>
<Nonce>ksSDGF1jdfD=</Nonce>
<Created>2010-07-15T07:12:19.573Z</Created>
</UsernameToken>
</Security>
...

```

From z/VSE 4.2 onwards, z/VSE supports the following methods:

- Transport-layer authentication by using:
 - HTTP authentication (Basic and Digest Access Authorization)
 - SSL Client Authentication with HTTPS
- Message-layer authentication by using:
 - A UsernameToken (plaintext password or password digest)
 - An X.509 Certificate (BinarySecurityToken)

5.2.5 How the z/VSE host can act as the SOAP server

Figure 5-3 on page 135 shows how SOAP can be used in a CICS environment when the z/VSE host acts as the SOAP server that provides SOAP services (in terms of z/VSE environment, a CICS User Transactions).

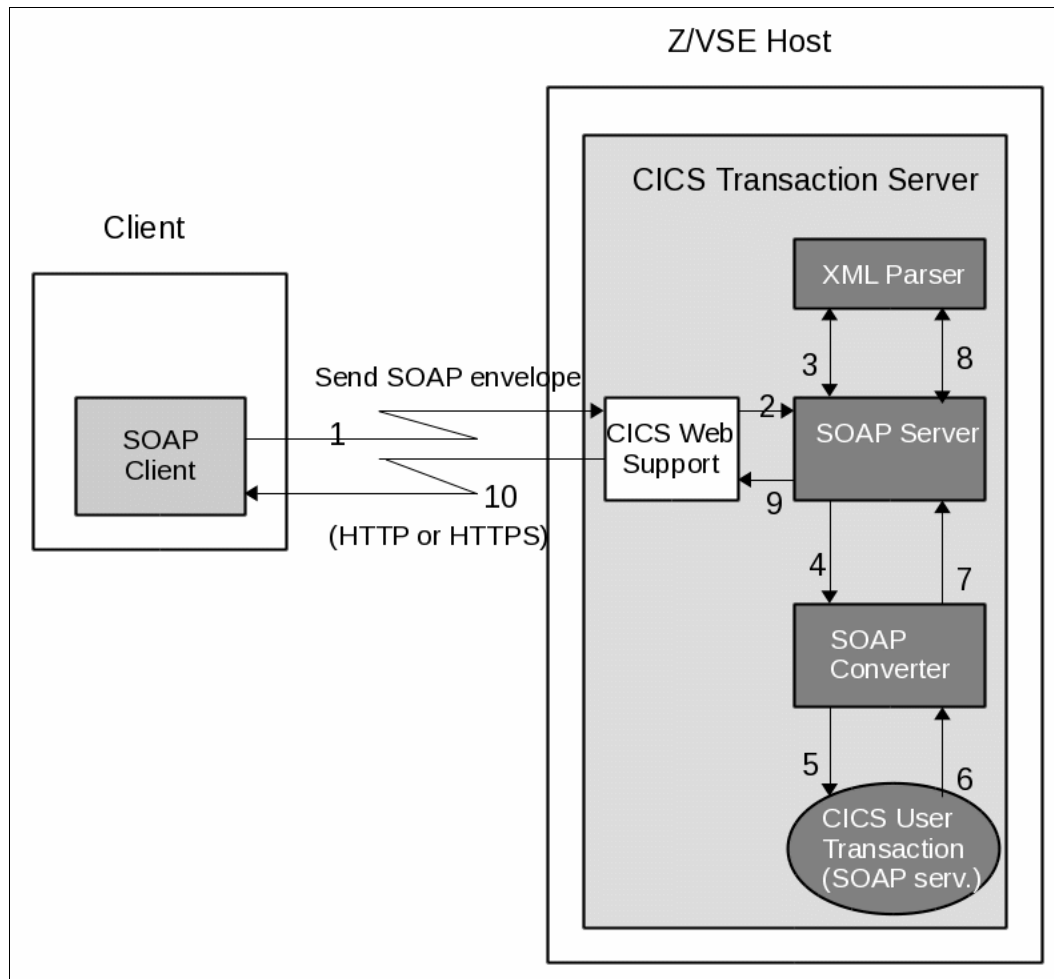


Figure 5-3 How SOAP is used when the z/VSE host acts as a SOAP server

The following steps summarize the process that is shown in Figure 5-3:

1. The SOAP client (for example, a platform that uses Microsoft .NET, IBM WebSphere Application Server, Apache SOAP, or AXIS) sends a SOAP envelope (in XML format) to the SOAP server that is running under CICS. The SOAP envelope is sent by using the CWS component of the CICS TS.
2. CWS forwards the SOAP envelope (in XML format) to the SOAP server that is running under CICS.
3. The SOAP server forwards the SOAP envelope to the XML parser, which also is running under CICS. The XML parser then parses the SOAP envelope from textual XML format into a tree representation of the data. For example, if the data is to be processed by a C program, the SOAP envelope is converted to a C program structure (with pointers) so that a C program that is running on the z/VSE host can process the data and return this parsed XML tree to the SOAP server.
4. The SOAP server forwards the parsed XML tree to the SOAP converter that is running under CICS. The SOAP converter de-serializes (decodes) the parameter subtree that is contained in the parsed XML tree and converts the parameter subtree into a binary representation. The IBM-supplied SOAP decoder is named IESSOAPD. Since z/VSE 5.2, a more powerful decoder (IESOASRV) supports data structures with arrays.

5. The SOAP converter forwards the binary representation of the parameter subtree to the CICS User Transaction that is running on the z/VSE host (the SOAP service) by using the communication area (COMMAREA) of the CICS user transaction. The CICS User Transaction then processes the data.
6. The reply is then sent from the CICS User Transaction (the SOAP service) back to the SOAP client, by reversing the process that is described in steps 6 - 10 in Figure 5-3 on page 135). The reply is sent by using the Communication area back to the SOAP converter, which serializes the parameters and returns them to the SOAP server. The SOAP server uses the XML parser to convert it from a tree-representation of the data to textual XML. The SOAP server then creates a SOAP envelope, which is then sent back (by using HTTP or HTTPS) to the SOAP client. The SOAP client can then convert the SOAP envelope to its own native data format and process the reply.

5.2.6 Using web Service Security features when z/VSE acts as the SOAP server

Consider the use of the following areas when you are using web Server Security in this situation:

- ▶ Transport-layer Encryption

When transport-layer authentication is used, z/VSE acts as an HTTP server. This authentication is implemented by using CWS as the HTTP server. CWS passes the SOAP request to the z/VSE SOAP Engine for further processing. Because CWS implements support for HTTP over SSL (HTTPS), the SOAP Engine inherits the security features from CWS. To use HTTPS, you must complete the following tasks:

- Configure TCPIP SERVICE in CICS for use with SSL.
- Create the required keys and certificates.

- ▶ Transport-layer Authentication

CWS supports SSL client authentication (HTTPS) and HTTP Basic Authentication, so the z/VSE SOAP Engine inherits the security features from CWS. To force a client to use HTTP basic authentication, you must configure the TCPIP SERVICE to use the CICS provided converter program DFH\$WBSB (specify URM=DFH\$WBSB). In addition, the z/VSE SOAP Engine extracts authentication information (user-ID and password for HTTP basic authentication or the mapped user-ID for SSL client authentication). This information can be used by the converter code to check whether transport layer authentication was used. If authentication was not used, the converter code might reject the request.

- ▶ Message-Layer Authentication

To support message layer authentication, the z/VSE SOAP Engine (that is, the z/VSE SOAP Server) extracts the authentication token from the SOAP header after parsing the XML data stream. In case of UsernameToken, the user ID and password must be verified against a local identity store. To perform this verification, the identity store must compare the plain text password of password digest against its stored password.

If user authorization also is to be performed, a user mapping must be performed to map the received user name to a z/VSE user ID. Also, a CICS SIGNON must be performed by using the mapped user to allow the transaction to run under that user.

In addition to UsernameToken, the use of certificates for authentication is possible. In this case, the converter code maps the received certificate to a z/VSE user. z/VSE supports this functionality as part of its support for SSL client authentication.

5.2.7 How the z/VSE host can act as the SOAP client

Figure 5-4 shows how SOAP can be used in a CICS environment, when the z/VSE host acts as the SOAP client.

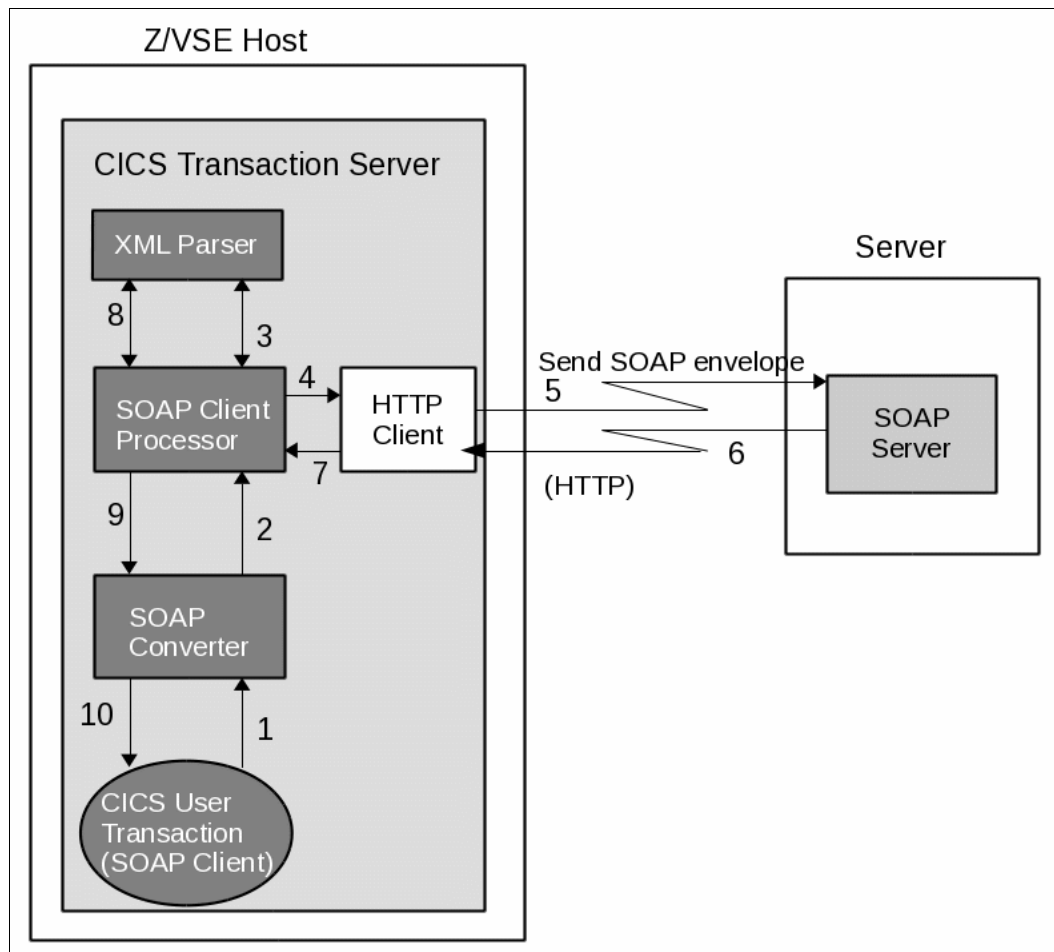


Figure 5-4 How SOAP is used when the z/VSE host acts as a SOAP client

The following steps summarize the process that is shown in Figure 5-4:

1. The CICS User Transaction that is running on the z/VSE host (the SOAP client) sends the binary representation of the parameters to the SOAP converter. This activity is done by using the communication area (COMMAREA) of the SOAP converter.
2. The encoder-part of the SOAP converter (IESSOAPE) serializes (encodes) the binary representation of the parameters into an XML tree. The SOAP converter forwards the XML tree to the SOAP client-processor that is running under CICS.
3. The SOAP client-processor generates the SOAP envelope and forwards it to the XML parser that is running under CICS. The XML parser then converts the XML tree into a textual XML format of the SOAP envelope. The XML parser returns the textual XML format to the SOAP client-processor.
4. The SOAP client-processor forwards the textual XML format to the HTTP client running under CICS.

5. The HTTP client sends the SOAP envelope (in textual XML format) to a SOAP server (for example, a platform that uses Microsoft .NET, IBM WebSphere Application Server, Apache SOAP, or AXIS) by using HTTP. The SOAP envelope can also be routed by using a SOCKS or Proxy server.
6. The reply is then sent from the SOAP server back to the CICS User Transaction (the SOAP client) by reversing the process that is described in steps 6 - 10 in Figure 5-4 on page 137). The SOAP server sends the reply back to the HTTP client, which forwards it to the SOAP client-processor. The SOAP client-processor calls the XML parser to parse the textual XML format into a tree representation. The tree is passed to the SOAP converter, which de-serializes the parameters into a binary representation and forwards them to the CICS User Transaction. The CICS User Transaction then processes the reply.

5.2.8 Using web Service Security features when z/VSE acts as the SOAP client

Consider the following areas with web Security features where z/VSE acts as the SOAP client:

► Transport-layer Encryption

When transport-layer authentication is used, z/VSE acts as an HTTP client. The HTTP client that is implemented in z/VSE then supports HTTPS. To use HTTPS, complete the following tasks:

- Specify `https://` in the URL.
- Provide a public/private key pair with certificates. For more information about how to specify the keys, see the skeleton SKSOAPOP in VSE/ICCF Library 59.

► Transport-Layer Authentication

From z/VSE 4.2 onwards, the HTTP Client supports SSL/HTTPS, so you can use SSL client authentication by using certificates. If requested, the SSL protocol can send the client's certificate to the server (service provider). If required, the server can use the client's certificate to perform authentication and authorization. In addition, HTTP basic authentication is supported by the z/VSE HTTP Client.

► Message-Layer Authentication

From z/VSE 4.2 onwards, the z/VSE SOAP Engine (that is, the z/VSE SOAP Client) supports the authentication token in the SOAP header. For the UsernameToken, the user application (or converter code) that is requesting the service must pass the user name and password to the SOAP Engine. If authentication is done by using a certificate, the certificate name must be provided. Code for passing this information can be part of the user application or converter code.

5.3 3270 bridge

The 3270 bridge facility runs 3270-based transactions in a browser without the need to change the associated application.

The following types of 3270-based programs are available:

- Basic Mapping Support (BMS)-based programs that use the MAP parameter in a terminal command
- Non-BMS based programs that use Terminal Control 3270 data streams (CEMT)

The 3270 bridge is an integral part of CICS web Support. It follows the same control flow, up to the alias transaction.

The switch to the bridge facility occurs when DFHWBTTA is detected in the incoming HTTP data stream as the program name followed by the name of the transaction to be run (see Figure 5-5).

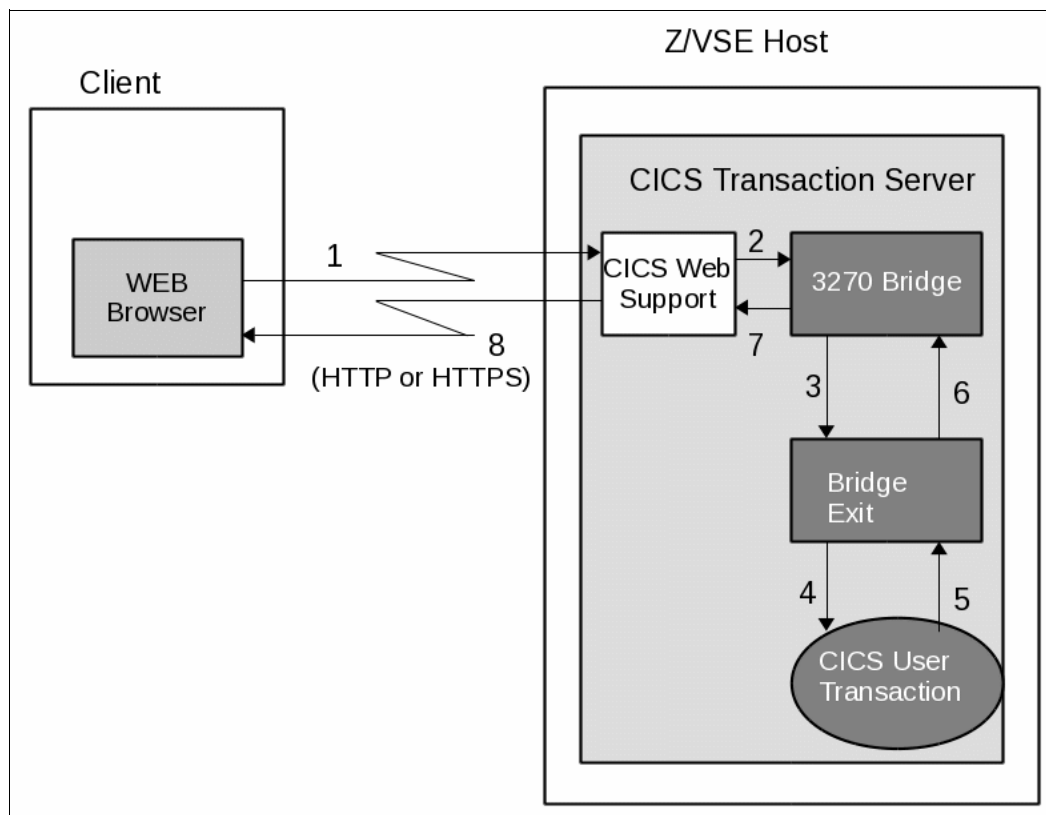


Figure 5-5 Running a transaction with CICS web Support: 3270 bridge

The following steps summarize the process that is shown in Figure 5-5:

1. The web browser sends an HTTP request.
2. The alias links to DFHWBTTA and passes the COMMAREA setup by the converter program. If no converter program was called, the COMMAREA contains the entire request.
3. DFHWBTTA finds the transaction ID for the terminal-oriented transaction from the HTTP request. DFHWBTTA then starts the 3270 user transaction immediately and assigns the CICS web bridge exit DFHWBLT.
4. For non-BMS programs, HTML conversion to 3270 data stream is done automatically. For a BMS-based program, HTML conversion is done by using an HTML template.
5. When the application program attempts to write data, the data is intercepted by CICS and given to the 3270 bridge exit, which passes it to the alias.
6. The 3270 data stream is converted to HTML by the 3270 bridge using a template that is generated out of the BMS map.
7. The HTML data is passed back to the CICS web support.
8. The web browser displays a form that corresponds to the 3270 map.

5.3.1 DFHWBTTA

The terminal translation program DFHWBTTA is a callable CICS-supplied program that provides an interface between web browsers and user 3270 CICS transactions by using the 3270 bridge facility. DFHWBTTA and its associated programs perform the translation between HTML and 3270 data streams or BMS maps. DFHWBTTA supports non-conversational, conversational, and pseudo-conversational transactions.

Note: Before you can run a BMS transaction, you must provide HTML templates that correspond to the maps you are using in the 3270 transaction. HTML templates are generated from existing BMS mapset definitions, as described in “Creating HTML templates from BMS definitions” on page 141.

Input to DFHWBTTA

If you want to run a 3270 transaction, you must specify DFHWBTTA as the program to be called. DFHWBTTA picks up the transaction ID (transid) of the application (and optional data) to execute under the 3270 bridge environment from the URL. The following URL format is required:

`http://machine.name:port/converter/alias/DFHWBTTA/transid`

If no converter is required, use “CICS” in its place. The CICS-supplied alias is “CWBA”. Also note that the transid of the application to execute under the 3270 bridge facility must follow the last slash (/), as shown in the following example:

`http://.../cics/cwba/DFHWBTTA/stat`

To pass optional data with the request, use plus signs (+) rather than blank spaces. CWS interprets a (+) sign as a blank space. The input to the user transaction ends with the first “real” blank space.

Note: After you receive your response (for example, to the specified URL), you must use the buttons that are provided in the browser output to simulate the various 3270 keys, not the keys on your keyboard.

Because you interact with the host program is by using Program Function Keys (PFKeys), it is not a good practice to use the Back or Forward buttons of your browser to move through your application’s output.

Output from DFHWBTTA

DFHWBTTA presents an HTTP response to the encode function of the converter (if any). This response is contained in a buffer that begins with a 32-bit unsigned number that specifies the length of the buffer, followed by the actual HTTP response. The HTML in the response corresponds to the output BMS map or 3270 data stream from the transaction program. This output might be customized, as described in “Customizing the generated HTML output” on page 155.

5.3.2 Programs with BMS support

In releases earlier than CICS TS 1.1.1, BMS provided the following assembler macros for defining maps:

- ▶ DFHMSD: Defines a mapset.
- ▶ DFHMDI: Defines a single map as a collection of fields.
- ▶ DFHMDF: Defines an individual field within the map.

Before you can start writing a program that uses a map, you must assemble these macros twice.

When you define TYPE=MAP on the DFHMSD macro, you can assemble and link edit a load module that is named the *physical mapset*. The physical mapset is loaded by CICS at execution time and is used to transform the application data to a 3270 data stream or vice versa.

When you specify TYPE=DSECT on the DFHMSD macro, you assemble the symbolic mapset. This mapset is a series of data structures in the language that is specified in the LANG option. A symbolic map is copied into the program and enables you to refer to the fields in the maps by name without having to know details about the physical position in the window.

Various modifications to BMS macros and new tools enable generating HTML templates from BMS maps and customizing the BMS maps to make the output more suitable to the web browser environment.

Creating HTML templates from BMS definitions

In this section, we describe how to create HTML templates from BMS mapset definitions.

For BMS programs that want to use the 3270 bridge facility, their BMS maps must be reassembled by specifying TYPE=TEMPLATE on the DFHMSD macro or by specifying SYSPARM=TEMPLATE in the parameters that are passed to the assembler. Assembling the DFHMSD macro then generates an HTML template to be used during the BMS mapping operation.

Note: The label on the DFHMSD macro is used to name the HTML templates that are produced for each map in the mapset that is processed.

No changes are required to the application program, nor is there a need to generate a DOCTEMPLATE RDO definition unless you want to specify a VSE/ESA library other than the IBM default library (DFHHTML) in which to store your templates.

Note: Installations without access to the original source code for BMS mapsets can re-create the BMS macro statements (with some limitations) by using the DFHBMSUP utility. Figure 5-6 on page 142 shows for a sample job to create MACRO format from assembled mapset. Adapt the EXTENT information in the two jobs to allocate a 15-track sized file.

```

* $$ JOB JNM=DFHBMSUP,DISP=D,CLASS=0
// JOB DFHBMSUP          RE-CREATE MACRO SOURCE FROM ASSEMBLED MAPSET
* WHEN SECURED SYSTEM, USE // ID CARD TO AUTHORIZE JOB
// OPTION PARTDUMP
* BMSOUT FILE SIZE AT LEAST 1 CYL
// DLBL BMSOUT,'BMSUP-OUTFIL',0,SD
// EXTENT SYS019,SYSWK1,1,0,63211,15
// ASSGN SYS019,DISK,VOL=SYSWK1,SHR
// ASSGN SYS009,SYSLST
// LIBDEF *,SEARCH=PRD2.CONFIG
// EXEC DFHBMSUP,SIZE=DFHBMSUP,PARM='DFHOSTM'
/*
/&
* $$ EOJ

```

Figure 5-6 Sample job to create MACRO format from assembled mapset

Use a job to generate the include book from the temporary file as shown in Figure 5-7.

```

* $$ JOB JNM=DITBMSUP,DISP=D,CLASS=0
// JOB DFHDITUP          PRINT DFHBMSUP MACRO SOURCE FILE
* WHEN SECURED SYSTEM, USE // ID CARD TO AUTHORIZE JOB
// UPSI 1
// DLBL BMSOUT,'BMSUP-OUTFIL',0,SD
// EXTENT SYS019,SYSWK1,1,0,63211,15
// ASSGN SYS019,DISK,VOL=SYSWK1,SHR
// EXEC DITTO
* DITTO SP FILEIN=BMSOUT
$$DITTO SL FILEIN=BMSOUT,LIBOUT=PRD2.CONFIG,MEMBEROUT=DFHOSTM.A
$$DITTO EOJ
/*
/&
* $$ EOJ

```

Figure 5-7 Sample job to generate include book in VSE library

Generating HTML templates from maps

Of the two methods for generating an HTML template, the easiest method is to specify SYSPARM='TEMPLATE' in the options being passed to the assembler when generating the map. The alternative method is to update the source for each mapset definition and add TYPE=TEMPLATE on the DFHMSD macro.

The Interactive Interface provides a compile skeleton to generate the HTML template (see Figure 5-8). To use that template, store the include book into the ICCF library and use the "Program Development Library" function (5-1-1) and select option 8 (compile).

IESLIBM		COMPILE JOB GENERATION	
SOURCE MEMBER:	DFHOSTM		
SOURCE TYPE.....	3	1=Online Program 3=Map Definition	2=Batch Program 4=Batch Subroutine
LANGUAGE.....	1	1=HLASM 4=C VSE	2=PL/I VSE 5=RPG II
TEMPLATE.....	1	3=COBOL VSE 6=FORTTRAN	
DB2 SERVER.....	2	1=Yes, 2=No	
DL1	3	1=Yes, 2=No	
CATALOG.....	1	1=CALL IF, 2=HLPI, 3=No	
JOBNAME.....	COMWACK	1=Yes, 2=No	
OUTPUT MEMBER.....	COMWA77_	Name of the job to generate	
PASSWORD.....		Leave blank to submit the job immediately. Enter a name and a password (optional) to save it (existing member is overwritten).	
PF1=HELP		3=END	4=RETURN

Figure 5-8 Interactive User Interface compile job generation input

In the generated job (as named in the OUTPUT MEMBER field) make the changes that are shown in Figure 5-9 for part 1 of the job).

```
* $$ JOB JNM=COMWACK,DISP=D,CLASS=A,NTFY=YES
* $$ LST DISP=D,CLASS=Q,PRI=3
// JOB COMWACK COMPILE PROGRAM DFHOSTM
// SETPARM CATALOG=1
// SETPARM HTM=1
// IF CATALOG = 1 THEN
// GOTO CAT
// OPTION NODECK,ALIGN,LIST,SYSPARM='MAP'
// GOTO GENER
/. CAT
// LIBDEF PHASE,CATALOG=LIB.SUBLIB (1)
// OPTION CATAL,NODECK,ALIGN,LIST,SYSPARM='MAP'
  PHASE DFHOSTM,*
  MODE RMODE(ANY),AMODE(31)
/. GENER
// EXEC ASMA90,SIZE=(ASMA90,64K),PARM='EXIT(LIBEXIT(EDECKXIT)),SIZE(MAXC
      -200K,ABOVE)'

  PRINT NOGEN
* $$ SLI ICCF=(DFHOSTM),LIB=(0010)
/*
// IF CATALOG NE 1 OR $MRC GT 4 THEN
// GOTO ENDM
// EXEC LNKEDT,SIZE=256K
/*
* $$ PUN DISP=I,DEST=*,PRI=9,CLASS=A
// ASSGN SYSIPT,SYSRDR
// EXEC IESINSRT
$ $$ LST DISP=D,CLASS=Q,PRI=3
#/ JOB COMWACK CATALOG MAP DFHOSTM
// EXEC LIBR
  ACCESS SUBLIB=lib.sublib (2)
  CATALOG DFHOSTM.A REPLACE=YES
* $$ END
// ON $CANCEL OR $ABEND GOTO ENDJ2
// OPTION NOLIST,ALIGN,DECK,SYSPARM='DSECT'
// EXEC ASMA90,SIZE=(ASMA90,64K),PARM='EXIT(LIBEXIT(EDECKXIT)),SIZE(MAXC
      -200K,ABOVE)'

  PRINT NOGEN
* $$ SLI ICCF=(DFHOSTM),LIB=(0010)
/*
/. ENDJ2
```

Figure 5-9 Job to create the HTML template (Part 1 of 2)

Figure 5-10 shows part 2 of the job to create the HTML template.

```
// EXEC IESINSRT
/*
#&
* $$ END
// IF HTM NE 1 THEN
// GOTO ENDM
// EXEC IESINSRT
$ $$ LST DISP=D,CLASS=Q,PRI=3
#/ JOB COMWACK CATALOG HTML DFHOSTM
// EXEC LIBR
    ACCESS SUBLIB=PRD2.DFHDOC
* $$ END
// ON $CANCEL OR $ABEND GOTO ENDJ3
// OPTION NOLIST,ALIGN,DECK,SYSPARM='TEMPLATE'
// EXEC ASMA90,SIZE=(ASMA90,64K),PARM='EXIT(LIBEXIT(EDECKXIT)),SIZE(MAXC
    -200K,ABOVE)'
    PRINT NOGEN
* $$ SLI ICCF=(DFHOSTM),LIB=(0010)
/*
/. ENDJ3
// EXEC IESINSRT
/*
#&
$ $$ EOJ
* $$ END
/. ENDM
/&
* $$ EOJ
```

Figure 5-10 Job to create the HTML template (Part 2 of 2)

Catalog the generated parts into the proper sublibrary by using the following items:

- ▶ .phase: Into PRD2.CONFIG
- ▶ A-book: Into the corresponding source library
- ▶ HTML template: Into PRD2.DFHDOC

To demonstrate the 3270 bridge facility, we used the CICS sample program that is generating BMS output to allow a user to gather performance and tuning data. The program name is DFH0STAT. The mapset name is DFHOSTM and the transid to start this program is STAT.

The first part of the source for the BMS mapset defined for our program is shown in Figure 5-11 on page 146.

DFHOSTM	DFHMSD	TYPE=DSECT,MODE=INOUT,CTRL=FREEKB,LANG=COBOL,	C
		TIOAPFX=YES,TERM=3270-2,MAPATTS=(COLOR,HILIGHT),	C
		DSATTS=(COLOR,HILIGHT)	
DFHOSTM	DFHMDI	SIZE=(24,80)	
	DFHMDF	POS=(01,20),LENGTH=38,ATTRB=(ASKIP,NORM),	C
		COLOR=BLUE,	C
		INITIAL='Sample Program - CICS Statistics Print' @P2C	
PDATE	DFHMDF	POS=(02,61),LENGTH=8,ATTRB=(ASKIP,NORM),	C
		COLOR=TURQUOISE	
PTIME	DFHMDF	POS=(02,71),LENGTH=8,ATTRB=(ASKIP,NORM),	C
		COLOR=TURQUOISE	
	DFHMDF	POS=(04,01),LENGTH=60,ATTRB=(ASKIP,NORM),	C
		COLOR=GREEN,	C
		INITIAL='Type in destination fields if required. Press EC nter to print'	
	DFHMDF	POS=(07,05),LENGTH=14,ATTRB=(ASKIP,NORM),	C
		COLOR=GREEN,	C
		INITIAL='Jobname . . . :'	
PJOBNM	DFHMDF	POS=(07,20),LENGTH=8,ATTRB=(ASKIP,NORM),	C
		COLOR=TURQUOISE	
	DFHMDF	POS=(08,05),LENGTH=14,ATTRB=(ASKIP,NORM),	C
		COLOR=GREEN,	C
		INITIAL='Applid . . . :'	
PAPPLID	DFHMDF	POS=(08,20),LENGTH=8,ATTRB=(ASKIP,NORM),	C
		COLOR=TURQUOISE	
	DFHMDF	POS=(09,05),LENGTH=14,ATTRB=(ASKIP,NORM),	C
		COLOR=GREEN,	C
		INITIAL='Sysid . . . :'	
PSYSID	DFHMDF	POS=(09,20),LENGTH=4,ATTRB=(ASKIP,NORM),	C
		COLOR=TURQUOISE	
	DFHMDF	POS=(11,05),LENGTH=14,ATTRB=(ASKIP,NORM),	C
		COLOR=GREEN,	C
		INITIAL='Node :'	
PNODE	DFHMDF	POS=(11,20),LENGTH=8,ATTRB=(UNPROT,FSET,IC),	C
		COLOR=TURQUOISE,HILIGHT=UNDERLINE	
	DFHMDF	POS=(11,29),LENGTH=1,ATTRB=(ASKIP,NORM)	
	DFHMDF	POS=(11,32),LENGTH=34,ATTRB=(ASKIP,NORM),	C
		COLOR=GREEN,	C
		INITIAL='Type in a valid Node. * is default'	
	DFHMDF	POS=(12,05),LENGTH=14,ATTRB=(ASKIP,NORM),	C
		COLOR=GREEN,	C
		INITIAL='Userid :'	
PUSERID	DFHMDF	POS=(12,20),LENGTH=8,ATTRB=(UNPROT,FSET),	C
		COLOR=TURQUOISE,HILIGHT=UNDERLINE	
	DFHMDF	POS=(12,29),LENGTH=1,ATTRB=(ASKIP,NORM)	
	DFHMDF	POS=(12,32),LENGTH=36,ATTRB=(ASKIP,NORM),	C
		COLOR=GREEN,	C
		INITIAL='Type in a valid Userid. * is default'	
	DFHMDF	POS=(13,05),LENGTH=14,ATTRB=(ASKIP,NORM),	C
		COLOR=GREEN,	C
		INITIAL='Class :'	

Figure 5-11 Define mapset DFH0STM (Part 1 of 2)

Figure 5-12 shows part 2 of defining mapset DFH0STM.

PCLASS	DFHMD F POS=(13,20),LENGTH=1,ATTRB=(UNPROT,FSET),	C
	COLOR=TURQUOISE,HILIGHT=UNDERLINE	
	DFHMD F POS=(13,22),LENGTH=1,ATTRB=(ASKIP,NORM)	
	DFHMD F POS=(13,32),LENGTH=35,ATTRB=(ASKIP,NORM),	C
	COLOR=GREEN,	C
	INITIAL='Type in a valid Class. A is default'	
	DFHMD F POS=(15,05),LENGTH=14,ATTRB=(ASKIP,NORM),	C
	COLOR=GREEN,	C
	INITIAL='TS Queue Name '	@P3A
PQUEUE	DFHMD F POS=(15,20),LENGTH=8,ATTRB=(UNPROT,FSET),	C
	COLOR=TURQUOISE,HILIGHT=UNDERLINE	@P3A
	DFHMD F POS=(15,29),LENGTH=1,ATTRB=(ASKIP,NORM)	@P3A
	DFHMD F POS=(15,32),LENGTH=35,ATTRB=(ASKIP,NORM),	C
	COLOR=GREEN,	C
	INITIAL='Type in TS Queue name, to send out-'	@P3A
	DFHMD F POS=(16,32),LENGTH=35,ATTRB=(ASKIP,NORM),	C
	COLOR=GREEN,	C
	INITIAL='put to this TS queue instead.'	@P3A
	DFHMD F POS=(17,05),LENGTH=14,ATTRB=(ASKIP,NORM),	C
	COLOR=GREEN,	C
	INITIAL='Abbreviated '	@P3A
PSHORT	DFHMD F POS=(17,20),LENGTH=1,ATTRB=(UNPROT,FSET),	C
	COLOR=TURQUOISE,HILIGHT=UNDERLINE	@P3A
	DFHMD F POS=(17,22),LENGTH=1,ATTRB=(ASKIP,NORM)	@P3A
	DFHMD F POS=(17,32),LENGTH=35,ATTRB=(ASKIP,NORM),	C
	COLOR=GREEN,	C
	INITIAL='Enter x for abbreviated TS report'	@P4C
PMSG1	DFHMD F POS=(22,01),LENGTH=79,ATTRB=(ASKIP,BRT),	C
	COLOR=NEUTRAL,CASE=MIXED	
PMSG2	DFHMD F POS=(23,01),LENGTH=79,ATTRB=(ASKIP,BRT),	C
	COLOR=NEUTRAL,CASE=MIXED	
	DFHMD F POS=(24,01),LENGTH=79,ATTRB=(ASKIP,NORM),	C
	COLOR=BLUE,	C
	INITIAL='F3=Exit to CICS	C
		,
	DFHMSD TYPE=FINAL	
	END	

Figure 5-12 Define mapset DFH0STM (Part 2 of 2)

Figure 5-13 shows the normal 3270 output when transid STAT is entered in the 3270 CICS window. (This customization is ignored with a regular 3270 data stream generation.) For more information about these macros, see “Customizing the generated HTML output” on page 155.

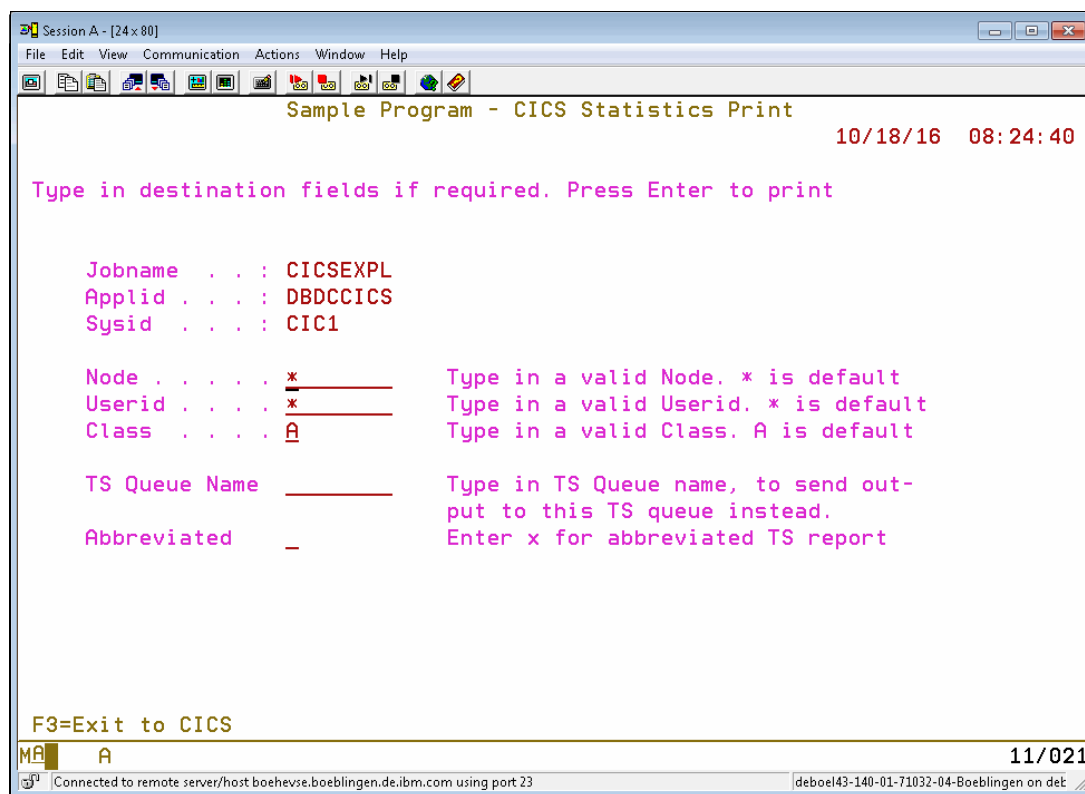


Figure 5-13 Sample DFH0STAT program window

Map output on a 3270 window

We generated our HTML template by using the information that is described in “Creating HTML templates from BMS definitions” on page 141.

Note: Use caution if you intend to modify the HTML source output because that output is case-sensitive. BMS generates buttons to represent 24 PF keys, three PA keys, and an Enter key, as shown in Figure 5-20 on page 155.

When you generate HTML templates from BMS maps, templates can be larger than 32 k. If templates are larger than 32 k, they cannot be used by the 3270 web bridge. This problem is not apparent until a transaction that uses the map is run by using the 3270 bridge. When this problem occurs, message DFHWB0133 is issued and 500 Internal Server Error is displayed on the browser.

Figure 5-14 on page 149 shows part 1 of 6 of the example generated HTML for the DFH0STM mapset.


```

<!doctype html public "-//W3C//DTD HTML 3.2//EN">
<html>
<head>
<title>CICS web Support BMS screen emulation</title>
<meta name="generator"
content="IBM_CICS_Transaction_Server/2.1.0(zVSE)">
<script language="JavaScript">
<!--
function dfhsetcursor(n)
{for (var i=0;i<document.DFH0STM.elements.length;i++)
{if (document.DFH0STM.elements[i].name == n)
{document.DFH0STM.elements[i].focus();
document.DFH0STM.DFH_CURSOR.value=n;
break}}}}
function dfhinqcursor(n)
{document.DFH0STM.DFH_CURSOR.value=n}
// -->
</script>
<style type="text/css">
<!--
.BRIGHT {font-weight: bold
}
-->
</style>
</head>
<body
onLoad="dfhsetcursor('&DFH_CURSPOSN;')"
bgcolor="#C0C0C0" text="#000000"
link="#0000FF" vlink="#800080" alink="#FF0000">
<h1>CICS web Support BMS screen emulation</h1>
<form name="DFH0STM" method="POST" action="&DFH_ACTION_URL;">
<input type="hidden" name="DFH_STATE_TOKEN" value="&DFH_STATE_TOKEN;">
<input type="hidden" name="DFH_CURSOR" value="&DFH_CURSPOSN;">
<!--DFHROW.015=001,002,004,007,008,009,011,012,013,015,016,017,022,023,0
24
-->
<!--DFHCOL.006=001,005,020,032,061,071
-->
<table>
<td height=0 width=4%>
</td>
<td height=0 width=15%>
</td>
<td height=0 width=12%>
</td>
<td height=0 width=29%>
</td>
<td height=0 width=10%>
</td>
<td height=0 width=9%>
</td>

```

Figure 5-14 Generated HTML for DFH0STAT mapset (Part 1 of 6)

Figure 5-16 shows part 2 of 6 of the example generated HTML for the DFH0STM mapset.

```
<tr>
<td colspan=2 >
</td>
<td colspan=4 nowrap>
<font color="#0000FF">
Sample Program - CICS Statistics Print
</font>
</td>
</tr>
<tr>
<td colspan=4 >
</td>
<td colspan=1 nowrap>
&F020610008_PRE;&F020610008_PDATE;&F020610008_SUF;
</td>
<td colspan=1 nowrap>
&F020710008_PRE;&F020710008_PTIME;&F020710008_SUF;
</td>
</tr>
<tr>
<td colspan=6 nowrap>
<font color="#008000">
Type in destination fields if required. Press Enter to print
</font>
</td>
</tr>
<tr>
<td colspan=1 >
</td>
<td colspan=1 nowrap>
<font color="#008000">
Jobname . . :
</font>
</td>
<td colspan=4 nowrap>
&F070200008_PRE;&F070200008_PJOBNM;&F070200008_SUF;
</td>
</tr>
<tr>
<td colspan=1 >
</td>
<td colspan=1 nowrap>
<font color="#008000">
Applid . . . :
</font>
</td>
```

Figure 5-15 Generated HTML for DFH0STAT mapset (Part 2 of 6)

Figure 5-16 shows part 3 of 6 of the example generated HTML for the DFH0STM mapset.

```
<td colspan=4 nowrap>
&F080200008_PRE;&F080200008_PAPPLID;&F080200008_SUF;
</td>
</tr>
<tr>
<td colspan=1 >
</td>
<td colspan=1 nowrap>
<font color="#008000">
Sysid . . . :
</font>
</td>
<td colspan=4 nowrap>
&F090200004_PRE;&F090200004_PSYSID;&F090200004_SUF;
</td>
</tr>
<tr>
<td colspan=1 >
</td>
<td colspan=1 nowrap>
<font color="#008000">
Node . . . . .
</font>
</td>
<td colspan=1 nowrap>
&F110200008_PRE;&F110200008_PNODE;&F110200008_SUF;
</td>
<td colspan=3 nowrap>
<font color="#008000">
Type in a valid Node. * is default
</font>
</td>
</tr>
<tr>
<td colspan=1 >
</td>
<td colspan=1 nowrap>
<font color="#008000">
Userid . . . .
</font>
</td>
```

Figure 5-16 Generated HTML for DFH0STAT mapset (Part 3 of 6)

Figure 5-17 shows part 4 of 6 of the example generated HTML for the DFH0STM mapset.

```
<td colspan=1 nowrap>
&F120200008_PRE;&F120200008_PUSERID;&F120200008_SUF;
</td>
<td colspan=3 nowrap>
<font color="#008000">
Type in a valid Userid. * is default
</font>
</td>
</tr>
<tr>
<td colspan=1 >
</td>
<td colspan=1 nowrap>
<font color="#008000">
Class . . . .
</font>
</td>
<td colspan=1 nowrap>
&F130200001_PRE;&F130200001_PCLASS;&F130200001_SUF;
</td>
<td colspan=3 nowrap>
<font color="#008000">
Type in a valid Class. A is default
</font>
</td>
</tr>
<tr>
<td colspan=1 >
</td>
<td colspan=1 nowrap>
<font color="#008000">
<q>TS Queue Name </q>
</font>
</td>
<td colspan=1 nowrap>
&F150200008_PRE;&F150200008_PQUEUE;&F150200008_SUF;
</td>
<td colspan=3 nowrap>
<font color="#008000">
Type in TS Queue name, to send out-
</font>
</td>
</tr>
<tr>
<td colspan=3 >
</td>
```

Figure 5-17 Generated HTML for DFH0STAT mapset (Part 4 of 6)

Figure 5-18 shows part 5 of 6 of the example generated HTML for the DFH0STM mapset.

```

<td colspan=3 nowrap>
<font color="#008000">
<q>put to this TS queue instead.      </q>
</font>
</td>
</tr>
<tr>
<td colspan=1 >
</td>
<td colspan=1 nowrap>
<font color="#008000">
<q>Abbreviated      </q>
</font>
</td>
<td colspan=1 nowrap>
&F170200001_PRE;&F170200001_PSHORT;&F170200001_SUF;
</td>
<td colspan=3 nowrap>
<font color="#008000">
<q>Enter x for abbreviated TS report  </q>
</font>
</td>
</tr>
<tr>
<td colspan=6 nowrap>
&F220010079_PRE;&F220010079_PMSG1;&F220010079_SUF;
</td>
</tr>
<tr>
<td colspan=6 nowrap>
&F230010079_PRE;&F230010079_PMSG2;&F230010079_SUF;
</td>
</tr>
<tr>
<td colspan=6 nowrap>
<font color="#0000FF">
<q>F3=Exit to CICS
      </q>
</font>
</td>
</tr>
</table>
<br>
<input type="submit" name="DFH_PF1" value="PF01">
<input type="submit" name="DFH_PF2" value="PF02">

```

Figure 5-18 Generated HTML for DFH0STAT mapset (Part 5 of 6)

Figure 5-19 shows part 6 of 6 of the example generated HTML for the DFH0STM mapset.

```
<input type="submit" name="DFH_PF3" value="PF03">
<input type="submit" name="DFH_PF4" value="PF04">
<input type="submit" name="DFH_PF5" value="PF05">
<input type="submit" name="DFH_PF6" value="PF06">
<input type="submit" name="DFH_PF7" value="PF07">
<input type="submit" name="DFH_PF8" value="PF08">
<input type="submit" name="DFH_PF9" value="PF09">
<input type="submit" name="DFH_PF10" value="PF10">
<input type="submit" name="DFH_PF11" value="PF11">
<input type="submit" name="DFH_PF12" value="PF12">
<br>
<input type="submit" name="DFH_PF13" value="PF13">
<input type="submit" name="DFH_PF14" value="PF14">
<input type="submit" name="DFH_PF15" value="PF15">
<input type="submit" name="DFH_PF16" value="PF16">
<input type="submit" name="DFH_PF17" value="PF17">
<input type="submit" name="DFH_PF18" value="PF18">
<input type="submit" name="DFH_PF19" value="PF19">
<input type="submit" name="DFH_PF20" value="PF20">
<input type="submit" name="DFH_PF21" value="PF21">
<input type="submit" name="DFH_PF22" value="PF22">
<input type="submit" name="DFH_PF23" value="PF23">
<input type="submit" name="DFH_PF24" value="PF24">
<br>
<input type="submit" name="DFH_PA1" value="PA1">
<input type="submit" name="DFH_PA2" value="PA2">
<input type="submit" name="DFH_PA3" value="PA3">
<input type="submit" name="DFH_CLEAR" value="Clear">
<input type="submit" name="DFH_ENTER" value="Enter">
<input type="reset" value="Reset">
<!-- The following variables are the
names of the fields that could contain the next
CICS transaction id -->
<input type="hidden" name="DFH_NEXTTRANSID.1" value="F110200008_PNODE">
<input type="hidden" name="DFH_NEXTTRANSID.2" value="F120200008_PUSERID
">
<input type="hidden" name="DFH_NEXTTRANSID.3" value="F130200001_PCLASS">
<input type="hidden" name="DFH_NEXTTRANSID.4" value="F150200008_PQUEUE">
<input type="hidden" name="DFH_NEXTTRANSID.5" value="F170200001_PSHORT">
</form>
</body></html>
```

Figure 5-19 Generated HTML for DFH0STAT mapset (Part 6 of 6)

Figure 5-20 shows the 3270 bridge generated HTML page that is displayed by a browser.

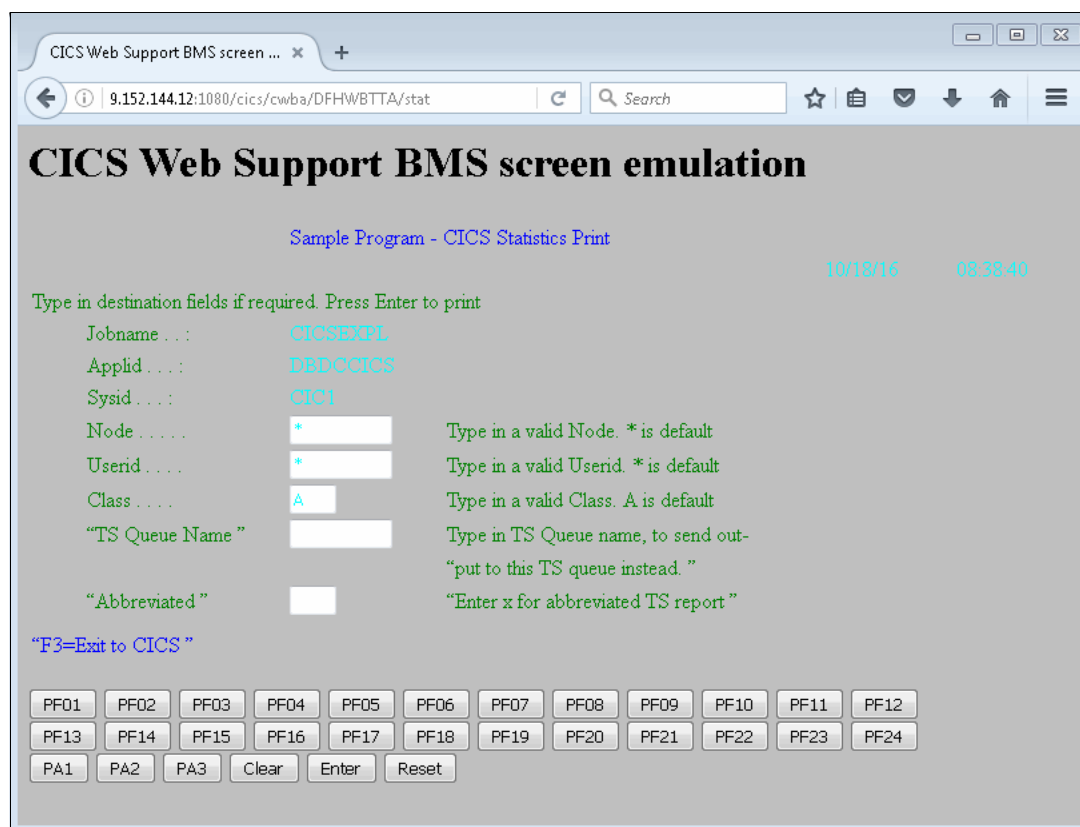


Figure 5-20 3270 bridge generated HTML page that is displayed by a browser

Customizing the generated HTML output

To improve the layout of 3270-based output for a browser, modifications can be made easily by using the following new BMS customization macros:

- ▶ DFHMDX
- ▶ DFHWBOUT

These macros are inserted into the BMS map source, but affect only on the generated HTML. The definition of a customizing macro must be written according to the rules for assembler macro definitions and must also follow the rules for assembly language macro statements.

A customizing macro definition contains the following elements (as described in Figure 5-21 on page 156):

- ▶ A MACRO statement to begin the definition.
- ▶ The name of the macro.
- ▶ Any number of invocations of the DFHMDX macro. DFHMDX is invoked from within macro DFHMSX.
- ▶ A MEND statement to end the definition.

For more information about these customizing macros, see *CICS Transaction Server for VSE/ESA internet Guide Release 1*, SC34-5765.

```

MACRO
DFHMSX
DFHMDX MAPSET=*,MAP=*,
        BGCOLOR=YELLOW,
        TEXT=BLUE,
        RESET=NO
MEND

```

Figure 5-21 Customizing a macro definition from DFH0STM

DFHMSX and DFHMDX

You can use the DFHMSX macro to define your own customization macro that is used when the templates are created from the BMS map definitions. With the DFHMDX macro, HTML templates can be customized to perform the following tasks:

- ▶ Support the application's use of keys that are not in the standard output.
- ▶ Suppress the HTML reset function, which does not correspond to any 3270 function.
- ▶ Change the appearance of the keys, or the text associated with them.
- ▶ Provide an HTML title for the HTML page.
- ▶ Provide a masthead graphic for the HTML page.
- ▶ Change the color of the background or specify a special background.
- ▶ Modify the BMS colors.
- ▶ Suppress parts of the BMS maps.
- ▶ Add web browser control functions and JavaScript functions; for example, to the HTML page.

When CICS creates the templates for each of your BMS map definitions, it invokes the DFHMSX customizing macro. Each DFHMDX macro is processed in sequence and the parameter values are stored (if applicable). Where a duplicate parameter is specified for a particular map or mapset, the new value replaces the previous value for that map or mapset. The first invocation of DFHMDX sets defaults for the values to be applied to subsequent invocations of DFHMDX by specifying an asterisk (*) for the mapset name and map name.

5.4 ECI/CICS Transaction Gateway

The CTG provides a comprehensive set of Java-based programs for access to CICS applications from a web browser. These programs include Java classes and JavaBeans for writing application-specific server programs (servlets) and browser programs (applets).

The CTG is a three-tier connector model. The middle tier often is a Netfinity or IBM RS/6000® server with WebSphere Application Server installed to provide the web server, Java virtual machine (JVM), and an application server to control the CTG program. The CTG is not based on CICS web Support.

CTG can be used in one of the following ways:

- **COMMAREA-based method**

CTG receives a request from the browser and sends a back-end request, including data in a COMMAREA that uses a CALL to the CICS Universal Client (which is also part of the CTG). The CICS Universal Client transforms the incoming request into a standardized External Call Interface (ECI) CALL and sends it with the COMMAREA down to CICS, which is known as a *Distributed Program Link* (DPL). After processing the request by CICS, the COMMAREA is sent back to CTG and, from there, data is sent to the browser.

- By using CTG's capability to act as a converter of HTML data stream to 3270 data stream and vice versa. This mechanism is based on the External Presentation Interface (EPI).

For more information about the CICS Transaction Gateway, see the [CICS Transaction Gateway website](#).

5.4.1 How the ECI and CICS Transaction Gateway are used

The CICS TG provides a CICS Java class library that includes classes that provide an application programming interface (API) and are used to communicate between the JavaGateway application and a Java application or applet.

The class JavaGateway is used to establish communication with the Gateway process and uses Java's sockets protocol. The class ECIRquest is used to specify the ECI calls that are passed to the gateway.

The multithreaded architecture of the CICS TG enables a single gateway to support multiple concurrently connected users.

The CICS TS supports the TCP/IP protocol for connections to the CICS TG, which enables the CICS Universal Client to use the ECI by using TCP/IP. However, the EPI is not supported here.

Figure 5-22 shows how the CICS TG and ECI are used in a three-tier environment.

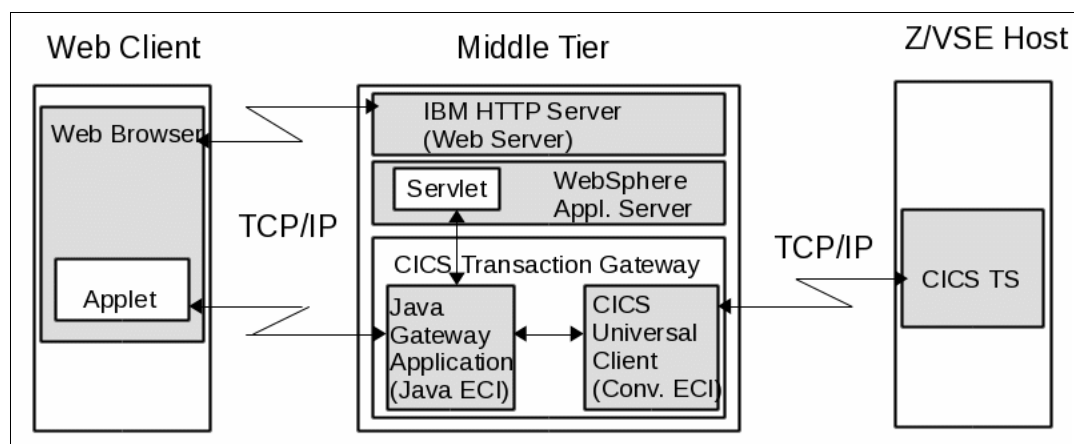


Figure 5-22 How the CICS Transaction Gateway and ECI are used

5.4.2 How the CICS Transaction Gateway accesses CICS

The flow of control when a web browser calls CICS transaction processing facilities by using the CICS Transaction Gateway (as shown in Figure 5-22 on page 157) uses the following steps:

1. The web browser calls the web server on the middle-tier by using HTTP to get HTML pages.
2. When the browser, which is interpreting the HTML and presenting it to the user, finds an applet tag, it calls the web server on the middle-tier to get the applet and the classes that it needs. It then runs the applet.
3. An applet that is going to communicate with CICS creates a JavaGateway object. Creating this object causes a call to the CICS TG long-running task on the middle tier.
4. The applet creates an ECIRRequest object to represent its request for a CICS program and calls the flow method of the JavaGateway object, which passes the instance of the ECIRRequest object.
5. The CICS TG on the middle-tier receives the request and calls the CICS program on the z/VSE host.
6. When the CICS program ends the results are returned to the web browser by using the CICS TG on the middle-tier.

The flow of data when a web browser calls CICS transaction processing facilities using the gateway, is as follows:

1. The web browser acquires data for the CICS program from the user.
2. The web browser constructs a communication area for the CICS program that is to supply transaction processing services.
3. The CICS TG on the middle-tier receives the communication area and passes it to the CICS program on the z/VSE host. The contents of the communication area are translated from ASCII (in the gateway) to EBCDIC (in the CICS TS).
4. The CICS program on the z/VSE host supplies the transaction processing services, inquiring on and perhaps changing CICS resources. If the program ends normally, changes to recoverable resources are committed. If the program ends abnormally, the changes are backed out.
5. The communication area is translated from EBCDIC to ASCII and returned to the gateway on the middle tier, which forwards it to the web browser.
6. The web browser presents information to the user.

5.4.3 External Call Interface

The ECI allows a non-CICS application to call a CICS program in a CICS server. The application can be connected to several servers at the same time. It also can have several program calls outstanding at the same time.

The CICS program cannot perform terminal I/O, but can access and update all other CICS resources. The same CICS program can be called by a non-CICS application by using the ECI or by a CICS program by using EXEC CICS LINK. Data is exchanged between the two programs by using a COMMAREA, in a similar way to CICS. The user can specify the length of the COMMAREA data to optimize performance.

Calls can be made by using one of the following methods:

- Synchronously

Synchronous calls return control when the called program completes and the information that is returned is immediately available.

- Asynchronously

Asynchronous calls return control without reference to the completion of the called program and the application can ask to be notified when the information becomes available.

Calls can also be extended. A single logical unit of work can cover two or more successive calls, though only one call can be active for each logical unit of work at any time. If it uses asynchronous calls, the application can manage many logical units of work concurrently.

The called program can update resources on its own system, it can use DPL to call CICS programs on other systems. It also can access resources on other CICS systems by function shipping, by distributed transaction processing (DTP), or (in the CICS TS for z/VSE environment) by the front end programming interface (FEPI).

For more information about the external access interfaces, see *CICS Family: Client/Server Programming*, SC33-1435.

5.5 CICS Listener

z/VSE provides a concurrent server for TCP/IP clients that can interact with TCP/IP applications that are running under CICS TS. This concurrent server is named the CICS Listener. The CICS Listener is available since VSE/ESA 2.5.

In CICS terms, the CICS Listener is a long-running CICS transaction that acts as a “switchboard”. With incoming requests from a client, the CICS Listener starts a CICS server transaction (as specified by the client). A TCP/IP communication path between the client and the CICS server transaction also is established (see Figure 5-23).

The CICS Listener supports establishing TCP/IP socket applications, which are built on the client/server model. This process results in a TCP/IP client that is communicating with and requesting services from a TCP/IP server.

The CICS Listener supports the concurrent server model. A concurrent server accepts the client's connection request and then starts a child server to handle the client's service requests. While the child server is working on the client's service requests, the concurrent server is free for other clients' connection requests.

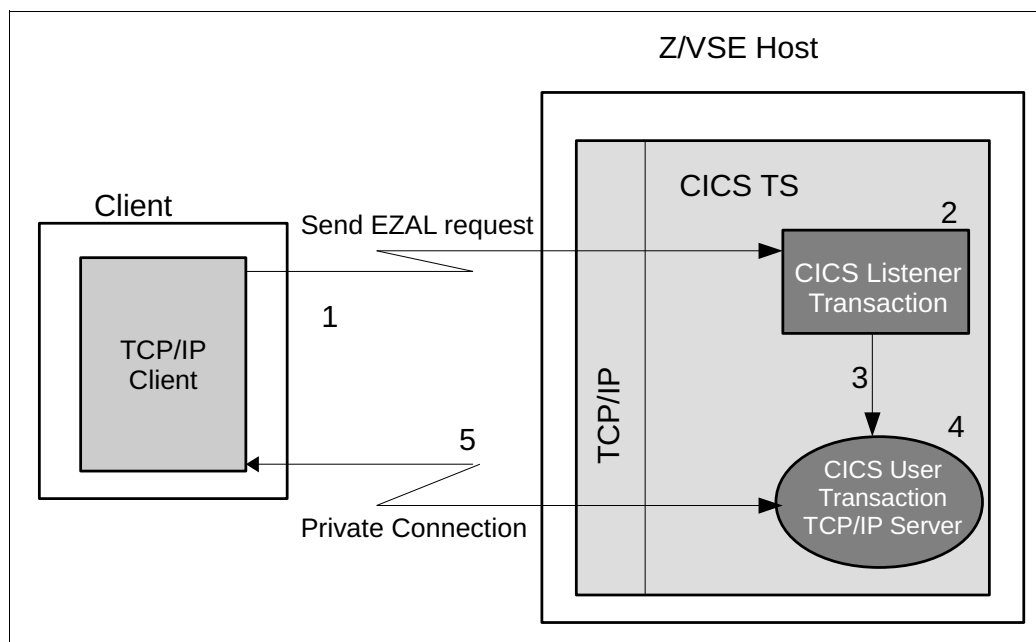


Figure 5-23 The CICS Listener in z/VSE

The following steps summarize the process that is shown in Figure 5-23:

1. TCP/IP client CONNECTs to EZAL.
2. CICS Listener calls the security exit to check the client authentication.
3. Server transaction is started.
4. Server transaction picks up the socket by TAKESOCKET request
5. Client and server communicates without the support of CICS Listener.

z/VSE supplies the CICS Listener transaction (to act as a concurrent server) and provides environmental support for this IBM-supplied listener.

This environmental support can be used by user-written listeners, if those listeners adhere to the requirements as described in *TCP/IP for VSE/ESA – IBM Program Setup and Supplementary Information*. The IBM-supplied CICS Listener is provided by using program EZACIC02, which is driven by CICS transaction EZAL. After activation, EZAL waits on connection requests from TCP/IP clients.

When a TCP/IP client successfully CONNECTed to EZAL, it must submit a data string on its first SEND request to EZAL as listed in Table 5-1.

Table 5-1 Data String for First SEND Request to EZAL

Field	Length	Required?	Description
Transaction ID	1 - 4 bytes	Yes	CICS transaction ID that the listener is going to start.
Client-in-data	1 - 35 bytes	No	First data to be passed to the child server transaction.
Startup type	2 bytes	No	Type of startup of server transaction (IC=Interval Control or TD=Transient Data).
Interval time	6 bytes	No	Interval time (hhmmss) for IC startup type.

Fields must be separated by commas.

EZAL analyzes this character string and, after having checked that the requested transaction is available, it calls the “security exit” module that is defined in the configuration file. This security exit module is provided with all of the data that is passed from the client, including the client’s IP address and port number. The security exit then can permit or to reject startup of the requested transaction.

Note: No IBM-supplied security exit module is available for the CICS Listener. For more information about writing your own security exit module, see *TCP/IP for VSE/ESA – IBM Program Setup and Supplementary Information*.

When the security exit module does not reject the startup request, the socket communication is made available to the server transaction by using a GIVESOCKET socket call and the server transaction is started. The data (without delimiters in between) that is listed in Table 5-2 is passed to the server transaction with the FROM parameter on the EXEC CICS START statement.

Table 5-2 Data String Passed with FROM Parameter

Field	Length	Description
Socket Descriptor	4 bytes	Socket descriptor to be used with TAKESOCKET request.
Listener Client ID	40 bytes	Listener’s client ID to be used with TAKESOCKET request.
Client-in-data	36 bytes	First data that is sent from client.
Socket Address	16 bytes	Socket address of own host: The 2-byte Addressing Family (AFINEY=2), the 2-byte port, and the 4-byte IP address of the host, followed by eight reserved bytes.

After the server transaction receives this data, it must pick up the socket communication by using a TAKESOCKET function call. With the successful TAKESOCKET function, the CICS Listener is posted to close its socket communication with the client. From that point, direct communication is established between the client and the server transaction.

EZAL is then again waiting on connection requests from other clients.

Messages that are generated by the IBM-supplied CICS Listener all start with prefix EZY, as described in *z/VSE TCP/IP Support* (SC34-2640-02).

5.6 Evaluation

The major characteristics of web techniques that are supported by z/VSE are listed in Table 5-3.

Table 5-3 Major characteristics of web techniques that are supported by z/VSE

Criteria	SOAP	3270 Bridge	CICS TG	CICS Listener
Tier	2	2	3	2
Protocol	HTTP	HTTP	ECI	Application defined
SSL	YES	YES	NO	YES, if Listener program uses GSK calls
Parameter Exchange	COMMAREA	COMMAREA/BMS	COMMAREA	TCP/IP Send/Receive
3270 Datastream	NO	YES	NO	NO



Channels and containers

The channels and containers concept is used for passing data between programs. Up to IBM IBM CICS Transaction Server (CICS TS) v1.1, the predominant concept is the communication area. The channels and containers approach provides an easy and flexible way for exchanging large amounts of structured data between CICS programs, whereas communication areas (COMMAREAs) are limited to 32 KB.

IBM CICS Transaction Server for z/VSE supports channels and containers through the EXEC CICS application programming interface (API) for use within CICS programs. CICS TS for z/VSE supports a subset of the functionality that is available on IBM CICS TS for z/OS.

For more information about these channels and containers, see *Using IBM CICS Transaction Server Channels and Containers*, SG24-7227.

In this chapter, we describe the concept of channels and containers and the application programming interface (API) to use this functionality. We also describe the approach that is used to convert COMMAREA applications to the new concept.

This chapter includes the following topics:

- ▶ 6.1, “COMMAREA overview” on page 164
- ▶ 6.2, “Passing data requirements” on page 165
- ▶ 6.3, “Channels and containers approach” on page 165

6.1 COMMAREA overview

The standard method of passing data from one application to another has been the in-memory process COMMAREA. The COMMAREA is limited in size to 32 KB. Before web applications were available, this approach was not a concern as data stored per record was kept deliberately small.

A COMMAREA is a facility that is used to transfer information between two programs. The caller must allocate the area within the transactions storage scope. Depending on the mechanism that is used to transfer the control to the called program, the data flow can be one-way or returning data to the caller.

A so-called “LINK” transfers control to another program and expects a return, whereas the so-called “XCTL” transfers the control to another program without returning to the caller. With a “RETURN”, a program can pass a COMMAREA to the next transaction. In the latter cases, CICS might copy the COMMAREA into a new storage area to retain the data when the caller or returning program is reclaimed.

When control is returned to the caller, the called program can insert data into the COMMAREA that is evaluated by the original caller program. The caller expresses the length of the COMMAREA as a halfword (2 bytes) binary value, which imposes a limit of 32763 bytes for the COMMAREA.

A LINK that uses a COMMAREA is shown in Figure 6-1.

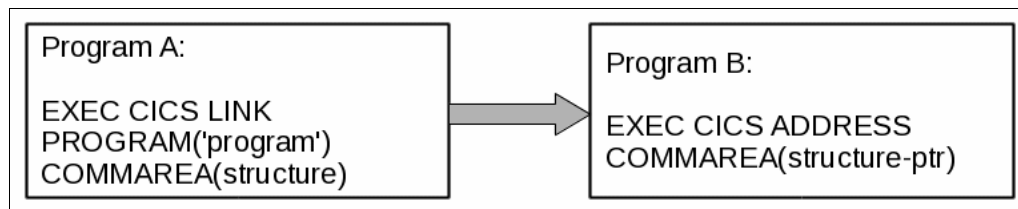


Figure 6-1 LINK that uses COMMAREA

The information in the COMMAREA can be readable data, such as names or binary data. When executing programs within the same CICS region, this distinction is irrelevant; however, passing data between different platforms might require the conversion of certain parts of the data (for example, EBCDIC to ASCII conversion). In the COMMAREA method, this approach is a cumbersome under the control of the system programmer who uses a DFHCNV table.

COMMAREA features the following characteristics:

- ▶ Required processing is low.
- ▶ COMMAREA is not recoverable.
- ▶ CICS holds a COMMAREA in CICS main storage until the user responds with the next transaction.
- ▶ A COMMAREA is available only to the first program in the next transaction, unless this program explicitly passes the data to another program or a succeeding transaction.
- ▶ The use of the COMMAREA option on the RETURN command is the principal example of a safe programming technique that you can use to pass data between successive transactions in a CICS pseudo-conversational application.

6.2 Passing data requirements

Passing data includes the following requirements:

- ▶ Memory constraints relief

Modern CICS applications are required to process large quantities of structured parameter data in Extensible Markup Language (XML) and non-XML formats, such as JavaScript Object Notation (JSON). Therefore, the data to be passed can be vastly larger than the 32 KB limit of COMMAREA.

This situation occurs because the integration of CICS applications with the elements of enterprise solutions (outside the bounds of the CICS environment) is becoming the norm. The consequence of effectively extending CICS applications to new enterprise solutions is that the constraints that are imposed on the COMMAREA size by the system might be too inflexible.

- ▶ Passing only parts back and forth

A program can call different sub programs that can require different data to work on. When a COMMAREA is used, the caller might create one layout for the passed data for simplicity reasons.

A program that is returning data to the caller can passback only a small amount of data, which indicates the results of the work but do not require to passback the originally received data; that is, the process of selecting data to be passed to another program must be flexible.

- ▶ Data conversion

This task must be done under the control of the application programmer by using APIs that enable the data transfer.

6.3 Channels and containers approach

CICS TS Version 2.1 introduced a new approach that provided an easy and more flexible mechanism for exchanging large volumes of structured parameter data between CICS programs. This new approach was provided by new capabilities that are known as channels and containers.

A *channel* is a uniquely named reference to a collection of containers. CICS provides an EXEC API that associates a named channel with a collection of one or more containers.

A *container* is a named reference to a storage area that is managed by CICS that can hold any form of application data. A container can be any size and can hold data in any format that the application requires. An application can reference any number of containers if they fit into the virtual storage of the CICS partition. CICS provides EXEC API verbs to create, delete, reference, access, and manipulate a container and to associate it with a channel.

The following requirements must be met for passing data:

- ▶ Space limit: Each container can be of any size, limited only by the amount of available storage.
- ▶ Flexibility: A program can combine different channels that comprised only the relevant data for the called program. A sub program that returns data can pass back only data that is relevant for the caller that is ignoring the input data.

- Data conversion: The API to manipulate containers converts each container separately according to its requirements. The conversion is done under programmer control and omits the cumbersome DFHCNV process.

6.3.1 General concepts

Consider the following points regarding channels and containers:

- Containers are named blocks of data that are designed for passing information between programs. You can think of them as named COMMAREAs.
- The container size is limited only by the amount of available storage.
- Containers are grouped in sets that are called channels. Programs can pass a single channel between them. You can think of a channel as a parameter list. The same channel can be passed from one program to another.
- To create named containers and assign them to a channel, a program uses the following command:

```
EXEC CICS PUT CONTAINER(container-name) CHANNEL(channel-name)
```

The program can then pass the channel and its containers to a second program by using the CHANNEL(channel-name) option of the **EXEC CICS LINK**, **XCTL**, **START**, or **RETURN** commands. Figure 6-2 shows how to pass a channel on a link.

```
EXEC CICS PUT CONTAINER(structure-name)
      CHANNEL(channel-name)
      FROM(structure)
EXEC CICS LINK PROGRAM(PROG2)
      CHANNEL(channel-name)
```

Figure 6-2 Passing a channel on a LINK

- The second program can read containers that are passed to it by using the following command:

```
EXEC CICS GET CONTAINER(container-name)
```

This command reads the named container that belongs to the channel with which the program was started (see Figure 6-3).

```
EXEC CICS GET CONTAINER(structure-name)
      CHANNEL(channel-name)
      INTO(structure)
```

Figure 6-3 Receiving a container

- If the second program is started by a **LINK** command, it can also return containers to the calling program. It can perform this process by creating containers, or by reusing containers. Figure 6-4 shows how to return a container.

```
EXEC CICS PUT CONTAINER(structure-name)
      FROM(structure)
EXEC CICS RETURN
```

Figure 6-4 Returning a container

- Channels and containers are visible only to the program that creates them and to the programs to which they are passed. When these programs end, CICS automatically deletes the containers and their storage.
- Channels and COMMAREAs EXEC CICS LINK, EXEC CICS XCTL, and EXEC CICS RETURN can pass a channel or a COMMAREA only. However, Program A can pass data in a COMMAREA to Program B, which then creates a channel to pass the data on to Program C. Because Program B receives the data that is returned from Program C in a container, Program B moves the container data into a COMMAREA and it is here that Program A expects to find it.

6.3.2 Channels

A channel is a uniquely named reference to a collection of application parameter data that is held in containers. It is analogous to a COMMAREA, but is not subject to the constraints of a COMMAREA.

You can choose a channel name that is a meaningful representation of the data structures with which the channel is associated. For example, in a human resource application, a channel name might be <employee-info>.

This collection of application parameter data serves as a standard mechanism to exchange data between CICS programs. CICS TS provides an EXEC API that associates a named channel with a collection of one or more containers, which offers an easy way to group parameter data structures that can pass to a called application. CICS TS removes a channel when it can no longer be referenced (when it becomes out of scope).

Current channel

A program's current channel (if any exist) is the channel that started it. The current channel is set by the calling program or transaction, by transferring the control to the called program through a LINK, XCTL, START, and pseudo- conversational return with the CHANNEL parameter.

Although the program can create other channels, the current channel for a particular invocation of a particular program never changes. It is analogous to a parameter list. If a channel is not explicitly specified, the current channel is used as the default value for the CHANNEL (<channel-name>) parameter on the EXEC CICS command. This process is shown in Figure 6-5 on page 168.

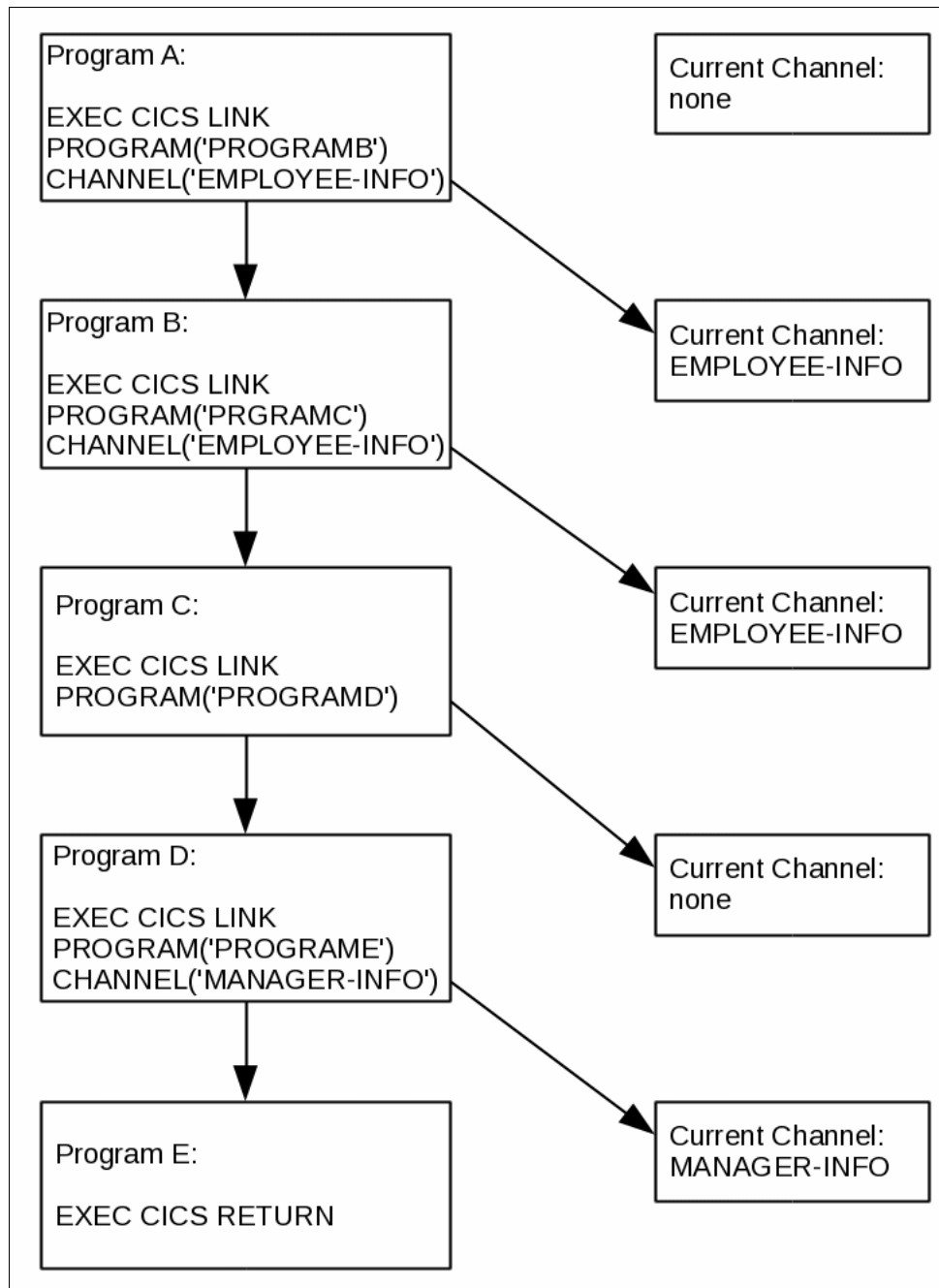


Figure 6-5 Current channel

Typically, programs that exchange a channel are written to handle that channel. Therefore, client and server programs know the name of the channel and the names and number of the containers in that channel.

However, if a server program or component is written to handle more than one channel (for example), it must discover on invocation which of the possible channels it was passed.

A program can discover its current channel (the channel with which it was started) if it issues the following command:

```
EXEC CICS ASSIGN CHANNEL
```

If there is no current channel, the command returns blanks.

The program can also retrieve the names of the containers in its current channel by browsing; however, often this process is not necessary. A program that is written to handle several channels is often coded to be aware of the names and number of the containers in each possible channel.

Important: A browse does not ensure the order in which containers are returned.

To get the names of the containers in the current channel, use the browse commands, as shown in Figure 6-6.

```
EXEC CICS STARTBROWSE CONTAINER BROWSETOKEN(data-area)
EXEC CICS GETNEXT CONTAINER(data-area) BROWSETOKEN(token)
EXEC CICS ENDBROWSE CONTAINER BROWSETOKEN(token)
```

Figure 6-6 Browsing containers in a channel

Having retrieved the name of its current channel and, if necessary, the names of the containers in the channel, a server program can adjust its processing to suit the kind of data that it was passed.

Tip: For a program that is creating a channel, the assign channel command returns blanks unless it was started by using start, link, or XCTLI, which specifies the channel name.

Channel scope

The scope of a channel is the code (the program or programs) from which you can access it.

The scope of channel is shown in Figure 6-7.

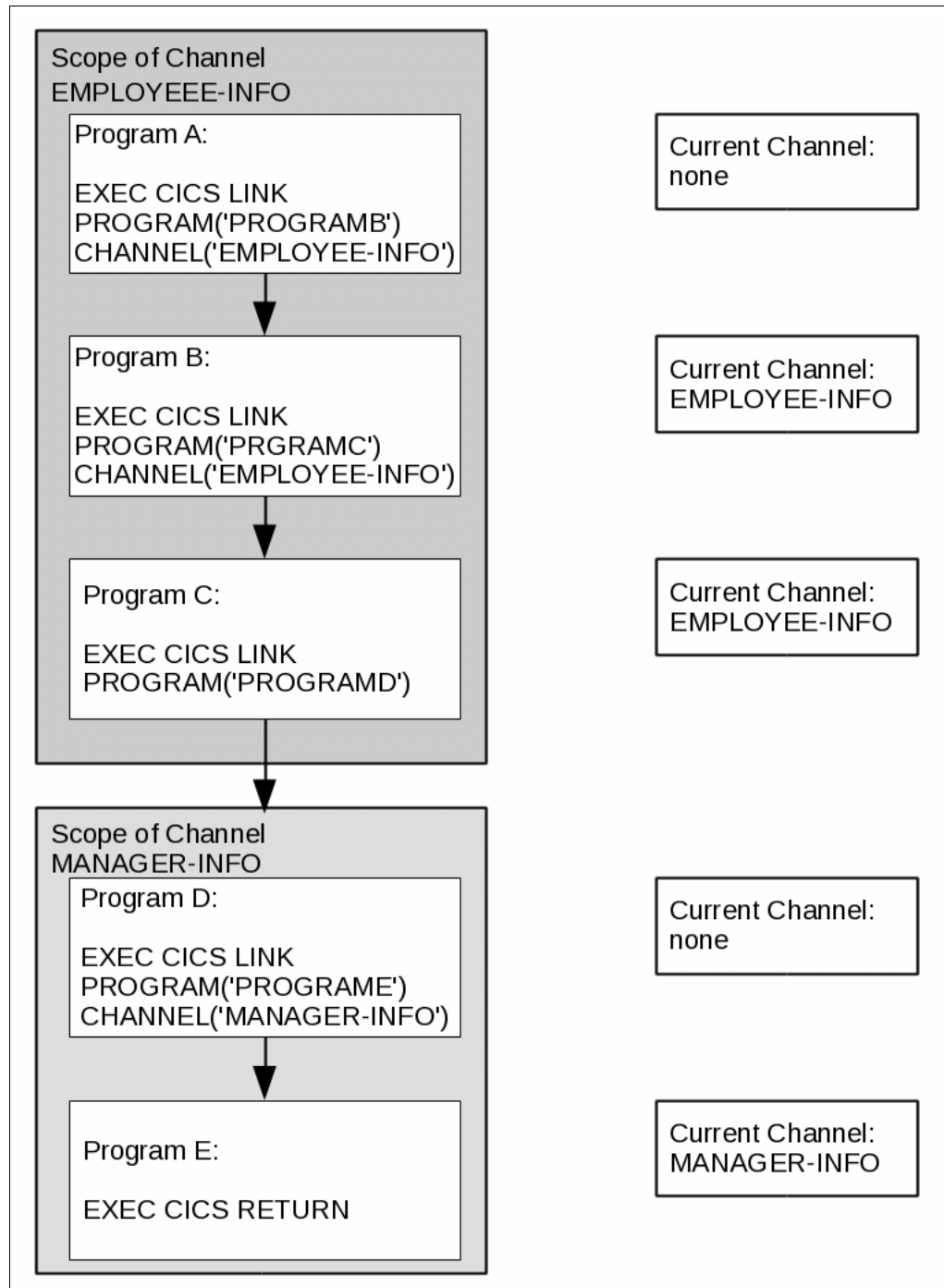


Figure 6-7 Channel scope

Figure 6-7 on page 170 shows the scope of the EMPLOYEE-INFO channel, which consists of Program A (the program that created it), Program B (for which it is the current channel), and Program C (for which it is also the current channel). Figure 6-7 also shows the scope of the MANAGER-INFO channel, which consists of Program D (which created it) and Program E (for which it is the current channel).

Lifetime of a channel

A channel is created when it is named on an EXEC CICS command. The following command is used to create a channel:

```
EXEC CICS PUT CONTAINER
```

Within this command, specifying the CHANNEL parameter creates the channel and associates the container with it.

A channel is deleted when it goes out of scope to the programs in the linkage stack, which means that no programs can access it. This action causes CICS to delete the channel.

API to manage a channel

Figure 6-8 shows the APIs that are used to create and manage a channel. The commands that are described in this section create a channel if there is no channel that is created with that name.

<pre>EXEC CICS PUT CONTAINER CHANNEL</pre> <p>Creates a channel and places data into a container within the channel</p> <pre>EXEC CICS GET CONTAINER CHANNEL</pre> <p>Retrieves the container data passed to the called program</p> <pre>EXEC CICS MOVE CONTAINER CHANNEL AS TOCHANNEL</pre> <p>Moves a container from one channel to another channel</p> <pre>EXEC CICS DELETE CONTAINER CHANNEL</pre> <p>Deletes a container</p> <pre>EXEC CICS ASSIGN CHANNEL</pre> <p>Returns the name of the program's current channel, if one exists</p> <pre>EXEC CICS LINK PROGRAM CHANNEL</pre>
--

Figure 6-8 Create and manage a channel

Links to the program on a local or remote system pass the channel and container data. The linked program can update or add new containers and pass back data to the calling program. The following command transfers control to the program passing the channel and container data:

```
EXEC CICS XCTL PROGRAM CHANNEL
```

The following command starts a task on a local or remote system, copies the named channel and containers, and passes it to the started task:

```
EXEC CICS START TRANSID CHANNEL
```

The following command returns control to CICS, passing the channel and containers to the next transaction:

```
EXEC CICS RETURN TRANSID CHANNEL
```

Channel name character set

A channel name consists of 1 - 16 characters. The following characters can be used in a channel name:

- ▶ A - Z, a - z
- ▶ 0 - 9
- ▶ Special characters, such as \$@#/%&?!:=",;<>.-_

You cannot use leading and embedded blank characters. If the supplied name includes fewer than 16 characters, it is padded with trailing blanks.

6.3.3 Containers

A container is a uniquely named block of data that can be passed to a subsequent program or transaction. It refers to a particular parameter data structure that is within a collection of virtually any form of application parameter data.

You can choose a container name that has a meaningful representation of the data structure. For example, in a human resource application, the container name might be <employee-name>.

CICS TS provides EXEC API verbs to create, delete, reference, access, and manipulate a container and to associate it with a channel (see Figure 6-9).

<pre>EXEC CICS PUT CONTAINER CHANNEL Creates a channel and places data into a container within the channel EXEC CICS GET CONTAINER CHANNEL Retrieves the container data passed to the called program EXEC CICS MOVE CONTAINER CHANNEL AS TOCHANNEL Moves a container from one channel to another channel EXEC CICS DELETE CONTAINER CHANNEL Deletes a container from a channel EXEC CICS STARTBROWSE CONTAINER Start a browse of the containers associated with a channel EXEC CICS GETNEXT CONTAINER Return the name of the next container associated to the channel EXEC CICS ENDBROWSE CONTAINER Ends the browse of the containers associated with the channel</pre>

Figure 6-9 Container-related API

A container can be any length and a container size is constrained only by the available user storage in the CICS address space. It can include data in any format that an application requires. An application can create any number of containers and use separate containers for different data types, such as binary and character data. This capability helps ensure that each container structure is based on a unique area of memory.

It also minimizes the potential for errors that commonly arise when parameter data for multiple applications is overloaded in a single memory area. The potential errors are minimized by isolating different data structures and making the association between data structure and purpose clear.

6.3.4 CICS read-only containers

CICS can create channels and containers for its own use and pass them to user programs. In some cases (in particular when CICS uses containers for security information), CICS marks these containers as read-only. This condition is used so that the user program cannot modify data that CICS requires on return from the user program.

User programs cannot create read-only containers.

You cannot overwrite, move, or delete a read-only container. Therefore, if you specify a read-only container on a put container, move container, or delete container command, you receive an INVREQ condition.

Currently, CICS TS for z/VSE does not create read-only containers.

6.3.5 Data conversion

There are two types of containers available: Non-character data (BIT) and character data (CHAR). CHAR containers enable application programs to convert data between different encodings. The data conversion model that channel applications use is much simpler than the model that COMMAREA applications use. BIT containers identify binary data that cannot be converted.

This situation occurs because the system programmer controls the data conversion in COMMAREA applications. However, the application programmer controls the data conversion in channel applications by using simple API commands.

The following cases are examples of when data conversion is necessary:

- ▶ When character data is passed between platforms that use different encoding standards; for example, Extended Binary Coded Decimal Interchange Code (EBCDIC) and American Standard Code for Information Interchange (ASCII).
- ▶ When you want to change the encoding of some character data from one coded character set identifier (CCSID) to another.

Applications that use channels to exchange data use a simple data conversion model. Frequently, no conversion is required. When conversion is required, you can use a single programming instruction to tell CICS to handle it automatically.

Channel applications use the data conversion model, which is much simpler than the one that the COMMAREA applications use. The data in channels and containers is converted under the control of the application programmer by using API commands.

The application programmer supports the following processes:

- ▶ The application programmer is responsible only for the conversion of user data, which is the data in containers that the application programs create. CICS converts the system data automatically, where it is necessary.
- ▶ The application programmer is concerned only with the conversion of character data. The conversion of binary data, between big-endian and little-endian, is not supported.
- ▶ Applications can use the container API as a simple means of converting character data from one code page to another. Figure 6-10 on page 174 shows converting data from codepage1 to codepage2.

For data conversion purposes, CICS recognizes the following types of data:

- **CHAR** (character data)

Character data is a text string. The data in the container is converted (if necessary) to the code page of the application that retrieves it. If the application that retrieves the data is a client on an ASCII-based system, it is an ASCII code page. If it is a CICS TS for z/VSE application, it is an EBCDIC code page.

All of the data in a container is converted as though it was a single character string. For a single-byte character set (SBCS) code pages, a structure that consists of several character fields is equivalent to a single-byte character string. However, for double-byte character set (DBCS) code pages, this configuration is not the case. If you use DBCS code pages, you must put each character string into a separate container to ensure that data conversion works correctly.

- **BIT** (all non character data)

BIT is everything that is not designated as being of type CHAR. The data in the container cannot be converted. This value is the default value.

Use the **DATATYPE(CHAR)** option of the **PUT CONTAINER** command to specify that a container holds character data that is eligible for conversion. If the client and server platform are different, the **GET CONTAINER** command converts the data automatically.

Tip: When the API that is shown in Figure 6-10 is used, remember to add a temporary channel, such as CHANNEL(temp). If you fail to add this temporary channel, you are relying solely on the program to be called along with a channel.

Also, if the data is large, you must perform a delete container operation immediately.

As shown in Figure 6-10, the container API is explicitly used to convert character data from one code page to another.

```
EXEC CICS PUT CONTAINER(temp) DATATYPE(CHAR)
      FROMCCSID(codepage1) FROM(input-data)
EXEC CICS GET CONTAINER(temp) INTOCCSID(codepage2)
      SET(data-ptr) FLENGTH(data-len)
```

Figure 6-10 API to convert code page

6.3.6 Benefits of using channels and containers

The lifecycle and scope of channels and containers are controlled of the CICS system, which ensures data integrity and storage resource management.

The use of the capabilities of the channels and containers methodology results in the following benefits for applications:

- An unconstrained, CICS supported method of passing parameter data.
- Segregation of parameter data structures, each part represented by a named container structure.
- A loose functional coupling approach.
- The freedom to dynamically determine the nature of the passed data and to select the appropriate processing required.

- ▶ A CICS standard API for optimized exchange of data between CICS programs that are implemented in any CICS supported language.
- ▶ CICS managed lifecycle for channels and containers resources.
- ▶ Ease of parameter passing by use of unique named references.
- ▶ Ease of understanding by use of unique named references to parameter payload.
- ▶ Explicit code page conversion operations.

The internal CICS implementation of channels and containers is optimized for efficient memory management and data transfer. CICS ensures that only the necessary new and modified containers are transferred between the calling applications. This approach optimizes the performance of the calling mechanism.

Assuming that the containers separate the different parameter structures, the calling applications benefit from complete access to the data content in all containers that are in scope.

6.3.7 Porting COMMAREA to channels and containers

To port programs that are exchanging data through a COMMAREA on a link command, the format of the command must be changed. Proper commands also must be added to use channels and containers, as shown in Figure 6-11.

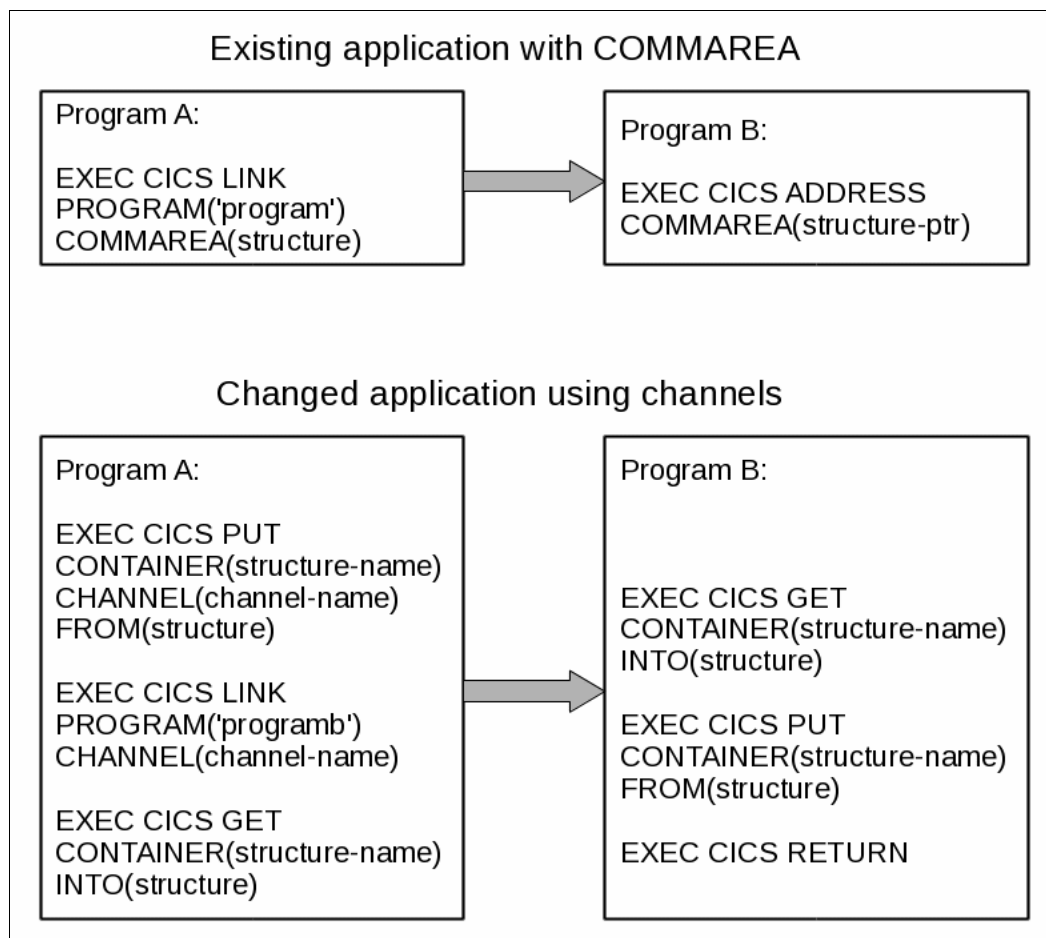


Figure 6-11 Changes from COMMAREA to channels using LINK

The same concept applies to programs that use the start command with the COMMAREA. Figure 6-12 shows an example of the changes that are necessary to convert an application program that uses an **EXEC CICS START** command with data to start passing a channel. The example also shows the commands that must be added or changed.

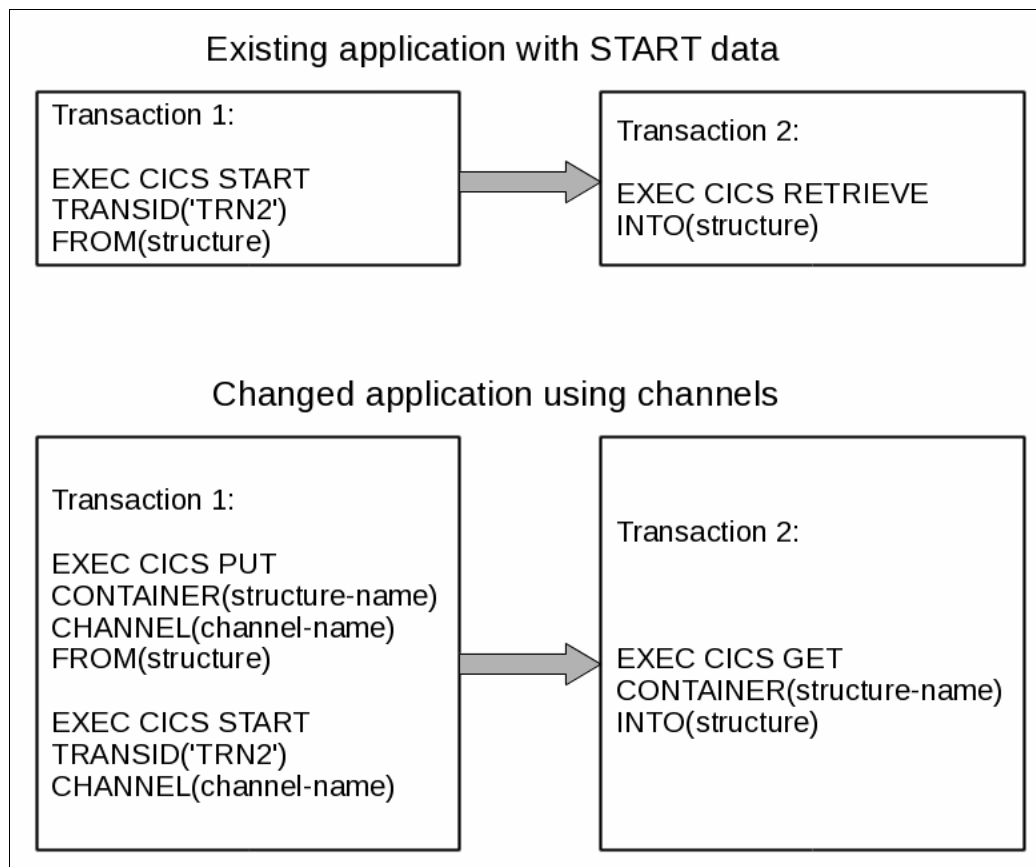


Figure 6-12 Changes from COMMAREA to channel using START

A program can issue multiple starts with data for a single transaction ID. CICS starts one instance of the transaction. The program can issue multiple retrieves to get the data. When the channel option is used on the start, CICS starts one transaction for each start request. The started transaction can access the contents of a single channel and get a copy of the channel.

Consider the following points when you are porting from a COMMAREA to channels and containers:

- ▶ CICS application programs that use traditional COMMAREAS to exchange data continue to work as before.
- ▶ EXEC CICS LINK and EXEC CICS START commands, which can pass either COMMAREAs or channels, can be dynamically routed.
- ▶ It is possible to replace a COMMAREA by using a channel with a single container. Although this method might seem the simplest way to move from COMMAREAs to channels and containers, it is not a good practice to use. The process that is used to replace the COMMAREA structure should include the following suggestions:
 - Use separate containers for input and output.
 - Use a dedicated container for error information.

- Use separate containers for each structure.
 - Use separate containers for CHAR and BIT data.
- ▶ Be aware that a channel might use more storage than a COMMAREA that is designed to pass the same data. Because you are taking the time to change your application programs to use this new function, you must implement the best practices for channels and containers.
 - ▶ Channels have several advantages over COMMAREAs and it is advantageous to design your channels to make the most of these improvements.
 - ▶ In previous releases, some applications used TS queues to pass more than 32 KB of data from one program to another because the size of COMMAREAs is limited to 32 KB and channels were not available. Typically, this process involved multiple writes to and reads from a TS queue. If you port one of these applications to use channels, be aware of the following points:
 - If the TS queue that is used by your application is in main storage, the storage requirements of the new, ported application are likely to be similar to the requirements of the application.
 - If the TS queue that is used by your application is in auxiliary storage, the storage requirements of the ported application are likely to be greater than the requirements of the application. The extra storage requirement is necessary because container data is held in storage rather than being written to disk.

6.3.8 Converting COMMAREA to channels example

In this section, we present a basic example for converting COMMAREA to channels.

COMMAREA implementation

In this section, we describe how to replace a COMMAREA by using a channels and containers solution. The first part of Figure 6-13 shows the classic COMMAREA solution. A program calls a subroutine that provides two variables that are grouped into a COMMAREA structure. The first variable is treated as the input field for the subroutine and the second variable receives the output the called program creates.

```
01 MYCOMMAREA.
03 LNAME PIC X(40).
03 PNAME PIC X(40).
01 PHONETIC PIC X(8) VALUE IS 'PHONETIC'.
.
.
MOVE 'MEYER' TO LNAME.
EXEC CICS LINK PROGRAM (PHONETIC)
COMMAREA(MYCOMMAREA)
END-EXEC.
DISPLAY PNAME.
*** PNAME CONTAINS 'MAYR'.
```

Figure 6-13 COMMAREA example

The second part of Figure 6-13 shows a name as input. The subroutine changes the name into a phonetic string and returns this string to the caller. For example, if you call the subroutine with different names (all of which sound the same but are spelled differently), the subroutine returns the same phonetic string for all names.

For example, the subroutine returns MAYR when it is called for Meyer, Mayer, Meier, or Maier. Figure 6-13 on page 177 is a program fragment that shows how to call the subroutine PHONETIC with a COMMAREA.

Channel implementation

Figure 6-14 shows how to transform the small piece of code into a channels and containers solution. In Figure 6-14, you can see that the input to the subroutine can now be treated as separate objects, meaning that the two variables can be defined independently. We advise that you perform this data separation, although it is still possible to send both variables in one container.

```
01 LNAME PIC X(40).
01 PNAME PIC X(40).
01 INPUTCONTAINER PIC X(16) VALUE IS 'MYINPUTCONTAINER'.
01 OUTPUTCONTAINER PIC X(16) VALUE IS 'MYOUTPTCONTAINER'.
01 CHANNELNAME PIC X(16) VALUE IS 'MYCHANNELNAME'.
01 CHANNELNAME
PIC X(16) VALUE IS 'MYCHANNELNAME'.
PHONETIC PIC X(8) VALUE IS 'PHONETIC'.
.
MOVE 'MEYER' TO LNAME.
EXEC CICS PUT CONTAINER(INPUTCONTAINER)
FROM(LNAME)
CHANNEL(CHANNELNAME)
END-EXEC.
EXEC CICS LINK PROGRAM (PHONETIC)
CHANNEL(CHANNELNAME)
END-EXEC.
EXEC CICS GET CONTAINER(OUTPUTCONTAINER)
INTO(PNAME)
CHANNEL(CHANNELNAME)
END-EXEC.
DISPLAY PNAME.
*** PNAME CONTAINS 'MAYR'.
```

Figure 6-14 Channels and containers example

When subroutines are called from various programs that use both COMMAREA and channels techniques, the subroutines must recognize which technique is being used in the current invocation. We suggest the use of the IBM CICS API **EXEC CICS ASSIGN CHANNEL** command to check whether a channel is used.

If the returned value is not equal to spaces, a channel was passed and you can process it; otherwise, use a COMMAREA.

It is feasible to perform the check if you reverse the process. Checking the EIBCALEN field for greater than zero indicates that this version is a COMMAREA version. However, we do not recommend this process because if it is a pseudo conversational program, the COMMAREA length is always zero on the first invocation. Therefore, you cannot trust this result as an indication for COMMAREA or channels technique in this case.

In Figure 6-15, we show the backend program *phonetic*, which was written to use both techniques.

```
WORKING-STORAGE SECTION.
01 CURRENTCHANNELNAME PIC X(16).
01 MYBROWSETOKEN PIC 9(8) BINARY.
01 INPUTCONTAINER PIC X(16) VALUE IS 'MYINPUTCONTAINER'.
01 OUTPUTCONTAINER PIC X(16) VALUE IS 'MYOUTPTCONTAINER'.
01 LASTNAME PIC X(40).
LINKAGE SECTION.
01 DFHCOMMAREA.
03 LNAME PIC X(40).
03 PNAME PIC X(40).
PROCEDURE DIVISION USING DFHCOMMAREA.
EXEC CICS ASSIGN CHANNEL(CURRENTCHANNELNAME)
END-EXEC.
IF CURRENTCHANNELNAME IS EQUAL TO SPACES
THEN
*
/* CONTINUE WITH COMMAREA */
CONTINUE
ELSE
* GET THE CONTAINER NAME
EXEC CICS STARTBROWSE CONTAINER
CHANNEL(CURRENTCHANNELNAME)
BROWSETOKEN(MYBROWSETOKEN)
END-EXEC
EXEC CICS GETNEXT CONTAINER(INPUTCONTAINER)
BROWSETOKEN(MYBROWSETOKEN)
END-EXEC
EXEC CICS ENDBROWSE CONTAINER
BROWSETOKEN(MYBROWSETOKEN)
END-EXEC
EXEC CICS GET CONTAINER(INPUTCONTAINER)
CHANNEL(CURRENTCHANNELNAME)
INTO(LASTNAME)
END-EXEC
END-IF
```

Figure 6-15 Decide channel or COMMAREA

As shown in Figure 6-15, the *STARTBROWSE* example works only when the subroutine is called with one container in the channel.

We advise that you access containers directly, as shown in the Figure 6-16. A STARTBROWSE loop makes sense, where a subroutine is written to handle more than one channel. Therefore, it must discover which of the possible channels it was passed.

```

WORKING-STORAGE SECTION.
01 L PIC 9(8) BINARY.
01 LASTNAME PIC X(40).
01 RC PIC 9(8) BINARY.
01 INPUTCONTAINER PIC X(16) VALUE IS 'MYINPUTCONTAINER'.
01 CURRENTCHANNELNAME PIC X(16).
LINKAGE SECTION.
01 DFHCOMMAREA.
03 LNAME PIC X(40).
03 PNAME PIC X(40).
PROCEDURE DIVISION USING DFHCOMMAREA.
EXEC CICS ASSIGN CHANNEL(CURRENTCHANNELNAME)
END-EXEC.
IF CURRENTCHANNELNAME IS EQUAL TO SPACES
THEN
*
/* CONTINUE WITH COMMAREA */
CONTINUE
ELSE
* GET THE CONTAINER NAME
EXEC CICS GET CONTAINER(INPUTCONTAINER)
CHANNEL(CURRENTCHANNELNAME)
INTO(LASTNAME)
FLENGTH(L)
RESP(RC)
END-EXEC
IF RC NOT = DFHRESP(NORMAL)
THEN
*
*** DO SOME ERROR PROCESSING HERE ***
CONTINUE
END-IF
END-IF.

```

Figure 6-16 Get container directly

If the container name that the subroutine uses is known, you can access the container directly. If you know that you are always using channels, the assign channel statement is not required. In this case, the channel that is used to start the subroutine is known as the current channel. Although the subroutine might create other channels, the current channel does not change.

A main program that is started for the first time does not have a current channel. You must reinvoke this program by using an **EXEC CICS RETURN TRANSID** command with a channel for this main program to have a current channel. Subroutines have a current channel when they are started with a channel.

Figure 6-16 also shows the process that you can use to get the input container directly without the use of a browse loop. In this case, the name must be standard and not changed. Always check the response code for unexpected events; for example, CONTAINERERROR, which means that the specified container cannot be found.



CICS customization

IBM CICS provides the following interfaces to allow you to access CICS control blocks, enhance or modify CICS, and customize the product for special needs and functions:

- ▶ System generation
- ▶ Initialization and termination processing
- ▶ User exits
- ▶ User-replaceable modules
- ▶ System programmer interfaces

Source and object compatibility is not always ensured for most of these interfaces when you migrate from one CICS release to another. You might have to recompile and rework the logic that you implemented.

If you have programs that directly access control blocks, modify CICS code, or use any other undocumented interfaces, you must rewrite these programs to the supported interfaces that are described in *CICS Transaction Server for VSE/ESA Customization Guide*, SC33-1652.

In this chapter, we describe these interfaces. For more information about new and changed options that affect customizing IBM CICS Transaction Server, see *CICS Transaction Server for VSE/ESA Migration Guide*, GC33-1646, and *CICS Transaction Server for VSE/ESA Customization Guide*.

This chapter includes the following topics:

- ▶ 7.1, “System generation” on page 184
- ▶ 7.2, “Initialization and termination processing” on page 184
- ▶ 7.3, “User exits” on page 186
- ▶ 7.4, “Global user exits” on page 186
- ▶ 7.5, “User-replaceable modules” on page 187
- ▶ 7.6, “System programmer interfaces” on page 189

7.1 System generation

You cannot generate CICS management modules in CICS Transaction Server (CICS TS). The DFHSG macros (used in CICS/VSE) to perform tailoring and system generation of CICS management modules are no longer provided. The generation sublibrary PRD2.GEN1 supports only system generation for the z/VSE supervisor.

Most of the CICS management modules are pre-generated and distributed as object code only (OCO). Because most of the CICS nucleus code loads above the 16 MB line, the size of full-function modules does not have the affect on the amount of available storage below the 16 MB line that it did in CICS/VSE.

You must use PLT initialization processing or the exit programming interface (XPI) to implement any modifications that you made to CICS management modules in CICS/VSE.

7.2 Initialization and termination processing

In this section, we describe the changes to the initialization and shutdown processing.

7.2.1 System initialization overlays (CICS/VSE)

You cannot use system initialization overlays to perform extra processing during CICS start. The SIMODS system initialization parameter is no longer supported.

7.2.2 Program list table programs

Programs that are to run during CICS initialization or shutdown must be defined in a program list table (PLT). The PLT must be named on the program list post initialization (PLTPI) or the program list shutdown (PLTSD) system initialization parameter. In this section, the PLT processing is described.

PLTPI processing

Programs in the PLTPI are processed in separate stages of CICS initialization, similar to the PLTSD. The PLTPI is divided into two parts by the PLT entry, as shown in the following example:

```
DFHPLT TYPE=ENTRY,PROGRAM=DFHDELIM
```

Programs that are listed before the DFHDELIM entry are processed in the second stage of CICS initialization. These programs can be used to enable user exits only; they must be written in assembler language, and they must run in AMODE(31).

Programs that are listed after the DFHDELIM entry are processed in the third stage of CICS initialization. This part is where you include the PLT entries from PLT tables in previous releases. The VSE-supplied PLT skeletons in ICCF library 59 have all the PLT entries after the DFHDELIM entry.

PLTSD processing

Shutdown PLT processing is essentially the same as in previous releases. Program entries that are listed before the DFHDELIM entry in the PLT are processed during the first quiesce stage of shutdown. Programs entries that are listed after the DFHDELIM entry are processed in the second quiesce stage of shutdown.

PLT program execution key

It is important to understand the execution key (EXECKEY) that is defined for programs that are used during PLT processing. Consider the following points:

- ▶ PLTPI programs that are started during initialization and PLTSD programs that are started by CEMT PERFORM SHUTDOWN run under tasks with TASKDATAKEY(CICS).
- ▶ Programs must run in EXECKEY(CICS) if they are started by a transaction that is defined with TASKDATAKEY(CICS). CICS passes control to each PLT program in EXECKEY(CICS), even if the program's resource definition online (RDO) definition specifies EXECKEY(USER).
- ▶ If your PLTPI or PLTSD program links to other programs, those programs must be defined with EXECKEY(CICS); otherwise, they abend with an AEZD abend.

We recommend that you define PLTPI and PLTSD programs with EXECKEY(CICS) to avoid this problem. This definition includes any programs to which your PLT programs link. The default is EXECKEY(USER) for RDO and PPT program definitions that are migrated from previous releases to CICS TS.

Table 7-1 lists the recommended usage for combinations of TASKDATAKEY on the transaction definition and EXECKEY on the program definition.

Table 7-1 TASKDATAKEY and EXECKEY combinations

TASKDATAKEY	EXECKEY	Recommended usage
USER	USER	For normal applications that use the CICS API.
CICS	USER	Not permitted. CICS abends any program that is defined with EXECKEY(USER) started under a transaction that is defined with TASKDATAKEY(CICS).
USER	CICS	For programs that must issue restricted VSE requests or modify CICS key storage.
CICS	CICS	For transactions (and component programs) that function as extensions to CICS, such as the CICS supplied transactions, or which require the same protection.

7.3 User exits

CICS TS continues to provide the following types of user exit interfaces as in previous releases:

- ▶ **Global user exit**
An exit interface that is started at specific processing points in CICS modules and domains.
- ▶ **Task-related user exit**
An exit interface that is used primarily for CICS applications to access external resource managers, such as DB2 Server for VSE.

7.4 Global user exits

The restructuring of CICS TS resulted in the following significant changes to the global user exit (GLUE) interface:

- ▶ Standard linkage conventions and interfaces are used for all GLUEs.
- ▶ An XPI provides a set of calls to selected CICS domains that service the request.
- ▶ More exits can use command-level functions.
- ▶ A total of 35 new exit points are available in CICS TS.
- ▶ The following exits were removed in CICS TS:
 - File Control: XFCIN, XFCINC, and XFCOUT
 - Task Control: XKCDISP, XKCBWT, and XKCAWT
 - Storage Control: XSCREQ
 - Terminal Control: XTCDAT
- ▶ The remaining 48 exit points that are available in previous releases (CICS and VSE) are still supported. GLUEs that are implemented on previous versions of CICS are not source-compatible and object-compatible on CICS TS. If you implemented any of these GLUEs, the exit programs must be reviewed, changed to work on CICS TS, and recompiled by using the CICS TS libraries.

If you have any GLUEs to migrate from CICS/VSE to CICS TS, you must alter these GLUEs to conform to the new exit standards, to be reentrant, and to run in a 31-bit environment.

Specify EXECKEY(CICS) in your RDO program definitions for your GLUE programs and any programs to which they pass control. This specification must be done for the same reasons that are described in “PLT program execution key” on page 185. GLUE programs are started in CICS-key.

For more information about implementing GLUEs and the new XPI, see *CICS Transaction Server for VSE/ESA Customization Guide*, SC33-1652.

Changes because of channel and container support

Certain GLUEs are passed the channel name of the application program, if any. Channels are created by application programs and are passed to programs that they call.

New fields that contain channel information are appended at the end of the DSECT. The storage area that is passed to the exit and mapped by the DSECT is allocated by CICS TS. Therefore, passing the new fields is not apparent to user exits. The user exit programs must be recompiled only if the channel-related fields are to be used within the exit program. To recompile, use the macros that are shipped in PRD1.BASE.

Channel and Container information are passed to following GLUE programs:

- ▶ XPCABND
- ▶ XPCTA
- ▶ XPCHAIR
- ▶ XPCFTCH
- ▶ XPCREQ
- ▶ XPCREQC
- ▶ XALCAID
- ▶ XICEREQ
- ▶ XICEREQC

7.4.1 Task-related user exits

Task-related user exits (TRUEs) are unchanged. The only parameters that might cause a problem are UEPCSA and UEPTCA because the CSA and TCA control blocks are not addressable.

The release of IBM DB2 for VSE shipped with z/VSE 6.1 was updated and provides the required TRUE support.

7.5 User-replaceable modules

A user-replaceable module (URM) is a CICS supplied program that is always invoked at a specific point in CICS processing. You can modify the supplied program with your own logic or replace it with your own version.

CICS TS (compared to CICS/VSE) changed the following guidelines for writing URMs:

- ▶ They must be written by using command-level interfaces.
- ▶ They must not access internal CICS control blocks. CICS passes information that is required by URMs in a communication area (COMMAREA), rather than in control block fields as in previous releases.
- ▶ They can be written in any language that is supported by CICS. (However, the journaling URMs, DFHXJCC, and DFHXJCO, must be written in assembler language.)
- ▶ They must be link-edited as AMODE(31) and support 31-bit addressing, including any programs that the URMs call.

You must review all of your URMs to ensure that they conform to the new guidelines.

Some URMs from earlier CICS versions can be source-compatible; however, they must be reassembled by using the CICS TS libraries.

The following URMs are no longer available with CICS TS:

- ▶ Security-related: DFHACEE, DFHXSE, and DFHXSP are replaced with RACROUTE calls to the security manager.
- ▶ Transaction restart: DFHRTY is replaced with DFHREST.

The following new URMs are available:

- ▶ DFHPGADX: Autoinstall program for programs
- ▶ DFHZATDY: Autoinstall program for APPC connections

The following modifications affect URMs:

- ▶ Data conversion program

Changes were made to the parameters that are passed to the user-replaceable data conversion program (DFHUCNV). Offsets in the DSECT changed and pointers to halfword length fields are now pointers to fullword length fields.

To macro DFHCNV, a new operand SYSDEF was added to the TYPE=INITIAL and TYPE=ENTRY macro parameters CLINTCP and SRVERCP. These macros define the user-replaceable data conversion table DFHCNV.

You must recompile program DFHUCNV and table DFHCNV.

- ▶ Dynamic transaction routing program

The communications area contains fields that are mapped by the DSECT DFHDYPDS. For more information about the DFHDYPDS DSECT, see *CICS Customization Guide*, SC33-1652. Field DYRVER contains the version number. The new version number of the dynamic routing program interface for CICS TS for z/VSE 2.1 is 4 (it was 3 for CICS TS for VSE/ESA).

7.5.1 z/VSE-supplied URMs

VSE supplies several URMs that were updated to conform to the guidelines for URMs in CICS TS. If you modified any of these programs, you must review your changes and integrate them into the new VSE versions of these programs.

Table 7-2 lists the VSE ICCF members and the CICS URM functions they perform. You can find the source code for these sample programs in ICCF library 59.

Table 7-2 VSE supplied CICS URMs

ICCF member	Function
IESZNEP	CICS Node Error Program (DFHZNEP) replacement. Command level version
IESZNEPS IESZNEPX	Additions to an existing Node Error Program. Command level version. (See Note 1.)
DFHPEP	CICS Program Error Program (DFHPEP) replacement. Command level and AMODE(31) version. (See Note 1 and Note 2.)
IESZATDX	CICS autoinstall program (DFHZATDX) replacement. Uses URM information that is passed in the COMMAREA.

Notes:

1. The DFHSNEP macro generates the AMODE(31) setting for each DSECT in the node error program. The VSE node error program logic issues a program control link to IESCLEAN to clean up Interactive Interface (IUI) control information after terminal errors.
2. The CICS EXEC commands are pre-translated in these VSE members.

7.6 System programmer interfaces

In addition to the interfaces described thus far, system programmers used other interfaces in CICS/VSE to access CICS information and perform system-related functions.

Although the interfaces that are described in this section are not all of the interfaces that the system programmers used, the more common interfaces are described here.

7.6.1 System programming macros

With the removal of CICS/VSE macro-level support, many of the following system programmer macro interfaces were removed:

- ▶ **DFHTC CTYPE**

This macro provided access to the terminal control table control blocks. These functions can be replaced with the **EXEC CICS INQUIRE** and **SET TERMINAL** commands.

- ▶ **DFHOC**

This macro was used to open and close files, extrapartition transient data queues, and dump data sets. It can be replaced with the **EXEC CICS SET FILE** commands.

- ▶ **DFHWTO**

This macro performed write-to-console operator functions. Some applications also used the COBOL and PL/I **DISPLAY** verbs to write to the console. These **DISPLAY** verbs were unsupported in a CICS environment. These methods can be replaced by using the **EXEC CICS WRITE OPERATOR** command.

- ▶ **DFHSEC**

This macro was used to start CICS security functions. Another technique that is commonly used to front-end CICS sign-on security was to link to the CICS sign on program (DFHSNP) and pass a TIOA formatted with the CSSN transaction and the user sign-on information. These < to what does “these” refer?>> do not work with CICS TS. The **EXEC CICS SIGNON** and **EXEC CICS SIGNOFF** commands can be used to replace these functions.

If you use any other system macro interfaces, see the system programming interface (SPI) that is described in *CICS Transaction Server for VSE/ESA System Programming Reference* for commands that can replace these functions.

7.6.2 Programmable interface to CEMT

You can still use this interface in CICS TS to start CEMT functions from an application program; however, we recommend that you use the **EXEC CICS INQUIRE|SET SPI** commands.

Several CEMT commands are available. Many of the commands are changed with new fields or fields that were removed. Some commands are obsolete.

For example, the **CEMT INQUIRE QUEUE** command on CICS/VSE V2R3 is now **CEMT INQUIRE TDQUEUE** on CICS TS. You might also get different results if you use abbreviations for some of the commands.

The supported CEMT functions in CICS TS are shown in Figure 7-1. If you use this interface, you might have to change your program because of these differences. For more information about the changes to the CEMT commands, see the *CICS Transaction Server for VSE/ESA Migration Guide*, GC33-1646.

INQ		
STATUS: ENTER ONE OF THE FOLLOWING OR HIT ENTER FOR DEFAULT		
AUTInstmodel	IRc	TRAnsaction
AUTOinstall	Journalnum	TRDumpcode
AUXtrace	MODename	TSqueue
Connection	MONitor	Vtam
DEletshipped	Netname	
DOctemplate	Partner	
DSAs	PROFile	
DSName	PROGram	
DUmpds	STatistics	
Exci	SYDumpcode	
FEConnection	SYStem	
FENode	TAsk	
FEPOol	TCLass	
FEPropset	TCPIP	
FETarget	TCPIPService	
File	TDqueue	
INttrace	TERminal	
		SYSID=CIC1
APPLID=DBDCCICS		
PF 1 HELP	3 END	9 MSG

Figure 7-1 CICS TS CEMT INQUIRE display

CEMT displays that were started from the VSE console also changed from CICS/VSE V2R3 and CICS TS. Figure 7-2 shows the console response from some typical CEMT commands issued from the console to a CICS TS partition.

```

161 cemt inq task
F2-0161
F2 0163
      Tas(0000026) Tra(CXPB)          Sus Tas Pri( 001 )
      Sta(S ) Use(DBDCCICS) Rec(X'D1459D37400F3B58') Hty(OPEN_ANY)
      Tas(0000028) Tra(ICVS)          Sus Tas Pri( 001 )
      Sta(S ) Use(CICSUSER) Rec(X'D1459D374014B658') Hty(USERWAIT)
      Tas(0000029) Tra(IESO)          Sus Tas Pri( 020 )
      Sta(S ) Use(CICSUSER) Rec(X'D1459D37C00BF852') Hty(EKCWAIT )
      Tas(0000052) Tra(CONL)          Sus Tas Pri( 255 )
      Sta(SD) Use(WACK ) Rec(X'D1459D56C2E52C51') Hty(USERWAIT)
      Tas(0001006) Tra(IECA)          Sus Tas Pri( 020 )
F2 0163 Sta(SD) Use(WACK ) Rec(X'D14D0406AB56D452') Hty(EKCWAIT )
      Tas(0001017) Tra(IECM)          Sus Tas Pri( 020 )
      Sta(SD) Use(WACK ) Rec(X'D14D04153A0C2952') Hty(EKCWAIT )
      Tas(0001018) Tra(CEMT) Fac(C001) Run Ter Pri( 255 )
      Sta(TO) Use(CNSL ) Rec(X'D14D04153A13BE52')
      RESPONSE: NORMAL TIME: 10.16.18 DATE: 09.05.16
      SYSID=CIC1 APPLID=DBDCCICS
161 cemt inq file(ies*)
F2-0161
F2 0163
      Fil(IESCNTL ) Vsa Ope Ena Rea Upd Add Bro Del
      Dsn( VSE.CONTROL.FILE )
      Fil(IESLDUM ) Vsa Clo Ena Rea Upd Add Bro Del
      Dsn( VSE.LDAP.USER.MAPPING )
      Fil(IESPRB ) Vsa Ope Ena Rea Upd Add
      Dsn( VSE.ONLINE.PROB.DET.FILE )
      Fil(IESROUT ) Vsa Ope Ena Rea Upd Add Bro Del
      Dsn( VSE.MESSAGE.ROUTING.FILE )
      Fil(IESTRFL ) Vsa Ope Ena Rea Upd Add Del
F2 0163 Dsn( VSE.TEXT.REPSTORY.FILE )
      RESPONSE: NORMAL TIME: 10.21.51 DATE: 09.05.16
      SYSID=CIC1 APPLID=DBDCCICS

```

Figure 7-2 CICS TS CEMT commands on console

You should include some operator training time in your migration to orient your computer operators to the new information that is displayed on the console by the CEMT commands.

7.6.3 System Programming Interface

The SPI commands provide a formal programming interface for writing programs to manage CICS system resources. By using these commands, you can perform many of the functions that were done by using direct access to control blocks.

CICS TS provides upward source compatibility and object compatibility for applications that use the SPI commands and which run correctly on CICS/VSE V2R3.

Several new SPI commands are available. Many of the existing commands have new options added and include some changed response values. For more information about the changes to the SPI interface that might affect applications that contain these commands, see *CICS Transaction Server for VSE/ESA Migration Guide*, GC33-1646.

If you compile any programs by using the SPI commands, you must specify the translator option SP; otherwise, the translator does not process the command.

If you use the CECI transaction to test SPI commands, the abbreviations for several of the commands that you can enter changed in CICS TS. For example, the **CECI INQUIRE TRAN(CEMT)** command that works on CICS/VSE V2R3 produces an error and defaults to the **CECI INQUIRE TRANC(CEMT)** command on CICS TS. This behavior is not a problem for the SPI commands in an application program because abbreviated commands are not allowed by the translator.

The supported CECI INQ functions in CICS TS are shown in Figure 7-3.

```
INQ
STATUS:  ENTER ONE OF THE FOLLOWING

AUTInstmodel Reqid      TSQName
AUTOinstall  STATistics TSQueue
Connection   STorage   Vtam
DEletshipped SYSDumpcode
DOctemplate  SYSTem
DSname       TAsk
DUmpds       TClass
Exitprogram  TCPIPService
File         TCPIP
Irc          TDqueue
Journalnum   TErmina1
MODename     TRACEDest
MONitor      TRACEFlag
Netname      TRACEType
Artner       TRANC1ass
PROFile      TRANDumpcode
PROGram      TRANSaction

PF 1 HELP 2 HEX 3 END 4 EIB 5 VAR 6 USER          9 MSG
```

Figure 7-3 CICS TS CECI INQUIRE display



Performance and tuning

This chapter reviews some of the performance and tuning aspects of IBM z/VSE 6.1 and the IBM CICS Transaction Server for z/VSE. This material is intended to give an introduction to the performance considerations and related facilities of the new release. For more information about performance and tuning, see *CICS Transaction Server for VSE/ESA Performance Guide*, SC33-1667.

This chapter also provides guidance in monitoring and tuning CICS performance, and describes the effect of many of the new parameters that are supplied with CICS TS.

In planning your migration, you must answer the following performance-related questions:

- ▶ How will the new releases affect your virtual storage requirements?
- ▶ Are your current system resources adequate?
- ▶ What performance and tuning data do you need?

This chapter reviews each of these areas, helps answer these questions, and includes the following topics:

- ▶ 8.1, “Virtual storage considerations” on page 194
- ▶ 8.2, “System resource requirements” on page 202
- ▶ 8.3, “Statistics and monitoring” on page 203

8.1 Virtual storage considerations

When migrating from CICS/VSE, several changes to VSE and CICS increase your overall system's virtual storage requirements in the following ways:

- ▶ More CICS storage areas are moved above the line into 31-bit GETVIS.
- ▶ CICS virtual storage requirements increased with the restructured CICS.
- ▶ New and changed VSE and CICS functions make more use of VSE data spaces.

You must change your VSE startup definitions and your CICS partition sizes to allow for these changes.

8.1.1 VSE startup parameters

You should review the following VSE IPL and JCL startup parameters:

- ▶ **ALLOC**

Increase the CICS partition allocation sizes. CICS F2-partition sizes for the environment B system are 50 MB. For the environment C system, this size is 256 MB.

- ▶ **PASIZE**

Increase the maximum private area size to allow for the larger CICS partitions. The PASIZE for environment B is 150 MB. For environment C, the size is 512 MB.

- ▶ **SYSDEF**

Increase the system limits for the total amount of data space that can be allocated. Consider the following points:

- The supplied SYSDEF statement in the ALLOC procedure specifies a maximum DSIZE of 40 MB for environment B, and 256 MB for environment C.
- The VSE BSM uses a 960 K data space.
- The CICS Data Management Facility and Shared Data Tables require data spaces, the sizes of which are user-specified.

- ▶ **VSIZ**

Increase the VSE system total virtual size to accommodate the increase in CICS partition sizes and more data space requirements.

8.1.2 Shared Virtual Area

The 31-bit program area and the number of System Directory List (SDL) entries in the IPL Shared Virtual Area (SVA) statement for Environment B are larger. The 31-bit PSIZE value is 8 MB for environments B and C. The SDL value has 700 entries. The Interactive Interface (IUI) storage display that is shown in Figure 8-1 previews an overview of the SVA storage layout.

IESADMDSV1						SHARED VIRTUAL AREA LAYOUT		

'0AFFFFFF'X		SYSTEM GETVIS(31)		USED:	(HWM: 2864K)	2172K		22M
'0A474000'X				FREE:		9652K	12M	

		PROGRAM AREA(31)		USED:		7900K		
'09A00000'X				AVAILABLE:		2804K	10M	
=====								
'003DFFFF'X		V-POOL				64K		
'003B5000'X		SYSTEM LABEL AREA (SLA)				108K	172K	

		SYSTEM GETVIS(24)		USED:	(HWM: 692K)	676K		3204K
'00212000'X				FREE:		1000K	1676K	

		PROGRAM AREA(24)		USED:		977K		
'000CF000'X				AVAILABLE:		315K		
							1356K	
		SYSTEM DIRECTORY LIST (SDL)				64K		
'000BF000'X								

PF1=HELP 2=REFRESH 3=END 4=RETURN 6=PARTITION								

Figure 8-1 z/VSE SVA storage layout

You can load most of the SVA-eligible CICS modules and programs in the extended SVA in 31-bit storage. Loading programs into 31-bit storage can provide you storage relief for the 24-bit SVA if you currently load CICS modules in the SVA.

The number and size of CICS SVA-eligible modules increased. You might need to increase the 31-bit SVA size to fully use the SVA for CICS modules.

8.1.3 CICS partition layout

The Interactive User Interface (IUI) storage display for the CICSICCF partition from our installed z/VSE V6R1 system is shown in Figure 8-2.

IESADMDSPL		STATIC PARTITION LAYOUT	
PARTITION: F2		JOB NAME: CICSICCF PHASE: DFHSIP	
-----		-----	
'104FFFFF'X		GETVIS ANY	
		USED: 224M	1
		FREE: 32M	
- -(16M)- - - - -		HIGH WATER MARK: 227M	
A GETVIS BELOW		LARGEST FREE BLOCK: 32M	
USED: 8856K		PFIX (31 BIT) USED: 0B	
FREE: 2404K		PFIX (31 BIT) LIM.: 0B	256M
HIGH WATER MARK: 11M			
'00501000'X		LARGEST FREE BLOCK: 2384K	V
-----		-----	
PROGRAM AREA (EXEC SIZE)		LOADED PHASE: 582B	
'00500000'X	PFIX (24) USED: 32K LIM.: 144K	AVAILABLE: 3514B	4096B
-----		-----	
		PARTITION: 256M	
PF1=HELP		2=REFRESH	3=END
		4=RETURN	6=SVA

Figure 8-2 z/VSE 6.1 CICSICCF partition layout

For more information about the content of these storage areas, see “CICS TS partition content” on page 197.

Consider the following points:

- ▶ More storage is used in 31-bit GETVIS for CICS storage areas that are above the line compared to previous CICS releases.
- ▶ All of the CICS storage areas that are below the line are in 24-bit GETVIS except for the DFHSIP phase.
- ▶ The program area size must be large enough only to load the DFHSIP phase. Change your size parameter on the EXEC statement in your CICS job streams to SIZE=DFHSIP.
- ▶ The CICS partition sizes increased.

CICS TS partition content

The organization of the CICS partition in CICS TS is shown Figure 8-3.

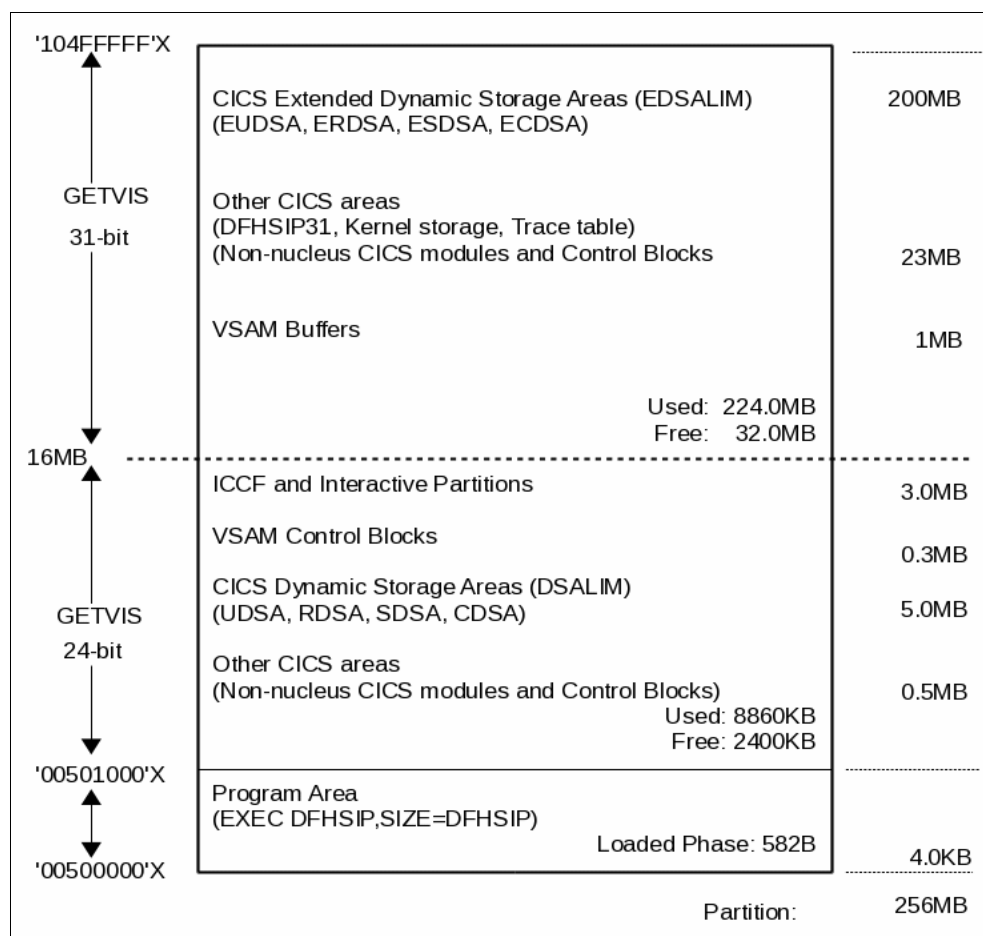


Figure 8-3 z/VSE 6.1 CICS ICCF storage layout

CICS allocates dynamic storage areas (DSAs) below and above the 16MB line. Consider the following points:

- Eight DSAs are available in CICS Transaction Server for VSE/ESA.
- Four DSAs are in extended storage above the 16 MB line.

The EDSALIM parameter in your CICS system initialization parameters is the maximum amount of 31-bit GETVIS that can be allocated for extended DSAs.

Note: Extended DSA (EDSA) storage is required. You must specify at least 10 MB or CICS does not initialize.

- Four DSAs are in 24-bit GETVIS below the line.

The DSALIM parameter in your CICS system initialization parameters specifies the maximum amount of 24-bit GETVIS that can be allocated for DSAs below the 16 MB line.

Note: DSA storage is required. The minimum specification is 2 MB.

- In addition to the DSA and EDSA storage, CICS uses 24-bit and 31-bit GETVIS storage for other CICS components.
- ICCF and interactive partition storage allocations are the same as in previous releases.
- The program area size is large enough only to load the DFHSIP phase. Change your size parameter on the EXEC statement in your CICS job streams to SIZE=DFHSIP. VSE rounds the SIZE allocation to a 4 K multiple.

You must increase your CICS partition allocations to provide sufficient 31-bit GETVIS storage for CICS Transaction Server (CICS TS).

CICS dynamic storage areas

Table 8-1 lists the eight CICS dynamic storage areas and the type of storage they contain. The extended DSAs in 31-bit GETVIS are prefixed with an uppercase letter E.

Table 8-1 CICS DSA Descriptions

DSA name	DSA type	Content
UDSA EUDSA	User DSAs	User-key task-lifetime storage.
SDSA ESDSA	Shared DSAs	Non-reentrant user-key programs and SHARED GETMAIN storage.
RDSA ERDSA	Read-only DSAs	Key-0 storage for all reentrant programs and tables.
CDSA ECDSA	CICS DSAs	Storage for all non-reentrant CICS key programs, CICS key task storage, and all CICS control blocks.

You can specify a fixed size for each of these areas by using system initialization parameters or, CICS can dynamically acquire storage for these areas as required.

You can change only the maximum limits while CICS is running by using one of the following CEMT commands:

- CEMT SET DSA
- CEMT SET SYSTEM

To view the DSA allocation status, you can use CEMT INQ DSA or function 3-6-4 (Display CICS TS storage) of Interactive Interface, as shown in Figure 8-4 on page 199. An example of the CEMT INQ SYSTEM command is shown in Figure 8-5 on page 201.

IESADMD CST	DISPLAY CICS TS STORAGE				Time: 13:22:09
Applid: DBDCCICS	Sysid: CIC1	Jobname: CICSICCF CICS TS Level: 210			
Storage Protection	ACTIVE	Reentrant Programs PROTECT			
		CICS Trace Table size.. 4096			
Extended DSA:	(All sizes in kbyte)				LIMIT 204800
	ECDSA	EUDSA	ESDSA	ERDSA	Totals
Current DSA Size	14336	1024	1024	8192	24576
Current DSA used	9928	128	128	7320	17504
*Peak DSA used	9952	128	128	7320	
Peak DSA Size	14336	1024	1024	8192	24576
Largest free area/Free Storage	0.23	1.00	1.00	0.98	
Times short-on-storage (SOS) ..	0	0	0	0	0
DSA:					LIMIT 5120
	CDSA	UDSA	SDSA	RDSA	Totals
Current DSA Size	512	256	256	512	1536
Current DSA used	404	16	200	388	1008
*Peak DSA used	424	40	216	388	
Peak DSA Size	512	256	256	512	1536
Largest free area/Free Storage.	0.96	0.98	0.93	0.90	
Times short-on-storage (SOS)...	0	0	0	0	0
PF1=HELP	2=REFRESH	3=END	4=RETURN		

Figure 8-4 Display CICS TS storage display (3-6-4)

If you increase the DSALIM parameter, you must have enough free 24-bit GETVIS in the CICS partition to satisfy the extra storage that is allocated for the DSA. If you increase the EDSALIM parameter, you must have enough free 31-bit GETVIS in the CICS partition to satisfy the extra storage that is allocated for the EDSA.

We recommend that you allow CICS to dynamically allocate the individual DSAs, even though you can fix the sizes with system initialization parameters.

The DSAs feature the following characteristics:

- ▶ Extended DSAs and the EDSALIM are allocated in multiples of 1 MB.
- ▶ DSAs that are below the line and the DSALIM are allocated in multiples of 256 K.
- ▶ CICS manages storage within each DSA by using a 4 K page size. The SIT PGSIZE parameter is no longer supported.
- ▶ SVA-eligible programs that are not in the SVA are loaded in the RDSA for 24-bit programs and in the ERDSA for 31-bit programs. Loading programs in (E)RDSA provides read-only access protection that is the same as loading programs in the SVA, provided the system initialization parameter RENTPGM=PROTECT is specified.
- ▶ New Resource Definition Online (RDO) parameters on the program and transaction definitions control which DSA is used for program-related and task-related storage.

On transaction definitions, the following parameters affect where storage is allocated:

- ▶ TASKDATAKEY(USER)

This parameter specifies that all task-lifetime storage is obtained from user-key storage in the user DSA below the line (UDSA) or the extended user DSA above the line (EUDSA). Task-lifetime storage includes the transaction work (TWA), the EXEC interface block (EIB), copies of working storage, explicit GETMAINS, and implicit GETMAINS, which result from the SET option on other commands.

► TASKDATAKEY(CICS)

This parameter specifies that all task-lifetime storage is obtained from CICS-key storage in the CICS DSA below the line (CDSA) or the extended CICS DSA above the line (ECDSA).

► TASKDATALOC(BELOW|ANY)

This parameter controls from which DSA the task-lifetime storage is obtained; that is, the DSA below the line (BELOW) or the extended DSA above the line (ANY). On program definitions, the following parameters affect where storage is allocated:

– EXECKEY(USER)

Non-reentrant programs defined with this option are loaded in the shared DSAs. RMODE(24) programs are loaded in the SDSA below the line and RMODE(31) programs are loaded in the ESDSA above the line.

– EXECKEY(CICS)

Non-reentrant programs defined with this option are loaded in the CICS DSAs. RMODE(24) programs are loaded in the CDSA below the line and RMODE(31) programs are loaded in the ECDSA above the line. Reentrant programs (programs compiled with the RENT option and link-edited SVA-eligible) are loaded in the read only DSAs regardless of the EXECKEY value. RMODE(24) programs are loaded in the RDSA below the line and RMODE(31) programs are loaded in the ERDSA above the line. The system initialization parameter RENTPGM=PROTECT must be specified for the read-only DSAs to be allocated from key-0 read-only storage.

– DATALOCATION(BELOW|ANY)

This parameter affects the addresses passed to the program in the address field of commands with the SET option. Data is copied below the line if necessary when DATALOCATION(BELOW) is specified.

Transaction definitions that are migrated to CICS TS include the following default settings:

- TASKDATAKEY(USER)
- TASKDATALOC(BELOW)

Program definitions that are migrated to CICS TS include the following default settings:

- EXECKEY(USER)
- DATALOCATION(BELOW)

An example of the **CEMT INQ SYSTEM** command is shown in Figure 8-5.

```
INQ SYSTEM
STATUS: RESULTS - OVERTYPE TO MODIFY
Aging( 05000 )           Oslevel(060100)
Akp(          )          Progautoctlg( Ctlgall )
Cdsasize(00524288)       Progautoexit( DFHPGADX )
Cicstslevel(020100)      Progautoinst( Autoactive )
Cmdprotect(Cmdprot)      Rdsasize(00524288)
Dfltuser(CICSUSER)       Reentprotect(Reentprot)
Dsalimit( 05242880 )     Release(0420)
Dtrprogram( DFHDYP )     Runaway( 0020000 )
Dumping( Sysdump )       Scandelay( 0100 )
Ecdsasize(0014680064)    Sdsasize(00262144)
Edsalimit( 0209715200 )  Sosstatus(Notsos)
Erdsasize(0008388608)    Storeprotect(Active)
Esdsasize(0001048576)    Time( 0001000 )
Eudsasize(0001048576)    Udsasize(00262144)
Maxtasks( 050 )
Mrobatch( 001 )
Oprel(93)
Opsys(E)

                                SYSID=CIC1 APPLID=DBDCCICS
RESPONSE: NORMAL                                TIME: 13.36.19 DATE: 09.07.16
PF 1 HELP                                3 END                                7 SBH 8 SFH 9 MSG 10 SB 11 SF
```

Figure 8-5 CEMT INQ SYS display

8.1.4 Storage requirements for maximum task specification

The role of the maximum task specification (MXT) parameter changed significantly in CICS TS. In CICS/VSE, you used the MXT parameter to limit the total number of system and user tasks that CICS allowed at any one time. In CICS TS, you use the MXT parameter to limit the total number of user tasks only. CICS TS provides for up to 15 system tasks.

The setting of MXT can significantly affect the amount of DSA and EDSA storage that is required for your CICS partition. Storage to support the maximum number of user tasks is pre-allocated from the DSAs.

The following formulas are used for calculating the DSA storage requirements:

- ▶ DSA MXT storage = $20K + (MXT * 2K) + (MXT/10 * 4K)$
- ▶ EDSA MXT storage = $120K + (MXT * 12K) + (MXT/10 * 4K)$

For example, the supplied DFHSIT skeletons in ICCF library 59 for CICSICCF and a second CICS partition specify MXT=50. The following storage is required:

- ▶ DSA MXT storage = $20K + (50 * 2K) + (50/10 * 4K) = 140K$
- ▶ EDSA MXT storage = $120K + (50 * 12K) + (50/10 * 4K) = 740K$

If you use MXT=999 as coded in the skeleton DFHSIT tables that are supplied with older VSE/ESA releases, the following storage is required:

- ▶ DSA MXT storage = $20K + (999 * 2K) + (999/10 * 4K) = 2414K$
- ▶ EDSA MXT storage = $120K + (999 * 12K) + (999/10 * 4K) = 12504K$

Note: Because this storage requirement is significantly larger, avoid setting MXT to a high value.

You can use your CICS shutdown statistics from your earlier CICS releases to help you set MXT. As an initial starting point, set MXT to the value in the “Peak Number of Tasks” statistic. If the “Number of Times at Max Task” statistic is high, you might need to increase MXT slightly.

The MXT value can be changed dynamically by using the **CEMT INQUIRE SYSTEM** command, as shown in Figure 8-5 on page 201. If you increase the MXT value, you must have sufficient storage available in the DSAs below and above the 16 MB line. CICS issues the error message “Short on Storage” or “Short on Extended Storage” if it cannot allocate the extra storage required for the MXT increase.

8.2 System resource requirements

Measure your current system processor and storage utilization to determine whether you need more capacity to support the migration to z/VSE V6R1.

If you have real storage constraint problems or high processor utilization, you should add real memory and processor capacity as needed.

8.2.1 Real storage considerations

The increase in virtual storage requirements might cause paging problems if you are using real storage-constrained today. You can use the **VSE SIR** command from the console to determine whether your VSE system is paging. You can also use the z/VSE IUI system activity dialog to identify paging rates and potential problems. When enough processor storage is available we recommend the no-page-dataset option (NOPDS Supervisor parameter that is set at IPL).

The use of this option avoids the overhead for VSE paging operations at all.

The real storage usage depends on the working set of your VSE system. You can have larger virtual storage allocations without affecting real storage if the working set does not increase dramatically.

We ran multiple CICS TS partitions in our VSE system with a VSIZE of 2048 MB. The system ran under VM. We did not observe any paging problems. Some paging occurred during CICS TS startups, as expected.

8.3 Statistics and monitoring

CICS TS changed the way it collects statistics and monitoring data. CICS Statistics and Monitoring domains issue requests to a new facility, the Data Management Facility (DMF), to record data to VSAM data sets.

You can also implement a CICS-supplied sample program to capture statistics. The sample program writes statistics to temporary storage or the POWER LST queue and does not require DMF.

8.3.1 Setting up the DMF

Complete the following steps to implement the DMF:

1. Define the DMF data sets.
2. Initialize the DMF data sets.
3. Generate the DMF startup table.
4. Create a DMF startup job.
5. Learn how to operate DMF.
6. Create job streams to process the DMF data.

For more information about the setting up and operating DMF, see *CICS Transaction Server for VSE/ESA Operations and Utilities Guide*, SC33-1654.

Defining DMF data sets

DMF data sets DFHDMFA and DFHDMFB are defined during the installation of z/VSE V6R1. Although these data sets are adequate for testing DMF, you should delete and redefine them with new allocations to meet your data collection needs.

The job stream that we created to redefine the DMF data sets is shown in Figure 8-6 on page 204.

```

* $$ JOB JNM=DMFDEFS,CLASS=0,DISP=D
* $$ LST CLASS=A,DISP=D
// JOB DMFDEFS          DEFINE CICS/TS DMF DATASETS
// EXEC IDCAMS,SIZE=AUTO
DELETE (CICSTS.SYSTEM.DFHDMFA) CL NOERASE PURGE -
  CATALOG(VSESP.USER.CATALOG)
DELETE (CICSTS.SYSTEM.DFHDMFB) CL NOERASE PURGE -
  CATALOG(VSESP.USER.CATALOG)
DEFINE CLUSTER (NAME(CICSTS.SYSTEM.DFHDMFA) -
  NONINDEXED -
  CYLINDERS(5) -
  REUSE -
  RECORDSIZE (125 32767) -
  SPANNED -
  VOLUMES (SYSWK1) -
  NOCOMPRESSED -
  SHAREOPTIONS (2) -
  TO (99366 )) -
  DATA (NAME (CICSTS.SYSTEM.DFHDMFA.@D@) -
  CONTROLINTERVALSIZE (4096)) -
  CATALOG (VSESP.USER.CATALOG)
DEFINE CLUSTER (NAME(CICSTS.SYSTEM.DFHDMFB) -
  NONINDEXED -
  CYLINDERS(5) -
  REUSE -
  RECORDSIZE (125 32767) -
  SPANNED -
  VOLUMES (DOSRES) -
  NOCOMPRESSED -
  SHAREOPTIONS (2) -
  TO (99366 )) -
  DATA (NAME (CICSTS.SYSTEM.DFHDMFB.@D@) -
  CONTROLINTERVALSIZE (4096)) -
  CATALOG (VSESP.USER.CATALOG)
/*
/&
* $$ EOJ

```

Figure 8-6 DMF VSAM data set definition

Complete the following steps to adapt the DMF data set definition to your system:

1. Define the DMF data sets for the VSE system (you can define up to 36 such data sets).

The following data sets are created during VSE installation:

- CICS.DBDCCICS.DFHDMFA
- CICS.DBDCCICS.DFHDMFB

We changed the names to emphasize that these data sets are system data sets that are used for all CICS partitions that are running in the VSE system, not just the DBDCCICS CICS.

2. Change the allocation to meet your system requirements.

The VSE installation creates DMF data sets with allocations of six tracks primary and two tracks secondary for 3390 DASD. These tracks are too small for normal use. We changed this setting to five cylinders with no secondary allocation for our testing. A likely better starting point is 10 cylinders.

CICS automatically switches DMF data sets when the currently active data set becomes full. If secondary allocation is used, switching between data sets does not occur until the file extends to the maximum number of VSAM secondary allocations.

These data sets are defined with the REUSE attribute, which results in 16 VSAM extents per volume for each DMF data set that is defined with secondary allocation.

3. Change the volumes to match your installation.

The VSE installation definitions use secondary allocation across multiple volumes. Primary allocation requires one volume only.

Initializing the DMF data sets

The VSE-supplied DMF data sets are initialized during the VSE installation process. You must reinitialize these data sets if you redefine the data sets to increase the allocations or to clear the data that was collected.

The job stream that we created to initialize the DMF data sets is shown in Figure 8-7.

```
* $$ JOB JNM=DMFINIT,DISP=D,CLASS=0
// JOB DMFINIT FORMAT CICS DMF DATASETS
// DLBL DFHDMFA,'CICSTS.SYSTEM.DFHDMFA',,VSAM,CAT=VSESPUC .1.
// DLBL DFHDMFB,'CICSTS.SYSTEM.DFHDMFB',,VSAM,CAT=VSESPUC .1.
// LIBDEF *,SEARCH=PRD1.BASE
// EXEC DFHDFOU,SIZE=DFHDFOU .2.
INDD(DFHDMFA, OPTIONS (CLEAR)) .3.
INDD(DFHDMFB, OPTIONS (CLEAR)) .3.
/*
/&
* $$ EOJ
```

Figure 8-7 DMF initialization job stream

The numbers that are highlighted in Figure 8-7 correspond to the following information:

1. Add these file labels to standard labels to replace the VSE-supplied definitions if you change the data set names.
2. DFHDFOU is the CICS-supplied DMF dump utility program.
3. One control statement is needed for each DMF data set that you want to initialize.

DMF startup table

The DMF startup table contains the initialization parameters that are used by the DMF startup job when it is activated. This table is used for DMF startup only and is not referenced by any CICS partitions in your system.

VSE supplies a sample DMF startup table in skeleton member DFHDMFSP in ICCF library 59. CICS supplies a default DMF startup table DFHDMFSU. The source is in library PRD1.BASE in member DFHDMFSU.A.

A modified copy of the VSE-supplied skeleton that we used for our implementation is shown in Figure 8-8.

```

* $$ JOB JNM=DFHDMFSP,CLASS=A,DISP=D,NTFY=YES
* $$ LST CLASS=Q,DISP=H
// JOB DFHDMFSP ASSEMBLE
// LIBDEF *,CATALOG=PRD2.CONFIG
* IN CASE GENERATION FEATURE IS INSTALLED ACTIVATE THE FIRST LIBDEF
* // LIBDEF SOURCE,SEARCH=(PRD2.GEN1,PRD1.BASE,PRD1.MACLIB)
// LIBDEF SOURCE,SEARCH=(PRD1.BASE,PRD1.MACLIB)
// OPTION CATAL,LIST
// EXEC ASMA90,SIZE=(ASMA90,64K),PARM='EXIT(LIBEXIT(EDECKXIT)),SIZE(MAXC
-200K,ABOVE)'

*****
*
* 5686-066 (C) COPYRIGHT IBM CORP. 1996
*
*****

      TITLE 'DFHDMFSP -- SUPPLIED BY VSE/ESA'
      PUNCH ' CATALOG DFHDMFSP.OBJ REP=YES'
      DFHDMFM TABLE,
          CATALOG=VSESP.USER.CATALOG,  USE VSESPUC
          FILELIST=(CICSTS.SYSTEM.DFHDMFA,CICSTS.SYSTEM.DFHDMFB),
          INTERVAL=3000,      30 MINUTES 0 SECONDS
          LISTDSN=YES,        SHOW DATASETS WHEN DMF STARTS
          SID=VSE,            SYSTEM IDENTIFIER
          SIZE=4,             USE A 4M DATA SPACE
          STATUS=ACTIVE,      DMF IS ACTIVE AT START
          SUFFIX=SP,          THIS TABLE IS CALLED DFHDMFSP
          TRACE=NO,           NO TRACE ACTIVITY
          TRTABSZ=1024,       TRACE TABLE SIZE IS 1M
          TYPE=0:255,         RECORD ALL DMF DATA RECORD TYPES
          USAGE=50            REDUCE SPACE WHEN 50% FULL

      END

/*
// IF $MRC GT 4 THEN
// GOTO NOLINK
// EXEC LNKEDT,PARM='MSHP'
/. NOLINK
/*
/&
* $$ EOJ

```

Figure 8-8 DMF startup table

The numbers that are highlighted in Figure 8-8 correspond to the following information:

1. Sets the name of the VSE catalog in which you defined the DMF data sets.

Theses names are the names of the data sets that you defined for DMF to use. If you have multiple DMF data sets, an easier way to code the file name list is shown in the following example:

```
GENFILES=4,GENPREFIX=(CICSTS.SYSTEM.DFHDMF)
```

This code generates four file names that include the suffix Z, Y, X, and W.

2. Defines how often you want DMF to dump the data in its data space to the currently active DMF data set.
3. Specifies the DMF data space size (include it in your data space requirements).
4. Defines the size of DMF's internal trace table.
5. Specifies how full the data space must be before DMF writes records to the currently active data set.

Creating a DMF startup job

DMF runs in its own partition that is separate from your CICS partitions. You can have only one DMF job that is running in your VSE system at a time. You must start DMF before CICS partitions can start recording statistics and monitoring data.

VSE supplies a sample DMF startup job stream in skeleton member SKDMFST in ICCF library 59. The supplied skeleton job contains a job step that reinitializes the DMF data sets before starting DMF. You should move this job step to a separate job; otherwise, data is cleared if the VSE system were IPLed before you processed the DMF data sets.

A copy of the VSE-supplied skeleton that we tailored for our DMF startup job is shown in Figure 8-9.

```
* $$ JOB JNM=DMFSTART,CLASS=Z,DISP=D,NTFY=YES .1.
* $$ LST CLASS=Q,DISP=H
// JOB DMFSTART DMF STATISTICS AND MONITORING SERVER
* -----*
*   THIS JOB STARTS THE DMF SERVER PARTITION FOR CICS/TS   *
* -----*
* FOR COMMUNICATION USE 'MSG XX' (PARTITION ID) AND .2.
* 'XX SETDMF .' COMMANDS.
* TERMINATING:
* 'XX SETDMF SHUTDOWN'
* -----*
// EXEC DFHDFSIP,SIZE=DFHDFSIP,OS390 .3.
SUFFIX=SP
/*
/&
* $$ E0J
```

Figure 8-9 DMF startup job stream

The numbers that are highlighted in Figure 8-9 correspond to the following information:

1. A dedicated partition must be allocated for DMF because it must be active before statistics and monitoring data can be recorded. A 5 MB partition is sufficient to run DMF. (We used one of the installations that are supplied class Z dynamic partitions.)
2. Instructions for communicating with DMF. For more information, see section “Operating DMF” on page 208.
3. Runs the DMF initialization program. IBM OS/390® is required on the EXEC statement to emulate IBM OS/390 services. The DSPACE parameter is not required because DMF uses SCOPE=COMMON data space. The SUFFIX parameter is the suffix of your DMF startup table.

Operating DMF

Consider the following operational aspects when DMF is used:

- ▶ Starting and stopping DMF
- ▶ Controlling DMF

DMF is started by releasing the job as shown in Figure 8-9 on page 207. Console messages from the startup of DMF are shown in Figure 8-10.

```
Z1 0045 DMF CONSOLE REPORT
      DFHDF0016 Data management facility has loaded program DFHDMFSP, which
      has entry point X'80636020'.
Z1 0045 DMF CONSOLE REPORT
      DFHDF0007 This startup is using suffix SP.      .1.
Z1 0045 DMF CONSOLE REPORT
      DFHDF0028 Data management facility trace status is set to NOTRACE.

Z1 0045 DMF CONSOLE REPORT
      DFHDF0025 Data management facility has created dataspace named
      DFHDF000, which is 4M bytes in size.      .2.
Z1 0045 DMF CONSOLE REPORT
      DFHDF0016 Data management facility has loaded program DFHDFFM , which
      has entry point X'80644750'.
Z1 0177 DFHDF3001 Data set status report.      .3.
Z1 0177 DFHDF3002 Data Set Name                                Status Avail(%)

Z1 0177 DFHDF3003 CICSTS.SYSTEM.DFHDMFA                        INIT      100

Z1 0177 DFHDF3003 CICSTS.SYSTEM.DFHDMFB                        INIT      100

Z1 0177 DFHDF3004 Data set status report is complete.
Z1 0177 DMF CONSOLE REPORT
      DFHDF3005 Data Management Facility dataset CICSTS.SYSTEM.DFHDMFA
      is open.      .4.
Z1 0045 DMF CONSOLE REPORT
      DFHDF0001 Data management facility is started.
```

Figure 8-10 DMF startup console messages

The numbers that are highlighted in Figure 8-10 correspond to the following information:

1. The suffix of the DMF startup table that us specified in the startup job.
2. The name and size of the data space that us allocated by DMF.
3. Status of the DMF data sets that are defined in the DMF startup table. The status indicates whether a data set is initialized (INIT), partially full (PARTIAL), or full (FULL).
4. The currently active DMF data set.

When the active data set fills up, DMF switches data sets automatically, a process that is similar to CICS journal switching. The console messages that are issued when the data sets are switched is shown in Example 8-1.

Example 8-1 Console messages

```
Z1 0085 DMF CONSOLE REPORT
DFHDF3006 Data Management Facility dataset CICSTS.SYSTEM.DFHDMFA is closed.
Z1 0085 DMF CONSOLE REPORT
DFHDF3005 Data Management Facility dataset CICSTS.SYSTEM.DFHDMFB is open.
```

If all the data sets are full, DMF issues the messages on the console that are shown in Example 8-2.

Example 8-2 DMF issued messages

```
Z1 0085 DFHDF3002 Data Set Name Status Avail(%)
Z1 0085 DFHDF3003 CICSTS.SYSTEM.DFHDMFA FULL 0
Z1 0085 DFHDF3003 CICSTS.SYSTEM.DFHDMFB FULL 0
Z1 0085 DFHDF3004 Data set status report is complete.
Z1 0085 DMF CONSOLE REPORT
DFHDF3007 Data Management Facility cannot find a dataset to open.
Reply 'GO' to cause DMF to continue operating but without a data set (data will
still be recorded in the data space), or 'RETRY' to cause DMF to retry its attempt
to open a data set
Z1-0085 DFHDF0000 Please enter reply
```

You need to run the DMF batch utility to dump and reinitialize the DMF data sets so that DMF can continue recording to the data sets. DMF continues to collect data in its dataspace until it fills; subsequent data recording are lost.

To communicate with the DMF partition, enter the **MSG xx** command from the VSE console, where **xx** is the partition in which DMF is running. DMF responds with the message that is shown Example 8-3.

Example 8-3 DMF operator communication

```
Z1 0047 DMF CONSOLE REPORT
DFHDF1001 Data management facility is ready for communication. At the prompt, you
may enter a DMF command, or request assistance by entering a '?'
Z1-0047 DFHDF0000 Please enter reply
```

Some of the DMF commands that you can enter are shown in Table 8-2.

Table 8-2 DMF operational commands

Console command	Function
?	Lists general operating instructions for DMF.
?DISPLAY	Lists information about the DISPLAY command.
?SETDMF	Lists information about the SETDMF command.
DISPLAY	Displays the current active data set.
DISPLAY O	Displays current settings for DMF.
SETDMF NOACTIVE	Suspends data recording.

Console command	Function
SETDMF ACTIVE	Resumes data recording.
SETDMF SWITCH	Closes the active data set and opens the next data set.
SETDMF FLUSH	Writes data in the data space to the active data set.
SETDMF SHUTDOWN	Terminates DMF after writing to the active data set and closing it.

You can also issue DMF commands from the console by using the following format:

```
MSG XX,DATA=SETDMF SHUTDOWN
```

8.3.2 Statistics

CICS TS changed the way it records and reports statistics data. In CICS/VSE, statistics were written to transient data and typically printed on SYSLST for subsequent review. In CICS TS, the CICS Statistics Domain issues requests to DMF to record statistics data in DMF data sets. You then must run a CICS batch utility job stream to select and format the statistics that you require.

As a temporary alternative before setting up DMF for statistics recording, you can implement a CICS-supplied sample program to record statistics. This program writes statistics information to the POWER/VSE LST queue or to CICS transient data. DMF is not required to produce this data.

When you are migrating from CICS/VSE, consider the following points regarding CICS resource definitions:

- ▶ CSSM and CSSN transient data queues are no longer used and can be removed from the Destination Control Table (DCT).
- ▶ The statistics data sets DFHSTM and DFHSTN are also obsolete.
- ▶ The CICS statistics transaction CSTT is replaced with CEMT commands.

Sample statistics program

The CICS sample statistics program produces statistics output that is similar to the shutdown statistics in older versions of CICS. This program demonstrates the use of new command level interfaces to collect statistics. The output from the sample program is written to the POWER LST queue by using the EXEC CICS SPOOL interface.

To install the sample statistics programs, complete the following steps:

1. Compile the sample statistics program mapset.

The source for the mapset is in member DFH0STM.A in VSE sublibrary PRD1.BASE. You must compile the mapset before compiling the sample program. The copybook from the mapset generation must be available in a sublibrary that is accessible by the sample program compile.

2. Translate and compile the sample statistics program.

The sample program source is in member DFH0STAT.C in PRD1.BASE. The program is written in COBOL. If no COBOL compiler is available, you can use member DFH0STAT in ICCF Library 59 to catalog the phase.

3. Add resource definitions to the CSD for the programs, the mapset, and the STAT transaction.

The RDO entries to support the sample statistics program are supplied in the CSD in RDO group DFH\$STAT. If you plan to invoke the sample statistics program from the shutdown PLT, you must ensure that the RDO program entries in this group specify CICS in the EXECKEY parameter. You must copy DFH\$STAT to your own RDO group to change the CICS-supplied definitions. CICS-supplied group entries cannot be changed.

4. Add support for the system spool interface to your CICS startup.

You must code SPOOL=YES in your system initialization parameters to support the EXEC SPOOL commands that are issued by the sample program.

The statistics output is in a POWER LST queue entry that is separate from the entry that contains the CICS startup JCL and CICS transient data messages. You can identify the statistics LST queue entry by the user ID name in the FROM field.

For example, in the LST queue display for CICS startup job CICSTS5 that is shown in Example 8-4, the user ID that submitted the statistics output is SYSCICS5. This name is the internal name of this CICS that is used by the spool interface and the default user ID that is used by the sample program.

Example 8-4 LST queue display for CICS startup job CICSTS5

```
F1 0001 1R46I LIST QUEUE P D C S PAGES CC FORM
F1 0001 1R46I CICSTS5 01052 3 D A 16 1 FROM=(SYSCICS5)
F1 0001 1R46I CICSTS5 01051 3 D A 23 1 TO=(RES5) FROM=(RES5)
```

STAT transaction

You can also invoke the sample statistics program by entering the STAT transaction from a terminal while CICS is running. The BMS map that is displayed and the options that you can change when you enter the STAT transaction are shown in Figure 8-11.

Sample Program - CICS Statistics Print		09/08/16 08:54:13
Type in destination fields if required. Press Enter to print		
Jobname . . . :	CICSICCF	
AppId . . . :	DBDCCICS	
Sysid . . . :	CIC1	
Node *	Type in a valid Node. * is default	
Userid *	Type in a valid Userid. * is default	
Class A	Type in a valid Class. A is default	
TS Queue Name	Type in TS Queue name, to send out- put to this TS queue instead.	
Abbreviated	Enter x for abbreviated TS report	
Statistics print successfully completed		
F3=Exit to CICS		

Figure 8-11 Sample STAT transaction display

Sample statistics program output

Figure B-1 on page 258 lists some of the output that is produced by the sample statistics program from one of our CICS partitions. A total of 16 pages of output were generated by the sample program.

In our test, we added an entry for the statistics program to the CICS shutdown PLT. You can add the sample program to your shutdown PLT to capture statistics when CICS is shut down. The VSE-supplied shutdown PLT skeletons in ICCF library 59 contain a commented-out entry for the sample statistics program.

The output of the sample statistics program is similar to the shutdown statistics from previous CICS versions. Some of the more familiar statistics are included:

- ▶ Task activity, including the total transactions, times at MXT, and peak tasks
- ▶ Storage usage
- ▶ Transaction and program usage
- ▶ File control and LSR pool activity
- ▶ Temporary storage and transient data usage

Several new statistics also are recorded to help tune and monitor CICS TS. New statistics are available for the temporary storage and transient data usage areas, and for the new parts of CICS TS, such as the new dynamic storage areas and domains. For more information about the statistics that CICS TS collects, see *CICS Transaction Server for VSE/ESA Performance Guide*.

Collecting statistics by using DMF

The following types of statistics can be collected by the CICS Statistics domain and written to DMF by using use a specific CICS command:

▶ Interval statistics

The CICS statistics domain collects statistics at intervals that you specify and write them to DMF. Interval statistics are recorded only if you specify the SIT option **STATRCD=ON** at CICS startup or enable recording by using the **CEMT SET STATISTICS ON** command. Statistics recording needs to be enabled to record interval statistics only.

The default interval for statistics recording is every 3 hours. You can change this option only by using the **CEMT SET STATISTICS** command. Statistics counters are reset after each interval. Capturing statistics this frequently can produce a large amount of data. (A sample report we created from one day of interval statistics was several hundred pages.)

▶ Requested statistics

Statistics can be requested at any time by issuing the **CEMT PERFORM STATISTICS** command. This command does not require you to enable statistics recording. Statistics counters are also not reset.

▶ End-of-day statistics

CICS collects and writes this set of statistics at logical end-of-day or when you perform a normal shutdown. The default end-of-day is midnight. You can change this setting by using the **CEMT SET STATISTICS** command.

End-of-day statistics are similar to statistics that are produced by the sample statistics program. These statistics can also be used as a replacement for the shutdown statistics that are produced by previous versions of CICS. Statistics recording does not need to be enabled to collect these statistics.

- Unsolicited Statistics

CICS writes statistics for resources that are dynamically allocated and deallocated. For example, statistics for autoinstall terminals are automatically recorded to DMF before those terminals are deleted after they are logged off.

Processing DMF statistics data

You can use the VSE-supplied skeleton SKDMFPR in ICCF library 59 to create a job stream to process DMF statistics data. A copy of the skeleton that we tailored to produce statistics reports from the DMF data sets is shown in Figure 8-12.

```

* $$ JOB JNM=DFHDMFPR,CLASS=A,DISP=D,NTFY=YES
* $$ LST CLASS=Q,DISP=H
// JOB DFHDMFPR PRINT DMF STATISTICS
* -----*
* THIS JOB PRINTS OFF STATISTIC DATA GATHERED BY DMF *
* -----*
* -----*
* FIRST CLOSE THE DMF DATASET AS FOLLOWS:
* MSG XX,DATA=SETDMF FLUSH .1.
* MSG XX,DATA=SETDMF SWITCH
* WHERE XX IS THE DMF SERVER PARTITION ID.
* -----*
// PAUSE
* *****
* STEP 1: UNLOAD DATA FROM THE DMF DATA SETS
* *****
// DLBL INDD1,'CICSTS.SYSTEM.DFHDMFA',,VSAM,CAT=VSESPUC
* DLBL INDD2,'CICSTS.SYSTEM.DFHDMFB',,VSAM,CAT=VSESPUC
// DLBL OUTDD1,'CICS420.DMF.DATA',0,SD
// EXTENT SYS044,SYSWK1,1,0,63180,15
// ASSGN SYS044,DISK,VOL=SYSWK1,SHR
// EXEC DFHDFOU
INDD ( INDD1, OPTIONS (DUMP)) .2.
INDD ( INDD2, OPTIONS (DUMP))
OUTDD ( OUTDD1, TYPE( 000:255) )
/*
* *****
* STEP 2: SORT, FORMAT AND PRINT THE STATISTICS RECORDS
*
* THE GENERATED OUTPUT DEPENDS ON THE SELECTION CRITERIA ENTERED
* IN DFHSTUP. YOU MAY SELECT ALL ENTRIES FOR A CERTAIN APPLICATION
* AS SHOWN OR YOU ALSO MAY SELECT ALL ENTRIES BY DATE:
* DATE START=MM/DD/YYYY,STOP=MM/DD/YYYY
* *****
// DLBL DFHSTAT,'CICS420.DMF.DATA',0,SD
// EXTENT SYS045,SYSWK1,1,0,63180,15
// ASSGN SYS045,DISK,VOL=SYSWK1,SHR
// DLBL DFHSTWK,'DOS.WORKFILE.SYS.SORT',0,VSAM,CAT=VSESPUC, X
RECORDS=1000,RECSIZE=4096
* DLBL DFHSTWK,'SORT.WORK.FILE.1',0,SD
* EXTENT SYS011,SYSWK1,1,0,63195,15
* ASSGN SYS011,DISK,VOL=SYSWK1,SHR
// EXEC DFHSTUP,SIZE=1M,OS390 .3.
SORT WORK=1
SELECT APPLID=DBDCCICS .4.
DATE START=01/01/2016,STOP=12/31/2016
COLLECTION TYPE=ALL
/*
/&
* $$ EOJ

```

Figure 8-12 Sample DMF statistics print job

The numbers that are highlighted in Figure 8-12 on page 214 correspond to the following information:

1. You must first issue these commands to flush data from the DMF data space and to close the active DMF data set before processing the DMF data. You can also accomplish the same tasks by shutting down DMF.
2. This job step dumps data from the DMF data sets to a sequential file. The input can be the last active DMF data set that was closed or can be multiple DMF data sets. You need one INDD statement for each DMF data set that you want to dump.
3. The CICS statistics utility program DFHSTUP is used to process statistics data from DMF. DFSORT or an equivalent sort product is required. OS390 emulation mode is required for execution of DFHSTUP.
4. DMF records statistics to the DMF data sets for all CICS TS partitions. You can specify parameters to control selection of statistics data by CICS partition, date, and type of statistics.

8.3.3 Monitoring

In addition to CICS statistics, you can capture monitoring data in CICS TS. The CICS Monitoring Facility (CMF) was available in older releases of CICS to collect monitoring data. CMF recorded monitoring data to CICS journals.

In CICS TS, the CICS monitoring domain issues requests to the DMF to record monitoring data to DMF data sets.

You must convert any user-written programs that process CMF data to use the new monitoring record formats that are stored in the DMF data sets.

Note: The IBM CICS Performance Analysis Reporting System (CICSPARS/VSE) product that processed CMF data from previous versions of CICS does not work with monitoring data that is produced by CICS TS. If you have vendor products that process CMF data, check with the vendor for updates to their products.

CICS supplies a sample program, DFH\$MOLS, to process monitoring data that is produced by CICS TS. You can use the source in member DFH\$MOLS.A as an example to create your own program.

VSE supplies a sample job stream to dump and process monitoring data from the DMF data sets. This sample uses DFH\$MOLS to process the data. It is in member DFHMOLS in Interactive Computing and Control Facility (ICCF) library 59.



CICS application program considerations

Because IBM CICS Transaction Server does not support macro-level applications, you must carefully check your applications. This chapter describes the changes that you must apply to your application programs if they use unsupported functions. In addition, it describes the DFHMSCAN and Application Migration Aid utilities that help with the migration.

This chapter includes the following topics:

- ▶ 9.1, “Compatibility” on page 218
- ▶ 9.2, “Migrating macro-level applications” on page 220
- ▶ 9.3, “z/VSE compile dialogs” on page 223
- ▶ 9.4, “CICS Basic Mapping Support (BMS)” on page 224

9.1 Compatibility

All applications that use the command-level interface that is described in *CICS Transaction Server for VSE/ESA Application Programming Reference*, SC33-1658, are source-compatible and object-compatible if the function and the programming language are still supported.

The programming language support for CICS TS is listed in Table 9-1.

Table 9-1 CICS TS Programming language support

Language	Support
Assembler	Assembler and high-level assembler.
LE/VSE Languages	COBOL for VSE/ESA, PL/I for VSE/ESA and C for VSE/ESA.
DOS/VSE COBOL	If link-edited with LE runtime library. (1)
VSE COBOL II	If link-edited with LE runtime library. (1)
DOS PL/I and C/370	Not supported. Recompile with the appropriate LE/VSE compiler
RPG II	Before using RPG II programs, establish the CICS/RPGII translator and catalog-related macros. Use samples from ICCF library 59 RPGINST and RPGSAMPL. The required runtime support (PTF UK60655) is already included in z/VSE 6.1. (2)

Notes:

1. Can run if link-edited with their respective runtime libraries; however, if they fail, they are not supported unless link-edited with LE/VSE runtime library.
2. Note on RPG II: To avoid short-on-storage below the line, the ILNERI RPGII object deck requires execution of a cleanup routine (PTF UK97635) because of unfreed RPGII program storage.

The cleanup program requires following CICS TS CSD definitions:

- Add program RPGIICLN to the CSD:
DEFINE PROG(RPGIICLN) GROUP(group) DA(BELOW) LANG(A)
- Add transaction CRPG to the CSD:
DEFINE TRANS(CRPG) GROUP(group) PROG(RPGIICLN)
Transaction security for CRPG must be allowed.

For more information about application programming interface changes, see *CICS Transaction Server for VSE/ESA Release Guide* and *CICS Transaction Server for VSE/ESA Migration Guide*.

9.1.1 Changes to API commands in CICS TS

Changes in CICS security affected some options on the following **EXEC CICS ASSIGN** and **EXEC CICS ADDRESS** commands:

- ▶ **EXEC CICS ASSIGN OPERKEYS** and **OPSECURITY** are not supported.
- ▶ **EXEC CICS ASSIGN USERNAME** returns the Programmer Name when the Basic Security Manager (BSM) is used. The field can be blank or have a value, depending on whether the External Security Manager (ESM) supports it.

- **EXEC CICS ADDRESS ACEE** enables an application to obtain security-related information that is available on earlier releases through methods that are now obsolete.

You can redesign applications that depend on these fields to use the **USERID** field on the **EXEC CICS ASSIGN** command.

The **EXEC CICS ADDRESS CSA** command was removed because CICS does not allow direct access to control blocks.

The **&DFHEIMX** global macro in application programs is no longer supported. You must remove this macro when you convert mixed-mode programs that contain both macro-level and command-level statements to all command levels.

EXEC DLI commands and their options are unchanged. If you have CICS DL/I programs that use the **CALLDLI** interface, you must change these interfaces to use the user interface block (UIB) rather than the task control area (TCA) to check return codes.

The built-in-functions program and the **BFP** parameter in the **SIT** are no longer supported. However, the command level built-in function **EXEC CICS BIF DEEDIT** is still supported.

CICS TS does not support macro-level application programs. You must convert these applications to command level and one of the supported programming languages to run them on CICS TS.

9.1.2 Changes to rounding for **ASKTIME** and **FORMATTIME** commands

Before CICS TS for z/VSE 2.1, **ABSTIME** values and formatted times that are returned by using **EXEC CICS** commands were rounded up or down to the nearest 100th of a second. Now, they are always truncated and the time is available in milliseconds. If you require the rounding behavior, you can code your application to perform rounding by using the following commands:

- **EXEC CICS ASKTIME ABSTIME**

The **ABSTIME** value that is returned by the **EXEC CICS ASKTIME** command is no longer rounded to the nearest 1/100 second. The absolute time that is returned is the system time-of-day clock, which is adjusted for the local time zone offset, truncated to the millisecond, and returned as a packed decimal of length 8 bytes. It represents the number of milliseconds since 00:00 on 1 January 1900 in the local time zone and is adjusted for Daylight Saving Time.

- **EXEC CICS FORMATTIME**

Before CICS TS for z/VSE 2.1, the **EXEC CICS FORMATTIME** command rounded up a returned time if the number of milliseconds was greater than 500, except in the case where the **ABSTIME** argument contained a value that represented the half-second before midnight, when no rounding was performed, and the **TIME** option returned 23:59:59. This rounding is no longer carried out and the returned time (for example, with the **TIME** option) is given with the number of completed seconds. You can use the new **MILLISECONDS** option to obtain the number of milliseconds and you can perform your own rounding if you need to replicate the former behavior of the command.

9.1.3 Changes to INQUIRE SYSTEM command

The OSLEVEL option was added to the **INQUIRE SYSTEM** command. This option returns a 6-byte field that shows the version, release, and modification level of the z/VSE product on which CICS TS for z/VSE is running. For example, z/VSE Version 6 Release 1 Modification Level 0 returns the value 060100.

9.2 Migrating macro-level applications

This section describes how to migrate macro-level applications from CICS/VSE 2.3 to command-level programs in CICS TS. The utilities are available that help you migrate your macro-level applications:

- ▶ DFHMSCAN
- ▶ Application Migration Aid

These utilities are described next.

9.2.1 DFHMSCAN

The CICS supplied program DFHMSCAN can help you identify programs that contain CICS macro-level interfaces.

Creating a summary report

You can run the job stream that is shown in Figure 9-1 to analyze programs in your VSE application libraries for macro-level code and produce a summary listing.

```
* $$ JOB JNM=DFHMSCAN,CLASS=0,DISP=D,NTFY=YES
* $$ LST CLASS=Q,DISP=H
// JOB DFHMSCAN SCAN FOR MACRO PROGRAMS
// ASSGN SYS001,SYSLST
// ASSGN SYS002,SYSLST
// LIBDEF PHASE,SEARCH=(PRD2.USER,PRD1.BASE) .1.
// EXEC DFHMSCAN,PARM='PRD2.USER,$SUMMARY' .2.
/*
/&
* $$ EOJ
```

Figure 9-1 DFHMSCAN sample summary report job

The numbers that are highlighted in Figure 9-1 correspond to the following information:

1. The sublibrary that contains DFHMSCAN must be in the LIBDEF search chain. You must include the sublibrary to be scanned in the LIBDEF statement and in the PARM statement.
2. All of the phases in sublibrary PRD2.USER are scanned for macro-level code. Only one sublibrary can be scanned in each execution of DFHMSCAN.

A report is produced like the report that is shown in Figure 9-2. The report identifies the following information:

- Module name and size
- Type of program: CICS module(CICSMOD) or language the program is written in
- Number of macro-level (ML) and command-level (CL) statements
- Number of unrecognized BALR statements: UR 14,14 and UR 14,15

DFHMSCAN PROGRAM - SUMMARY LISTING					SUBLIBRARY: PRD2.USER	
MODULE	SIZE	TYPE	ML STMTS	CL STMTS	UR 14,14	UR 14,15
DFHMSCAN	000028F8	CICSMOD	0	0	3	2
DFHPEP	00000122	ASSEMBLR	0	3	0	2
DFHZNEP	000005CA	CICSMOD	0	0	0	11
DFHOSTAT	0001854A	CICSMOD	0	0	0	85
DFHOSTM	000003A8	CICSMOD	0	0	0	0
IESZATCO	000010D2	ASSEMBLR	0	20	0	2
IESZATDX	000010CA	ASSEMBLR	0	20	0	2
OLDPEP	00000050	ASSEMBLR	1	0	0	0
OLDZNEP	0000061A	ASSEMBLR	1	0	0	4
TOTAL NO.		NUMBER	ASSEMBLER	COBOL	PL/I	
MODULES		OF MACRO	MACRO	MACRO	MACRO	
SCANNED		PROGRAMS	PROGRAMS	PROGRAMS	PROGRAMS	
9		2	2	0	0	

Figure 9-2 Sample DFHMSCAN summary report

Creating a detailed report

After you identify programs with macro-level statements, you can run the job stream that is shown in Figure 9-3 to obtain more information about these programs.

```
* $$ JOB JNM=DFHMSCAN,CLASS=0,DISP=D,NTFY=YES
* $$ LST CLASS=Q,DISP=H
// JOB DFHMSCAN SCAN PROGRAMS FOR MACROS
// ASSGN SYS001,SYSLST
// ASSGN SYS002,SYSLST
// LIBDEF PHASE,SEARCH=(PRD2.USER,PRD1.BASE)
// EXEC DFHMSCAN,PARM='PRD2.USER,OLDPEP,OLDZNEP' . .
/*
/&
* $$ EOJ
```

Figure 9-3 DFHMSCAN sample detail report job

Only the phases that are listed in the PARM statement are scanned for macro-level code in sublibrary PRD2.USER.

The resulting report is shown in Figure 9-4. For the selected phases, the report lists the following information:

- ▶ Each BALR statement found, its offset, and what macro it appears to be
- ▶ An interpretation of the type of macro statements that are found

DFHMSCAN PROGRAM - DETAILED LISTING						SUBLIBRARY: PRD2.USER	
PROGRAM MODULE NAME - OLDPEP / LOAD MODULE SIZE - 00000050 / PROGRAM ENTRY POINT - 005405A0							
OFFSET	ADDRESS	STORAGE CONTENT(HEX)				MACRO, COMMAND, KEYWORD AND/OR COMMENT	

00000018	005405B8	9288C080D207C084A0269200C08158E0D0E805EE DFHPC				XCTL	
00000026	005405C6	58E0D0E805EE9210C0809200C08158E0D0E805EE DFHPC				XCTL	
MODULE	SIZE	TYPE	ML STMTS	CL STMTS	UR 14,14	UR 14,15	

DFHMSCAN PROGRAM - SUMMARY LISTING						SUBLIBRARY: PRD2.USER	
MODULE	SIZE	TYPE	ML STMTS	CL STMTS	UR 14,14	UR 14,15	

OLDPEP	00000050	ASSEMBLR	1	0	0	0	
OLDPEP	00000050	ASSEMBLR	1	0	0	0	

Figure 9-4 Sample DFHMSCAN detail report

After you inventory your macro-level programs and identify the macro statements in them, you can convert them manually or use a conversion aid, such as the CICS Application Migration Aid (AMA).

9.2.2 CICS Application Migration Aid

The AMA converts COBOL, PL/I, or assembler language macro-level source programs to command level. Most macro-level code is converted directly to equivalent command-level statements.

Only macros that are described in *CICS Transaction Server for VSE/ESA Application Programming Reference* are converted. DFHFC TYPE=DL/I macro statements are not converted. Diagnostics are provided for this statement and other statements that cannot be converted.

AMA is no longer included with z/VSE 6.1, as described in *CICS Transaction Server for VSE/ESA Application Migration Aid Guide*, SC33-1943.

9.3 z/VSE compile dialogs

The z/VSE 6.1 Interactive User Interface (IUI) compile dialogs changed. You cannot generate compile job streams for DOS/VS COBOL, DOS PL/I, and C/370 by using the IUI dialogs. You must create your own compile procedures for these languages if you use the dialogs for them.

Note: FORTRAN language is batch only.

The IUI Compile window input for the compilation of RPG II online program is shown in Figure 9-5.

IESLIBM		COMPILE JOB GENERATION	
SOURCE MEMBER:	RPGTEST		
SOURCE TYPE.....	1	1=Online Program 3=Map Definition	2=Batch Program 4=Batch Subroutine
LANGUAGE.....	5	1=HLASM 4=C VSE	2=PL/I VSE 5=RPG II 3=COBOL VSE 6=FORTRAN
TEMPLATE.....	2	1=Yes, 2=No	
DB2 SERVER.....	2	1=Yes, 2=No	
DL1	3	1=CALL IF, 2=HLPI, 3=No	
CATALOG.....	1	1=Yes, 2=No	
JOBNAME.....	COMWACK	Name of the job to generate	
OUTPUT MEMBER.....	_____	Leave blank to submit the job immediately. Enter a name and a password (optional) to save it (existing member is overwritten).	
PASSWORD.....			
PF1=HELP		3=END	4=RETURN

Figure 9-5 z/VSE 6.1 Compile window: RPG II program

9.4 CICS Basic Mapping Support (BMS)

CICS BMS mapset DSECTs and objects are upward source-compatible and object-compatible.

BMS mapset objects can be loaded above or below the 16 MB line. The IUI compile window that is used to process a BMS mapset is shown in Figure 9-6. The mapset compile job stream that is generated by the IUI compile dialog is shown in Figure 9-7 on page 225.

IESLIBM		COMPILE JOB GENERATION	
SOURCE MEMBER:	DFHOSTM		
SOURCE TYPE.....	3	1=Online Program 3=Map Definition	2=Batch Program 4=Batch Subroutine
LANGUAGE.....	1	1=HLASM 4=C VSE	2=PL/I VSE 5=RPG II 3=COBOL VSE 6=FORTRAN
TEMPLATE.....	2	1=Yes, 2=No	
DB2 SERVER.....	2	1=Yes, 2=No	
DL1	3	1=CALL IF, 2=HLPI, 3=No	
CATALOG.....	1	1=Yes, 2=No	
JOBNAME.....	COMWACK	Name of the job to generate	
OUTPUT MEMBER.....	comwa88_	Leave blank to submit the job immediately. Enter a name and a password (optional) to save it (existing member is overwritten).	
PASSWORD.....			
PF1=HELP		3=END	4=RETURN

Figure 9-6 IUI Compile window: Mapset

The BMS mapset is cataloged with the RMODE and AMODE options that are shown in Figure 9-7 on page 225. Both 24-bit and 31-bit application programs can access BMS mapsets that are loaded above the line.

Also shown in Figure 9-7 on page 225 is Part 1 of the mapset compile job stream that is generated by the IUI compile dialog.

```

* $$ JOB JNM=COMWACK,DISP=D,CLASS=A,NTFY=YES
* $$ LST DISP=D,CLASS=Q,PRI=3
// JOB COMWACK COMPILE PROGRAM DFHOSTM
// SETPARM CATALOG=1
// SETPARM HTM=2
// IF CATALOG = 1 THEN
// GOTO CAT
// OPTION NODECK,ALIGN,LIST,SYSPARM='MAP'
// GOTO GENER
/. CAT
// LIBDEF PHASE,CATALOG=PRD2.USER
// OPTION CATAL,NODECK,ALIGN,LIST,SYSPARM='MAP'
    PHASE DFHOSTM,*
    MODE RMODE(ANY),AMODE(31)
/. GENER
// EXEC ASMA90,SIZE=(ASMA90,64K),PARM='EXIT(LIBEXIT(EDECKXIT)),SIZE(MAXC
    -200K,ABOVE)'
    PRINT NOGEN
* $$ SLI ICCF=(DFHOSTM),LIB=(0010)
/*
// IF CATALOG NE 1 OR $MRC GT 4 THEN
// GOTO ENDM
// EXEC LNKEDT,SIZE=256K
/*
* $$ PUN DISP=I,DEST=*,PRI=9,CLASS=A
// ASSGN SYSIPT,SYSRDR
// EXEC IESINSRT
$ $$ LST DISP=D,CLASS=Q,PRI=3
#/ JOB COMWACK CATALOG MAP DFHOSTM
// EXEC LIBR
    ACCESS SUBLIB=PRD2.USER
    CATALOG DFHOSTM.A REPLACE=YES
* $$ END
// ON $CANCEL OR $ABEND GOTO ENDJ2
// OPTION NOLIST,ALIGN,DECK,SYSPARM='DSECT'
// EXEC ASMA90,SIZE=(ASMA90,64K),PARM='EXIT(LIBEXIT(EDECKXIT)),SIZE(MAXC
    -200K,ABOVE)'
    PRINT NOGEN
* $$ SLI ICCF=(DFHOSTM),LIB=(0010)
/*

```

Figure 9-7 IUI Generated BMS mapset compile job (Part 1 of 2)

Part 2 of the mapset compile job stream that is generated by the IUI compile dialog is shown in Figure 9-8.

```
/. ENDJ2
// EXEC IESINSRT
/*
#&
* $$ END
// IF HTM NE 1 THEN
// GOTO ENDM
// EXEC IESINSRT
$ $$ LST DISP=D,CLASS=Q,PRI=3
#/ JOB COMWACK CATALOG HTML DFHOSTM
// EXEC LIBR
    ACCESS SUBLIB=PRD2.DFHDOC
* $$ END
// ON $CANCEL OR $ABEND GOTO ENDJ3
// OPTION NOLIST,ALIGN,DECK,SYSPARM='TEMPLATE'
// EXEC ASMA90,SIZE=(ASMA90,64K),PARM='EXIT(LIBEXIT(EDECKXIT)),SIZE(MAXC
    -200K,ABOVE)'
    PRINT NOGEN
* $$ SLI ICCF=(DFHOSTM),LIB=(0010)
/*
/. ENDJ3
// EXEC IESINSRT
/*
#&
$ $$ EOJ
* $$ END
/. ENDM
/&
* $$ EOJ
```

Figure 9-8 IUI Generated BMS mapset compile job (Part 2 of 2)



CICS problem determination

This chapter helps you make a quick start with the tracing and dumping facilities available with the IBM CICS Transaction Server and includes the following topics:

- ▶ 10.1, “CICS Transaction Server for z/VSE 2.1 environment overview” on page 228
- ▶ 10.2, “CICS tracing” on page 228
- ▶ 10.3, “CICS dumps” on page 245
- ▶ 10.4, “DUMP TABLE facility” on page 250
- ▶ 10.5, “CSFE and CEDF” on page 252

10.1 CICS Transaction Server for z/VSE 2.1 environment overview

The IBM CICS Transaction Server (CICS TS) for z/VSE 2.1 environment provides for greater flexibility and control of the tracing activity.

You can determine the following information for each specific case:

- ▶ The CICS domain or subcomponent that you want to trace
- ▶ The level of detail of the generated trace entries
- ▶ The scope of the trace for the following components:
 - A specific transaction
 - A specific terminal
 - Both a transaction and a terminal

The most significant changes in problem determination under CICS TS for z/VSE 2.1 are in the following areas:

- ▶ CICS traces:
 - New exception trace entries
 - New trace control keywords in the SIT
 - New CICS supplied transaction CETR
 - Utility program DFHTU420 to format auxiliary traces
 - INFOANA's DFHPD420 exit to format the CICS Internal Trace table from a SYSDUMP
- ▶ CICS dumps:
 - New keywords for memory dump control in the SIT
 - CEMT and CICS commands to control the DUMP table
 - Print memory dump exit for INFOANA: DFHPD420
 - Dump utility program DFH DU420 to format transaction memory dumps
- ▶ Changes to CSFE and CEDF
 - CSFE new syntax for TASK and TERMINAL subpools
 - CEDF support for remote transactions

10.2 CICS tracing

The following different types of CICS tracing types are available:

- ▶ STANDARD TRACE

This trace is performed by the trace domain at predetermined trace points in CICS code during the regular flow of control.
- ▶ SPECIAL TRACE

This trace is taken for selected tasks and for selected terminals at trace points other than the standard trace point. You can run special tracing, even if the flag for the master (standard) tracing is turned off.

- ▶ **USER TRACE**

This trace is requested by the application program (AP domain). Entries have point IDs in the range of AP0000 to AP00C2, and the numeric part of the entry ID is specified in the application.

- ▶ **EXCEPTION TRACE**

Trace entries are generated by CICS when exceptional or abnormal conditions are detected as part of the first failure data capture mechanism. This tracing cannot be suppressed; instead, entries are always written to the internal trace table.

10.2.1 Trace levels

Trace levels can vary in value 1 - 32 (most of the mainline trace points have a trace level of 1 or 2). The following trace values are available:

- ▶ **LEVEL-1 trace points**

These trace levels are designed to provide enough information to fix user errors. They are in the following areas:

- On entry to, and return from, every CICS domain. The information includes the domain call parameter list and the address of any data that is useful for a high-level understanding of the function to be performed.
- On entry to, and return from, major internal functions within a CICS domain. The information includes parameters that are passed on the call and any output from the function.
- Before and after calls to other programs; for example, VTAM.

- ▶ **LEVEL-2 trace points**

These levels are situated between LEVEL-1 trace points and assist with determining problems within the CICS code.

- ▶ **LEVEL-3 and above**

These levels are reserved for special cases and are likely to be used by IBM personnel only. Few components have trace levels above 2.

10.2.2 Control options

The initial settings for tracing are coded by using the appropriate keywords in the system initialization parameters.

The following system default settings are set by the CICS TS and z/VSE supplied samples for the system initialization table and provide trace settings similar to the CICS/VSE V2 environment:

- ▶ Standard tracing for all transactions
- ▶ Level-1 trace points for all CICS components

These settings are generally adequate to deal with user problems in a development or test CICS region.

For a CICS production region, you can choose to suppress all background tracing activity for performance. For more information, see “Production regions” on page 235.

With CICS TS for z/VSE 2.1, you can have tracing at the transaction level rather than the system level tracing of the previous CICS versions. For more information, see 10.2.4, “CETR overview” on page 231.

10.2.3 Default SIT options

The trace-related keywords are listed in Table 10-1.

Table 10-1 Trace-related keywords

SIT keyword	Usage
AUXTR=OFF	Disables the auxiliary trace.
AUXTRSW=NO	Disables auxtrace data sets switching.
INTTR=ON	Activates the main storage trace.
SPCTR=(1,2)	Sets levels 1 and 2 for all CICS components that are used by a transaction or a terminal (or both) when selected for special tracing.
SPCTRxx	Activates the special tracing for CICS component xx (not set).
STNTR=1	Sets the detail level for standard tracing for the whole CICS.
STNTRxx	Set the detail level for standard tracing for CICS component xx (not set).
SYSTR=ON	Master system trace flag that obtains standard trace entries of CICS system activity.
TRTABSZ=16	Sets the size of the internal trace table (in KBs).
TRTRANSZ=40	Sets the size of the transaction memory dump trace table (in KBs).
TRTRANTY=TRAN	Specifies that only transaction-related entries should be copied from the internal trace table to the transaction memory dump trace table.
USERTR=ON	Master user trace flag enabling user tracing.

10.2.4 CETR overview

CETR is a CICS supplied transaction with which you can control CICS tracing activity. The overriding options that are set by using CETR prevail for the current and subsequent CICS sessions and are reset only to the initial SIT values after a CICS cold start. When starting CETR from a CICS terminal, the window that is shown in Figure 10-1 is displayed.

CETR	CICS Trace Control Facility		CIC1 DBDCCICS
Type in your choices.			
Item	Choice	Possible choices	
Internal Trace Status	==> STARTED	STArTED, STOpped	
Internal Trace Table Size	==> 4096 K	16K - 1048576K	
Auxiliary Trace Status	==> STOPPED	STArTED, STOpped, Paused .1.	
Auxiliary Trace Dataset	==> A	A, B	
Auxiliary Switch Status	==> NO	NO, NExt, All	
Master System Trace Flag	==> ON	ON, Off .2.	
Master User Trace Flag	==> ON	ON, Off .2.	
When finished, press ENTER.			
PF1=Help 3=Quit 4=Components 5=Ter/Trn 9=Error List			

Figure 10-1 Initial window of CETR transaction

The numbers that are highlighted in Figure 10-1 correspond to the following information:

1. Auxiliary trace data set is stopped.
2. CICS Trace and User Trace are enabled.

The panel displays the current state of CICS Internal and Auxiliary trace and the settings of the master system and user trace flags. The values can be altered by entering new values for the settings.

Note: The following PF keys can be used:

- Online help is started by pressing **PF1**.
- Pressing **PF5** opens the Transaction and Terminal Trace Panel.
- Pressing **PF4** opens the Component Trace Options Panel, as shown in Figure 10-2 on page 232.

CETR	Component Trace Options		CIC1 DBDCCICS
Over-type where required and press ENTER.			PAGE 1 OF 3
Component	Standard	Special	
-----	-----	-----	
AP	1	1-2	
BF	1	1	
BM	1	1	
BR	1	1	
CP	1	1-2	
DC	1	1	
DD	1	1-2	
DH	1	1	
DI	1	1	
DM	1	1-2	
DS	1	1-2	
DU	1	1-2	
EI	1	1-2	
FC	1	1-2	
GC	1	1-2	
IC	1	1	
IE	1	1	
PF: 1=Help 3=Quit 7=Back 8=Forward 9=Messages ENTER=Change			

Figure 10-2 CETR transaction: component trace option panel

The Component Trace panel is used to inquire about and set the STANDARD and SPECIAL trace levels for the individual components of the CICS system. To see the meaning of the components abbreviations, press **PF1**.

Each CICS task is a STANDARD tracing task or a SPECIAL tracing task. This task attribute is analyzed by CICS when the task is attached, depending on the settings for the task and the settings for the terminal where the task is started.

By default, all tasks and terminals are STANDARD. You can override this task attribute on the Terminal and Transaction Trace panel of CETR. It is common practice to set higher trace levels for SPECIAL tracing than for STANDARD (see Figure 10-2).

You can disable tracing for components that are not involved with the problem you are debugging by entering OFF for the trace level for the component. If you set one terminal for SPECIAL and another terminal for STANDARD, you can debug the same transaction with different trace options, depending on the terminal from where you start the transaction.

10.2.5 Using CETR

When you press the **PF5** key in the Trace Control Facility panel, the Terminal Tracing panel is displayed. which is shown in Figure 10-3.

CETR	Transaction and Terminal Trace	CIC1 DBDCCICS
Type in your choices.		
Item	Choice	Possible choices
Transaction ID	==> TR01	Any valid 4 character ID
Transaction Status	==> SU	STandard, SPecial, SUPpressed
Terminal ID	==>	Any valid Terminal ID
Netname	==>	Any valid Netname
Terminal Status	==>	STandard, SPecial
Terminal ZCP Trace	==>	ON, OFF
When finished, press ENTER.		
PF1=Help 3=Quit 9=Error List		

Figure 10-3 CETR transaction: transaction and terminal trace panel: SU option

Enter a valid transaction ID or a valid terminal ID and press **Enter** to display the current tracing option for that transaction (STANDARD, SPECIAL, or SUPPRESSED) or terminal (STANDARD or SPECIAL).

Note: By default, STANDARD tracing is set for all transactions and terminals.

To suppress tracing for transaction TR01, complete the following steps:

1. Enter TR01 in the Transaction ID field.
2. Replace the default STANDARD in the transaction status field with the SU option.
3. Press **Enter**.

Now we set terminal A001 for STANDARD tracing, as shown in Figure 10-4.

CETR	Transaction and Terminal Trace		CIC1 DBDCCICS
Type in your choices.			
Item	Choice	Possible choices	
Transaction ID	====> TR01	Any valid 4 character ID	
Transaction Status	====> STANDARD	Standard, SPecial, Suppressed	
Terminal ID	====> A001	Any valid Terminal ID	
Netname	====> D0610001	Any valid Netname	
Terminal Status	====> STANDARD	Standard, SPecial	
Terminal ZCP Trace	====> OFF	ON, OFF	
When finished, press ENTER.			
PF1=Help		3=Quit	9=Error List

Figure 10-4 CETR transaction: transaction and terminal trace panel - ST option

Next, we clear the TRANSACTION ID field and set terminal A002 for SPECIAL, as shown in Figure 10-5.

CETR		Transaction and Terminal Trace		CIC1 DBDCCICS	
Type in your choices.					
Item		Choice		Possible choices	
Transaction ID		====>		Any valid 4 character ID	
Transaction Status		====>		STandard, SPecial, SUPpressed	
Terminal ID		====>	A002	Any valid Terminal ID	
Netname		====>	D0620001	Any valid Netname	
Terminal Status		====>	SPECIAL	STandard, SPecial	
Terminal ZCP Trace		====>	OFF	ON, OFF	
When finished, press ENTER.					
PF1=Help		3=Quit		9=Error List	

Figure 10-5 CETR transaction: transaction and terminal trace panel: SP option

The following settings are available:

- ▶ TR01 generates STANDARD trace entries when started from terminal A001.
- ▶ TR01 and any other transaction that is started from terminal A002 generates SPECIAL trace entries.
- ▶ TR01 is not be traced if it is started from any other (non-SPECIAL) terminal or by using the EXEC CICS START TRANSID command.

10.2.6 Tracing scenarios

Setting up the traces depends on the activities you perform. In this section, we describe the following scenarios:

- ▶ Tracing for production regions
- ▶ Tracing for development regions

Production regions

To eliminate the overhead of background tracing, complete the following steps:

1. Code the following SIT keywords:

- SYSTR=OFF

This keyword switches off the system master trace flag: all system tracing, except special and exception traces, are suppressed. No equivalent master trace flag is needed to turn the special trace on or off.

– AUXTR=ON

This keyword activates auxiliary tracing. However, SYSTR=OFF suppresses all of the standard tracing in the system. The only entries that are written to the auxiliary trace data set (assuming you do not specify special tracing for any task or terminal) are the exception trace entries.

In a long CICS run (typical in production regions) and because the (default) internal trace table size of 16 KB, it is likely that the internal trace table can fill up with the exception entries. During the subsequent wraparound, some of the entries can be lost. With AUXTR=ON, the internal trace table is used as a buffer, which ensures that all exception entries are captured in the auxiliary trace data sets.

If an unplanned system abend occurs, the exception trace entries are available for a first-hand diagnosis. It is unlikely that this information alone is enough to fix the problem, but it can lead you to identify a failing module or transaction. In this case, you can use CETR to set one terminal to SPECIAL; then, start the transaction you want to debug from the terminal you set. After you complete the process, do not forget to reset the terminal attribute to STANDARD.

– AUXTRSW=ALL

This keyword enables continuous switching between the two auxiliary trace data sets: DFHAUXT and DFHBUXT.

2. Ensure that no special tracing was specified for any task or terminal.
3. Set AUXTRACE to STARTED from the CETR initial window.

Development regions

The default settings are adequate for development or test regions (standard tracing is generated for all transactions).

If you must generate SPECIAL tracing when debugging a particular transaction, use CETR to set one terminal to SPECIAL and start the transaction that you want to trace from that terminal.

10.2.7 Trace formatting

The trace formatting utilities are shown in Table 10-2.

Table 10-2 Trace formatting utilities

To format	From	Utility
Internal trace table	CICS system memory dump	DFHPD420
Internal trace table	Transaction memory dump	DFHDU420
Auxiliary trace	Aux trace data set	DFHTU420

INFOANA exit DFHPD420

DFHPD420 is the exit for INFOANA to format and print CICS SDUMPs from the VSE Dump Library.

The CICS Internal Trace Table is printed when formatting the system memory dump if the Trace Domain is selected by using the following methods:

- ▶ Using the component keyword with a nonzero operand, for example TR=1.
- ▶ Using the nonzero DEFault, for example DEF=2.

Memory dumps from previous CICS versions cannot be formatted by using this exit.

To collect the CICS memory dumps that are generated by using the SDUMP macro, the following conditions are required:

- ▶ SYSDUMP option must be active for the partition.
- ▶ LIBDEF DUMP information must be available
- ▶ Enough space must be available in the VSE dump library.

Note: If a Dump Library Full condition arises when a CICS dump is being written, an unformatted dump is printed in SYSLST unless the VSE option SYSDMPC is set for the partition, in which case the dump is suppressed.

Formatting an SDUMP

A job to format a CICS dump is shown in Figure 10-6.

```
* $$ JOB JNM=PRTDUMP,DISP=D,PRI=8,NTFY=YES,CLASS=0
* $$ LST DISP=H,RBS=1000
// JOB PRTDUMP
// EXEC PROC=DTRINFOA
// EXEC INFOANA,SIZE=INFOANA
DUMP NAME SYSDUMP.F2.DF200007
SELECT DUMP VIEWING
CALL DFHPD420 DATA DEF=3,KE=1,SM=1,FCP=2,DS=0
/*
/&
* $$ E0J
```

Figure 10-6 Job to format a CICS memory dump

DFHPD420 is called to format the dump. The dump exit parameters are coded following the INFOANA DATA keyword.

The syntax of the component parameter is component-keyword=level of detail, where the following levels of detail can be used:

- ▶ 0: Suppress output for this component
- ▶ 1: Summary only (1 output line per entry)
- ▶ 2: Full format of control blocks
- ▶ 3: Summary and full format (that is, 1 and 2)

This job produces the following items:

- ▶ Summary and full format output for all the components that are not selected (DEF=3).
- ▶ Summary output only for the Kernel and Storage domains (KE=1,SM=1)
- ▶ Full format (but no summary) for the File Control Program (FCP=2)
- ▶ All output from the Dispatcher Domain is suppressed (DS=0)

Using the IUI dialogs to format the SDUMP

If you use Interactive User Interface (IUI) dialogs (fast path 4.3 from the IESEADM panel and select Analyze CICS Dump option), an INFOANA job stream is generated with the following options:

AP=3 KE=3 DS=3 TR=3 LD=3

The job prints abbreviated plus full format trace entries for the Application, Kernel, Dispatcher, Trace, and Loader domains.

You can store the JECL in your ICCF library and modify the selection according to your needs before submitting it for running. The INFOANA always prints the dump summary, even if no valid, or null, selection options are present, as shown in Figure 10-7.

```

=== DUMP SUMMARY

DUMPID:    49/0020
DUMPCODE:  SR0001 .1.
DATE/TIME: 15/02/23 15:12:24 (LOCAL) .2.
MESSAGE:   DFHSR0001 DBDCCICS An abend (code 0C4/AKEA) has occurred at
offset
SYMPTOMS:  PIDS/564805400 LVLS/420 MS/DFHSR0001 RIDS/DFHSRP PTFS/zVSE420 .3.
TITLE:     (None)
CALLER:    (None)
ASID:      X'0000'

```

Figure 10-7 Sample of a dump summary

The numbers that are highlighted in Figure 10-7 correspond to the following information:

1. A dump is created because of a storage protection exception.
2. Always check the DATE/TIME to ensure that this memory dump is the memory dump that you think it is.
3. This line shows the CICS component ID, release, error message (if any), and affected module.

Formatting the CICS Internal Trace table

If all you want to format from the SDUMP is the CICS Internal Trace table, complete the following steps (see Figure 10-8):

1. SELECT the TRACE COMPONENT (TR=1 - 3).
2. SUPPRESS the output from all other components (DEF=0).

```

* $$ JOB JNM=PRTDUMP,DISP=D,PRI=8,NTFY=YES,CLASS=0
* $$ LST DISP=H,RBS=1000
// JOB PRTDUMP
// EXEC PROC=DTRINFOA
// EXEC INFOANA,SIZE=INFOANA
DUMP NAME SYSDDUMP.F2.DF200007
SELECT DUMP VIEWING
CALL DFHPD420 DATA DEF=0,TR=3
/*
/&
* $$ EOJ

```

Figure 10-8 Sample job to format the trace table

The job produces a summary and a full formatted record (TR=3) for every trace entry (see Figure 10-9).

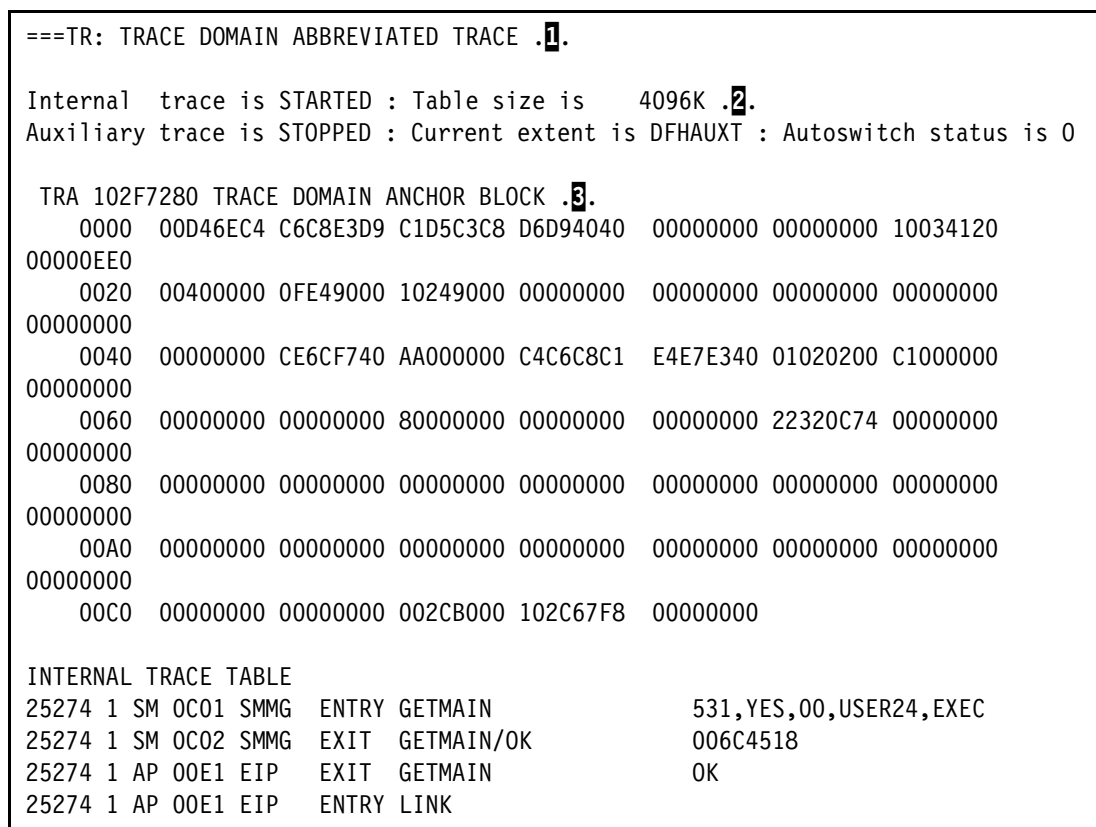


Figure 10-9 The SUMMARY trace report

The numbers that are highlighted in Figure 10-9 correspond to the following information:

1. This header is the header of the Trace Domain report.

Note: When browsing the formatted SDUMP, you can locate the header of a component report by the string of three equal signs followed by the component ID; for example, = = = S M marks the beginning of the Storage Management report.

2. This section shows information about the status of the trace environment when the dump was performed.
3. These entries are abbreviated trace entries.

The FULL trace report is shown in Figure 10-10.

```

===TR: TRACE DOMAIN FULL TRACE

INTERNAL TRACE TABLE

SM 0C01 SMMG ENTRY - FUNCTION(GETMAIN) GET_LENGTH(531) SUSPEND(YES)
INITIAL_IMAGE(00) STORAGE_CLASS(USER24) CALLER(EXEC)

TASK-25274 KE_NUM-003C TCB-002D4000 RET-8377E97C
TIME-14:32:02.1484335200 INTERVAL-**.***** =000001=
1-0000 00780000 00000011 00000000 00000000 B6780000 00000000
0259018C 8058ADC0 *.*****
0020 03C60F40 03CF9030 00000531 00020208 0179CC4A 01001201
03D9DE40 00000000 *.F. ....¢....R. ....*
0040 00780000 00000011 00000000 00000000 BC100000 00000000
04370134 C5E3E2C6 *.*****.ETSF*
0060 D9C5C540 006C4518 00000520 006C3898 0413F9F4 01000600
*REE .%.%.%.q..94.... *

SM 0C02 SMMG EXIT - FUNCTION(GETMAIN) RESPONSE(OK) ADDRESS(006C4518)

TASK-25274 KE_NUM-003C TCB-002D4000 RET-8377E97C
TIME-14:32:02.148 *REE .%.%.%.q..94.... *
1-0000 00780000 00000011 00000000 00000000 B6780000 00000000
0259018C 8058ADC0 *.*****
0020 03C60F40 006C4518 00000531 00020208 0179CC4A 01001201
03D9DE40 00000000 *.F. .%.%.%.¢....R. ....*
0040 00780000 00000011 00000000 00000000 BC100000 00000000
04370134 C5E3E2C6 *.*****.ETSF*
0060 D9C5C540 006C4518 00000520 006C3898 0413F9F4 01000600
*REE .%.%.%.q..94.... *
```

Figure 10-10 Corresponding FULL trace report

The FULL trace entry contains the following control blocks and Data information that is not shown in the corresponding abbreviated trace:

- ▶ Task number assigned by the Kernel
- ▶ TCB address of the VSE subtask
- ▶ Call return address
- ▶ Time stamp
- ▶ Data area

For more information about trace entries, see *CICS Transaction Server for z/VSE Trace Entries Handbook*, SX33-6108.

Trace entry selection

The DFHPD420 exit does not provide the capability to select trace entries from the internal trace table; instead, all entries are processed. The only choice that is available is the level of detail that is coded with the T R = component keyword.

If you want to use any valid selection parameters that are available to the auxiliary trace print utility DFHTU420 to format the internal trace table of an SDUMP, consider the following points:

- ▶ The TRACE domain must be selected: T R = n (where n is not zero).
- ▶ Use the TRace Selection (TRS=) parameter as a component keyword and code as the second operand on any of the select options supported by DFHTU420.

For more information about the selection parameters that are available for DFHTU420, see “Formatting the auxiliary trace” on page 243.

An example of the use of the TRS parameter is shown in Figure 10-11.

```
* $$ JOB JNM=PRTDUMP,DISP=D,PRI=8,NTFY=YES,CLASS=0
* $$ LST DISP=H,RBS=1000
// JOB PRTDUMP
// EXEC PROC=DTRINFOA
// EXEC INFOANA,SIZE=INFOANA
  SELECT DUMP MANAGEMENT
    DUMP NAME SYSDUMP.F2.DF200007
    RETURN
  SELECT DUMP VIEWING
    CALL DFHPD420 DATA DEF=0,
    DATA TR=2,
    DATA TRS=<TERMID=(A000)>
/*
/&
* $$ E0J
```

Figure 10-11 Using the TRS parameter

The TRS parameter suppresses entries from all components. These format entries are from the Trace Domain only. The TRS selection parameters (similar to the auxiliary trace utility, DFHTU420) must be enclosed in angled brackets.

Note: For INFOANA, the only valid continuation character in column 72 is the minus sign.

Formatting a transaction dump

The output of the CICS Dump domain is written to DASD or tape (depending on how you created and defined the transaction dump data sets to CICS).

The transaction dump utility DFHDU420 performs the following tasks:

- ▶ Writes a summary in the form of an index of all the dumps stored in the dump data set that is processed. The report is written to a print file that is assigned by the logical unit SYS009.
- ▶ Prepares the transaction dump output for printing.
- ▶ Prints the formatted information or scans the dump data set and prints a summary of its contents.

The internal trace table entries of a transaction dump are formatted as part of the dump processing by using DFHDU420.

By coding the system initialization parameter TRTRANTY=, you can determine the contents of the transaction dump trace table. The following options are available:

► **TRAN**

Only transaction-related entries are copied from the internal trace table into the transaction dump table. (This value is coded in the default SIT.)

► **ALL**

All the entries in the internal trace table are copied.

Control options

The control statements for DFHDU420 are submitted by using SYSIPT, between a SELECT and an END statement.

Consider the following conditions:

- If a SCAN statement is present, all other statements are ignored.
- If no control statements are submitted, all dumps are printed.
- The SELECT TYPE= statement provides the selection criteria for the dumps that you want to print.

You can select dumps that match one or more of the following arguments:

- **TRANID**
- **DUMPCODE**
- **DUMPID**
- **TIME**

You can use fully qualified or generic search arguments and a range of Boolean operands. An example of printing a CICS transaction dump is shown in Figure 10-12.

```
* $$ JOB JNM=DFHDU420,CLASS=A,DISP=D
// JOB SKDFHDUP PRINT CICS DUMP FILE CONTENTS
// PAUSE DID YOU PERFORM A 'CEMT SET DUMP SW' TO CLOSE DFHDMPA/B ?
// DLBL DFHDUMP, 'CICS.DUMPA',,VSAM,                                X
//                                     CAT=VSESPUC,DISP=(OLD,DELETE)
// ASSGN SYS009,SYSLST
// EXEC DFHDU420,SIZE=2048K,PARM='TRANSLATE=FOLD',OS390
//     SELECT TYPE=OR 1.
//         DUMPCODE=(ASRA) 2.
//         TRANID=(PAY*,TR+F) 3.
/*
/&
* $$ EOJ
```

Figure 10-12 Printing a transaction dump

The numbers that are highlighted in Figure 10-12 correspond to the following information:

1. This job sets the OR Boolean operand for the selection that indicates all dumps that match any of the DUMPCODE or the TRANID selection criteria are printed.
2. Specifies dumps with dump code ASRA.
3. Dumps of TRANIDs beginning with PAY followed by any characters and dumps of TRANIDs four characters long, beginning with TR followed by any non-blank character and ending with character F are included.

Formatting the auxiliary trace

The utility program DFHTU420 extracts, formats, and prints all or selected trace entries from one of the auxiliary trace data sets.

Only the auxiliary trace data sets that were opened for the most recent run of CICS can be used as input for DFHTU420. You select the trace entries by submitting the selection parameters by using SYSIPT or the PARM parameter of the EXEC statement.

You can specify that all entries are to be processed (the default setting) or select specific trace entries with certain attributes; for example, entries that are associated with any of the following items:

- ▶ Specific terminal
- ▶ Specific trace identifier (component+trace point number)
- ▶ Specific entry sequence number
- ▶ Specific transaction identifier
- ▶ Specific instance of a transaction ID (that is, a task)
- ▶ Specific kernel task
- ▶ Exception tracing

Control options

The SELECTION control options are listed in Table 10-3.

Table 10-3 SELECTION control options

Option	Usage
ALL	All entries are to be processed (default setting).
ENTRY_NUM	Sequence number (or range) of entries to be processed.
EXCEPTION	Exception trace entries ONLY.
KE_NUM	Kernel task number.
TASKID	Task identification or task number.
TERMID	Terminal identification.
TIMERG	Time period (hhmmss) for which trace entries are to be printed.
TRANID	Transaction identification.
TYPETR	Trace point identification in the format DDnnnn, where DD is the component ID and nnnn is the trace point identification.

The OUTPUT control options are listed in Table 10-4.

Table 10-4 OUTPUT control options

Option	Usage
ABBREV	Abbreviated trace, one line per entry.
FULL	Fully formatted entries (the default setting).
INTERVAL	Time interval between entries you want to highlight; each entry that is generated in the time interval is marked with an asterisk (*) for visual reference.
PAGESIZE	Number of lines per page (default is 55).
TIMESTAMP	To print time stamp instead of sequence number.
UPPERCASE	Output in uppercase (the default is mixed case).

Figure 10-13 shows an example of printing the auxiliary trace.

```

* $$ JOB JNM=DFHAUXPR,DISP=D,CLASS=0
// JOB DFHAUXPR PRINT CICS AUXILIARY TRACE DATASET CONTENTS
*
* THIS JOB CAN BE USED FOR CICS RELATED PROBLEM DETERMINATION
*
* STEP1: ACTIVATE INTERNAL TRACE VIA: 'CEMT SET INTRTRACE START'
* STEP2: ACTIVATE AUXTRACE OPERATING: 'CEMT SET AUXTRACE START'
* STEP3: RUN EVENT TO BE TRACED
* STEP4: STOP TRACING VIA:           'CEMT SET INTRTRACE STOP'
* STEP5: STOP TRACING VIA:           'CEMT SET AUXTRACE STOP'
* STEP6: REPLY 'END/ENTER' TO RUN THIS JOB ANALYZING TRACED DATA
*
* FOR PRINT OPTIONS REFER TO THE CICS MANUALS.
* POSSIBLE OPTIONS ARE E.G. 'ALL,FULL' OR 'ALL,ABBREV'
// PAUSE ---> PLEASE FOLLOW GENERAL INSTRUCTIONS ABOVE !!
// DLBL DFHAUXT,'CICS.AUXTRACE',0,VSAM,                                X
                                CAT=VSESPUC,RECSIZE=4096,              X
                                DISP=(OLD,DELETE),RECORDS=(400,0)
// EXEC DFHTU420,SIZE=1880K,OS390
  ABBREV,                        .1. X
  TRANID=(TECC),                 .2. X
  TIMESTAMP .3.
/*
/&
* $$ EOJ

```

Figure 10-13 Printing the auxiliary trace

The numbers that are highlighted in Figure 10-13 correspond to the following information:

1. This trace is the summary trace, one line per entry.
2. These transactions are traced.
3. This statement specifies that time stamps (maximum interval of 9.999999 seconds) are printed instead of the trace sequence numbers in an abbreviated trace.

You receive the same result by coding the selection statements in this way (see Figure 10-14).

<pre>// EXEC DFHTU420,SIZE=1880K,PARM=' ABBREV,TRANID=(TECC), TIMESTAMP',OS390</pre>	X
--	---

Figure 10-14 Single-line EXEC DFHTU420 statement

10.3 CICS dumps

The most significant changes in the dump environment under CICS TS include the following changes:

- ▶ CICS dumps by using SDUMP macro
- ▶ New dump-related keywords in the SIT
- ▶ Dump tables
- ▶ Extended CEMT commands for dump control
- ▶ New CICS commands for dump control

The following types of dumps can be produced in the CICS environment:

- ▶ Transaction dumps that include the following characteristics:
 - Written to the CICS transaction data sets DFHDMPA/B
 - User-requested in the following ways:
 - EXEC CICS DUMP TRANSACTION
 - By using a Transaction Dump call from a Global User Exit (GLUE).
 - By using the DUMP attribute of the RDO transaction definition.
 - Requested or forced by CICS following a transaction abend
- ▶ CICS dumps include the following characteristics:
 - Produced by using VSE SDUMP/SDUMPX macros
 - Written to the VSE Dump Library (SYSDUMP)
 - User-requested in the following ways:
 - EXEC CICS PERFORM DUMP
 - CEMT PERFORM DUMP or SNAP
 - SYSTEM_DUMP XPI call from a GLUE
 - Requested by global trap or trace exit
 - Requested by a node error program (NEP)
 - Performed by CICS following a system abend

For transaction dumps and system dumps, the dumping environment determines whether the dump is performed.

The CICS System Dump is produced by using VSE SDUMP or SDUMPX macros. SDUMP is used if the invoking program is running in primary address space control (ASC) mode. If invoked from a program that is running in Access Register (AR) mode or if data space is to be dumped, the dump request is issued by using the SDUMPX macro.

The output of a CICS system dump consists of the following components:

- ▶ VSE Supervisor
- ▶ SVA and other common areas
- ▶ CICS private storage areas

10.3.1 Preparing for stand-alone dump

Before using the stand-alone dump facility, you must to create a stand-alone dump tape or disk. Interactive Interface supports creating stand-alone devices. The dialog support is available under Selection 4 of the Interactive Interface administrator panel hierarchy.

We prepared a stand-alone dump disk that uses a spare 3390-3 DASD volume with the following characteristics:

- ▶ Allocate VTOC on end of first cylinder by using ICKDSF statement DOSVTOC(0,10,5)
- ▶ Dump data file starting at track 15
- ▶ Dump data file size of 50.000 tracks
- ▶ The VSE Dump program is placed on first cylinder (track 1 - 7)

The job that is shown in Figure 10-15 was used to create the SA-dump disk.

```
* $$ JOB JNM=SADMPDSK,DISP=D,PRI=9, C
* $$ NTFY=YES, C
* $$ CLASS=0
* $$ LST DISP=H,RBS=1000
// JOB SADMPDSK CREATE STANDALONE DUMP PROGRAM ON DISK
* *****
* * DLBL and EXTENT statements for dump file IJSYSDU
* *****
// DLBL IJSYSDU,'VSE.DUMP.FILE'
// EXTENT ,,,15,50000
* *****
* * DLBL and EXTENT statements for dump program file IJSYSDI
* *****
// DLBL IJSYSDI,'VSE.DUMP.PROGRAM'
// EXTENT ,,,1,7
// EXEC DOSVSDMP,PARM='CREATE DUMP DEVICE=205'
/*
/&
* $$ EOJ
```

Figure 10-15 Job to create SA-dump disk

10.3.2 Processing the CICS dump from a stand-alone disk

The DFHPD420 exit requires some control information from the VSE supervisor and the system GETVIS to be in the same dump file as the CICS TS partition.

Note: If the system hangs, you cannot run CEMT to perform an SDUMP; the only alternative to get a dump of the CICS region is to perform a stand-alone dump.

On a stand-alone dump, the VSE supervisor and the SVA are written into a single file. Each active address space eligible for dumping (per force of the SADUMP=nn option statement) are written as a separate file into the dump output device.

To process the CICS dump from a stand-alone output tape, complete the following steps:

- 1. Scan the stand-alone dump to locate the dump files to be unloaded (see Figure 10-16).

```
* $$ JOB JNM=SCANDMPD,DISP=D,PRI=9,
* $$ NTFY=YES,
* $$ CLASS=0
* $$ LST DISP=H,RBS=1000
// JOB SCANDMPD  SCAN DUMP FILE ON DISK
* *****
* * DLBL and EXTENT statements for dump file IJSYSDU
* *****
// DLBL IJSYSDU,'VSE.DUMP.FILE'
// EXTENT ,,,15,50000
// EXEC DOSVSDMP,PARM='SCAN DEVICE=205'
/*
/&
* $$ EOJ
```

Figure 10-16 Scan the SA-dump disk

Figure 10-17 shows the resulting output.

PRINTOUT OF VSE DUMP DATA SET				
DIRECTORY OF VSE DUMP DATA SET				
DUMP FILE	DUMP TYPE	NAME	DATE	DATA DUMPED
-----	-----	-----	-----	-----
001	SADUMP		2016/10/07	SUPERVISOR+SVA
002	SADUMP		2016/10/07	PMRAS-R
003	SADUMP		2016/10/07	PMRAS-00
004	SADUMP	SECSERV	2016/10/07	FB-PARTITION
005	SADUMP	TCPIP00	2016/10/07	F7-PARTITION
006	SADUMP	VTAMSTR	2016/10/07	F3-PARTITION
007	SADUMP	CICSEXPL	2016/10/07	F2-PARTITION
008	SADUMP	POWSTART	2016/10/07	F1-PARTITION
END OF DUMP				

Figure 10-17 SYSLST output

2. Onload the supervisor and CICS dumps into SYSDUMP (see Figure 10-18).

```
* $$ JOB JNM=DMPONL95,DISP=D,PRI=8, C
* $$ NTFY=YES, C
* $$ CLASS=0
* $$ LST DISP=H,RBS=1000
// JOB DMPONL95 ONLOAD DUMP FROM DISK
// ASSGN SYS009,205
// DLBL IJSYSDU,'VSE.DUMP.FILE'
// EXTENT SYS009,,15,50000
// EXEC PROC=DTRINFOA
// EXEC INFOANA,SIZE=300K
  SELECT DUMP MANAGEMENT
    DUMP NAME SYSDUMP.BG.SUPSVA01
    RETURN
  SELECT DUMP ONLOAD
    VOLID DISK SYS009
    FILE 1 LAST
    RETURN
  SELECT DUMP MANAGEMENT
    DUMP NAME SYSDUMP.F2.CI01EXPL
    RETURN
  SELECT DUMP ONLOAD
    VOLID DISK SYS009
    FILE 7 LAST
    RETURN
  SELECT END
/*
/&
* $$ EOJ
```

Figure 10-18 Onloadiing the dumps

You can now process SYSDUMP.F2.CI01EXPL by using DFHPD420.

Note: You should consider increasing the size of your SYSDUMP library to accommodate the larger dumps of your CICS TS regions. As the product is shipped, it can barely store more than two consecutive SDUMPs.

If you encounter problem, follow-up with the IBM Support Center after opening a PMR. For more information about how to pass data to IBM, see [Hints and Tips for z/VSE 6.1](#).

10.3.3 Program check and abend information

As part of the first failure data capture mechanism, when a program check or abend condition is detected in any domain, the following events occur:

- ▶ An exceptional trace entry is written to the internal trace table.
- ▶ An error message is issued.
- ▶ Program check and abend information is generated in the Kernel domain.

To view this Kernel output, complete the following steps:

1. If you select the output from the Kernel when formatting the dump (KE=n), the Error Table of the Kernel is printed (see Figure 10-19). Each row summarizes and identifies an abend.

==KE: KE Domain Error Table Summary						
ERR_NUM	ERR_TIME	KE_NUM	ERROR TYPE	ERR_CODE	MODULE	OFFSET
=====	=====	=====	=====	=====	=====	=====
00000131	10:02:42	003D	TRAN_ABEND_PERCOLATE	---/AE19	DFHEIP	000015E6
00000132	10:02:43	003D	TRAN_ABEND_PERCOLATE	---/AE19	DFHPCP	000004F2
00000133	10:02:43	003D	TRAN_ABEND_PERCOLATE	---/AE19	DFHEIP	000015E6

Figure 10-19 Kernel Error Table

2. If you select the Kernel output with the full format option (KE=2 or KE=3), the program check and abend information for each of the errors is also formatted. This information consists of the following details:
 - Registers and PSW at the time of failure.
 - A storage report for each task that had a program check or a program abend during the current CICS session.

10.3.4 Default SIT options

The initial dump options are set by using the SIT keywords that are listed in Table 10-5.

Table 10-5 Dump-related keywords

SIT keyword	Usage
DUMP=YES	Enable the dump domain to perform SDUMPs.
DUMPDS=AUTO	During CICS initialization, opens the Transaction Dump data set that was not in use in the previous CICS run.
DUMPSW=NO	If the transaction data set is full, operator is notified to perform the data set switch.
SYDUMAX=999	Sets no limit for system dumps per entry of the system dump table.
TRDUMAX=999	Sets no limit for transaction dumps per entry of the transaction dump table.
TRTRANSZ=40	Transaction dump trace table size in KB.
TRTRANTY=TRAN	Only transaction-related entries should be copied from the internal trace table to the transaction dump trace table.

The Kernel domain always performs an SDUMP if a unrecoverable failure occurs, regardless of the DUMP option in the SIT.

The storage for the Transaction Dump Trace Table is acquired above the 16 MB line by using a VSE GETVIS request that is issued when the transaction abend is detected.

10.4 DUMP TABLE facility

The DUMP TABLE facility enables the CICS users to control the actions to be performed when a particular abend code occurs.

The following dump tables are available:

- ▶ **TRANSACTION DUMP TABLE**
Defines the conditions and the options that are associated with transaction dump codes.
- ▶ **SYSTEM DUMP TABLE**
Defines the conditions and the options that are associated with CICS system dump codes.

10.4.1 Overview

If CICS detects a storage violation, the following events occur:

- ▶ An exception trace entry is written.
- ▶ The Message domain issues message DFHSM0102.
- ▶ CICS performs a system dump with a dump code SM0102. (Most of the system dump codes are the associated message number after the DFH prefix is removed.)
- ▶ The Dump Table is searched to take the actions that are necessary for this type of abend. If an entry is not found, CICS generates a temporary entry for this abend code, assuming default values.

Temporary entries that are created by CICS are not written to the CICS Global Catalog and remain active for the current CICS run only.

You can create a permanent entry in any of the dump tables by using a **CEMT** or **EXEC CICS** command. Permanent entries are written to the Global Catalog and are reset only after a CICS cold start. Each entry defines the actions to be taken by the Dump domain for a particular dump code.

The dump code for both transaction and system abends can be CICS supplied or user-defined.

To add an entry to the System Dump table, use one of the following commands:

- ▶ **CEMT SET SYDUMPCODE**
- ▶ **EXEC CICS SET SYSDUMPCODE**

For the Transaction Dump table, use one of the following commands:

- ▶ **CEMT SET TRDUMPCODE**
- ▶ **EXEC CICS SET TRANDUMPCODE**

The actions and options that you can specify for each of the tables are described next.

10.4.2 Transaction Dump table

Each entry in the Transaction Dump table specifies the following information for a transaction dump code:

- ▶ Whether a transaction dump is to be performed.
- ▶ Whether a system dump is to be performed with or without a transaction dump.

- ▶ Whether CICS is to be ended after an occurrence of this dump code.
- ▶ The maximum number of times this transaction dump code action should be taken during the CICS run, or before the counter is reset.

10.4.3 System Dump table

Each entry in the System Dump table specifies the following information for a system dump code:

- ▶ Whether a system dump is to be performed.
- ▶ Whether CICS is to be ended after an occurrence of this dump code.
- ▶ The maximum number of times this system dump code action should be performed during the CICS run or before the counter is reset.

Note: The CICS Kernel dumps cannot be suppressed.

10.4.4 Dump suppression for ASRA and ASRB abends

With CICS TS, use the dump tables and the enhanced CEMT transaction. With CICS/VSE Version 2, the SIT options PCDUMP= and ABDUMP= (now obsolete) were used for this purpose.

When a VSE abend or program check occurs in the AP domain or in a user application program, CICS issues message DFHSR0001 or DFHAP0001. DFHSR0001 is issued (and abend dump code SR0001 is performed) if the failing application is running in the USER key. However, if the application is running in the CICS key, DFHAP0001 is issued and abend dump code AP0001 is performed.

If you want to suppress CICS dumps when provoked by application errors, but still allow dumps to be performed if the abend is in the CICS code, use CEMT to suppress system dumps for SR0001 by adding the following entry to the systemdump table:

```
CEMT SET SYDUMPCODE(SR0001) ADD NOSYSDUMP
```

If you want to suppress the system dumps for abends and program checks in the CICS code, add the following entry for AP0001:

```
CEMT SET SYDUMPCODE(AP0001) ADD NOSYSDUMP
```

However, you might want to perform a system memory dump for the ASRA but not for the ASRB transaction abends. In this instance, use CEMT to create the entries in the Transaction Dump table, as shown in the following example:

```
CEMT SET TRDUMPCODE(ASRA) ADD SYSDUMP MAX(2) (1)
CEMT SET TRDUMPCODE(ASRB) ADD NOSYSDUMP
```

CICS performs a system dump for the first two ASRA occurrences only.

To display the options set for all defined dump table entries, use the following entry:

```
CEMT INQ TRDUMPCODE (*) and CEMT INQ SYDUMPCODE (*)
```

You can modify any of the entries by typing over the displayed options.

10.5 CSFE and CEDF

This section includes information about issues that must be considered when you are migrating from CICS/VSE 2.3. It also describes changes in CICS transactions CSFE and CEDF.

10.5.1 Changes to the CSFE DEBUG transaction

The syntax of the CSFE DEBUG transaction was changed to match the new SIT overrides that control storage checking.

With CICS/VSE 2.3, you used the following entry to specify that you wanted storage checking for the TASK user storage:

```
CSFE DEBUG,TASKSTG=ON OFF
```

Now, you use the following entry instead:

```
CSFE DEBUG,CHKSTK=(ALL CURRENT NONE)
```

In CICS/VSE 2.3, you specified the following entry when you wanted storage checking for the TERMINAL subpool:

```
CSFE DEBUG,FAQE=...,SUBPOOL=TP
```

Now, you use the following entry instead:

```
CSFE DEBUG,CHKSTRM=(CURRENT NONE)
```

10.5.2 CEDF support for remote transactions

The restriction on the use of CEDF for remote transactions was removed.

In a multiregion operation (MRO) or an intersystem communication (ISC) environment (Advanced Program-to-Program Communication only), you can use CEDF (in single session mode only) for transactions that are defined in the terminal-owning region (TOR) as remote. CICS automatically notifies the application-owning region (AOR) that the transaction is to be run in execution diagnostic facility (EDF) mode.

Note: CICS supports EDF for remote transactions when all the communicating regions are using CICS Transaction Server for z/VSE 2.1 or later only.



IBM-supplied CSD groups

Table A-1 lists some of the CICS System Definition (CSD) groups that are supplied by IBM.

Table A-1 IBM-supplied RDO groups

Group name	To define
CEE	Language environment group
CICREXX *	REXX/CICS
CMCI	CICS Explorer WEB
DFHAI62	Starter APPC connections group
DFHAKP *	Activity keypoint group
DFHBACK *	Dynamic backout group
DFHBMS *	Terminal page retrieval group
DFHCLNT *	CICS client support
DFHCONS *	VSE console support group
DFHCOMP1	Compatibility with CICS/VSE 2.3
DFHCOMP2	Required by users of the report controller
DFHDOC *	CICS Document Handler
DFHEDF *	Execution diagnostic group
DFHFE *	FE terminal test group
DFHFEPI *	Front end programming interface
DFHHARDC *	3270 print support group
DFHINQUI *	Group for EXEC CICS special commands
DFHINTER *	Command interpreter group
DFHIPECI *	CICS ECI over TCP/IP

Group name	To define
DFHISC *	Routing transaction group
DFHISCT *	Intersystem communication group
DFHJRNL *	Journal bootstrap group
DFHMISC *	Miscellaneous group
DFHMISC3 *	Miscellaneous group
DFHMSWIT *	Message switching group
DFHOPCLS *	Dynamic open and close group
DFHOPER *	EXEC master terminal group
DFHPGAIP *	Program manager autoinstall group
DFHRCF *	Reporter controller group
DFHRMI *	Resource manager interface group
DFHRSEND *	VTAM resend program group
DFHRSPLG *	Response logging group
DFHSIGN *	CICS sign on group
DFHSPI *	Resource definition online group
DFHSTAND *	Standard entries group
DFHTCL *	IBM-supplied TRANCLASS
DFHTERM *	Supplied model TERMINAL definitions
DFHTYPE *	Supplied TYPETERM definitions
DFHVTAM *	VTAM group
DFHVTAMP *	VTAM 3270 print group
DFHWEB *	CICS WEB support
EZA	EZA Interface
FCTC2	Supplied VSE FILE definitions for 2nd CICS
FCTSP	Supplied VSE FILE definitions for primary CICS
SMSSEYU	CICS Explorer WEB
TCPIP	TCP/IP environment group
VSEAI62	Sample DFHZATDY
VSETERM	Supplied VSE TERMINAL definitions for autoinstall
VSETERM1	Supplied VSE TERMINAL definitions during installation
VSETYPE	Supplied VSE TYPETERM definitions for autoinstall
VSESPG	Supplied VSE entries by DFHPPTSP and DFHPCTSP

Note: * These are DFHLIST resource definitions.



B

CICS TS statistics output examples

This appendix provides an example of output from a sample statistics program.

Sample Statistics Program output

Figure B-1 shows Part 1 of the output that is produced by the sample statistics program from a CICS TS partition.

Applid DBDCCICS	Sysid CIC1	Jobname CICSICCF	Date 09/08/16	Time 08:55:19	CICS 02.01.00	PAGE	1
System Status							
VSE Release : VSE/AF9.3.0							
CICS Startup. : WARM							
CICS Status : ACTIVE							
Storage Protection. . . : ACTIVE							
Reentrant Programs. . . : PROTECT							
Monitoring							
Monitoring : OFF							
Exception Class. . . . : OFF							
Performance Class. . . : OFF							
Exception Class Records : 0							
Exception Records Suppressed. . . : 0							
Performance Class Records : 0							
Performance Records Suppressed. . . : 0							
DMF Records : 0							
DMF Errors. : 0							
Statistics							
Statistics End-of-Day Time . . . : 00:00:00							
Statistics Interval. : 03:00:00							
Next Statistics Collection . . . : 00:00:00							
Statistics Recording : OFF							
Applid DBDCCICS	Sysid CIC1	Jobname CICSICCF	Date 09/08/16	Time 08:55:19	CICS 02.01.00	PAGE	2
Transaction Manager							
Total Accumulated transactions so far. . . : 267							
Accumulated transactions (since reset) . . : 267							
Maximum transactions allowed (MXT) : 50							
Times at MXT : 0							
Current Active User transactions : 4							
Peak Active User transactions. : 7							
Total Active User transactions : 244							
Current Running transactions : 1							
Current Dispatchable transactions. : 0							
Current Suspended transactions : 3							
Current System transactions. : 0							
Transactions Delayed by MXT. : 0							
Total MXT queueing time. : 00:00:00.00000							
Average MXT queueing time. : 00:00:00.00000							
Current Queued User transactions : 0							
Peak Queued User transactions. : 0							
Total Queueing time for current queued . . : 00:00:00.00000							
Average Queueing time for current queued : 00:00:00.00000							
Dispatcher							
Dispatcher start time. . . . : 08:21:36.13315							
Peak tasks : 26							
Current tasks. : 13							
Current ICV time : 1,000ms							
Current ICVR time. : 20,000ms							
Current ICVTSD time. . . . : 100ms							
Current PRTYAGING time . . . : 5,000ms							
Accumulated CPU Time : 00:00:00.64345 (Not Reset)							
Number of active CICS TCBs : 5							
TCB Name	TCB Status	TCB Start Time	Op. System Waits	Op. System Wait Time	TCB Dispatch Time		
QR_SUBD	Active	08:21:36.13315	3,794	00:33:42.60395	00:00:00.52446	00:00:00.00000	00:00:00.00000
RO_SUBD	Active	08:21:36.17465	44	00:33:41.07790	00:00:02.00901	00:00:00.00000	00:00:00.00000
SO_MODE	Active	08:21:37.05890	2	00:33:42.20268	00:00:00.00000	00:00:00.00000	00:00:00.00000
SL_MODE	Active	08:21:37.05884	4	08:21:37.09683	00:00:00.03798	00:00:00.00000	00:00:00.00000
SSL_MODE	Active	08:21:37.53139	1	00:33:41.73019	00:00:00.00000	00:00:00.00000	00:00:00.00000
Totals				00:00:02.57147	00:00:00.00000	00:00:00.00000	

Figure B-1 STAT Transaction output (Part 1 of 7)

Figure B-2 shows Part 2 of the output that is produced by the sample statistics program from a CICS TS partition.

Applid DBDCCICS	Sysid CIC1	Jobname CICSICCF	Date 09/08/16	Time 08:55:19	CICS 02.01.00	PAGE 3
Partition size established from ALLOC parameter . . . : 262,144K						
Storage BELOW 16MB						
Partition GETVIS area size under 16 Mb : 11,260K						
Partition GETVIS used area below 16 Mb : 8,752K						
Partition GETVIS free area below 16 Mb : 2,508K						
Partition GETVIS maximum used below 16 Mb : 11,260K						
Partition GETVIS largest free area below 16 Mb . . : 2,472K						
Current DSA Limit : 5,120K						
Current Allocation for DSAs . . : 1,536K						
Peak Allocation for DSAs . . . : 1,536K						
	CDSA	UDSA	SDSA	RDSA	Totals	
Current DSA Size :	512K	256K	256K	512K	1,536K	
Current DSA Used :	372K	12K	192K	272K	848K	
Current DSA Used as % of DSA . :	72%	4%	75%	53%	55% of DSA Size	
* Peak DSA Used :	380K	40K	208K	272K		
Peak DSA Size :	512K	256K	256K	512K		
Cushion Size :	64K	64K	64K	64K		
Free Storage (inc. Cushion) . . :	140K	244K	64K	240K		
* Peak Free Storage :	316K	256K	256K	280K		
* Lowest Free Storage :	132K	216K	48K	240K		
Largest Free Area :	128K	236K	64K	216K		
Largest Free Area as % of DSA . :	25%	92%	25%	42%		
Largest Free/Free Storage . . :	0.91	0.96	1.00	0.90		
Current number of extents . . . :	2	1	1	2	6	
Number of extents added . . . :	2	1	1	2		
Number of extents released . . :	0	0	0	0		
Getmain Requests :	1,538	989	65	19		
Freemain Requests :	1,414	986	36	0		
Current number of Subpools . . :	36	11	5	4	56	
Add Subpool Requests :	294	269	5	4		
Delete Subpool Requests . . . :	258	258	0	0		
Times no storage returned . . . :	0	0	0	0		
Times request suspended . . . :	0	0	0	0		
Current requests suspended . . :	0	0	0	0		
Peak requests suspended . . . :	0	0	0	0		
Requests purged while waiting :	0	0	0	0		
Times Cushion released :	0	0	0	0		
Times Short-On-Storage :	0	0	0	0		
Total time Short-On-Storage . . :	00:00:00.00000	00:00:00.00000	00:00:00.00000	00:00:00.00000		
Average Short-On-Storage time . :	00:00:00.00000	00:00:00.00000	00:00:00.00000	00:00:00.00000		
Storage Violations :	0	0	0	0	0	
Access :	CICS	USER	USER	READONLY		
'*' indicates values reset on last DSA Size change						

Figure B-2 STAT transaction output (Part 2 of 7)

Figure B-3 shows Part 3 of the output that is produced by the sample statistics program from a CICS TS partition.

Applid DBDCCICS	Sysid CIC1	Jobname CICSICCF	Date 09/08/16	Time 08:55:19	CICS 02.01.00	PAGE 4
Storage ABOVE 16MB						
Partition GETVIS area size above 16 Mb	262,136K					
Partition GETVIS used area above 16 Mb	222,604K					
Partition GETVIS free area above 16Mb	39,532K					
Partition GETVIS maximum used above 16 Mb	225,136K					
Partition GETVIS largest free area above 16 Mb	39,484K					
Current EDSA Limit.	204,800K					
CICS Trace table size	4,096K					
Current Allocation for EDSAs.	14,336K					
Peak Allocation for EDSAs	14,336K					
	ECDSA	EUDSA	ESDSA	ERDSA	Totals	
Current DSA Size.	5,120K	1,024K	1,024K	7,168K	14,336K	
Current DSA Used.	4,100K	64K	108K	6,936K	11,208K	
Current DSA Used as % of DSA.	80%	6%	10%	96%	78% of EDSA Size	
* Peak DSA Used	4,104K	128K	108K	6,936K		
Peak DSA Size	5,120K	1,024K	1,024K	7,168K		
Cushion Size.	128K	0K	128K	256K		
Free Storage (inc. Cushion)	1,020K	960K	916K	232K		
* Peak Free Storage	1,056K	1,024K	1,024K	2,340K		
* Lowest Free Storage	1,016K	896K	916K	232K		
Largest Free Area	984K	960K	916K	168K		
Largest Free Area as % of DSA	19%	93%	89%	2%		
Largest Free/Free Storage	0.96	1.00	1.00	0.72		
Current number of extents	5	1	1	5	12	
Number of extents added	5	1	1	5		
Number of extents released.	0	0	0	0		
Getmain Requests.	14,524	4	49	243		
Freemain Requests	5,956	3	1	3		
Current number of Subpools.	184	11	6	4	205	
Add Subpool Requests.	442	269	6	4		
Delete Subpool Requests	258	258	0	0		
Times no storage returned	0	0	0	0		
Times request suspended	0	0	0	0		
Current requests suspended.	0	0	0	0		
Peak requests suspended	0	0	0	0		
Requests purged while waiting	0	0	0	0		
Times Cushion released.	0	0	0	0		
Times Short-On-Storage.	0	0	0	0		
Total time Short-On-Storage	00:00:00.00000	00:00:00.00000	00:00:00.00000	00:00:00.00000		
Average Short-On-Storage time	00:00:00.00000	00:00:00.00000	00:00:00.00000	00:00:00.00000		
Storage Violations.	0	0	0	0	0	
Access.	CICS	USER	USER	READONLY		
'*' indicates values reset on last DSA Size change						

Figure B-3 STAT transaction output (Part 3 of 7)

Figure B-4 shows Part 4 of the output that is produced by the sample statistics program from a CICS TS partition.

Applid DBDCCICS	Sysid CIC1	Jobname CICSICCF	Date 09/08/16	Time 08:55:19	CICS 02.01.00	PAGE	5
Loader							
Library Load requests.	357	Total Program Uses				1,280	
Total Library Load time.	00:00:00.53718	Program Use to Load Ratio.				3.58	
Average Library Load time.	00:00:00.00150						
Library Load requests that waited.	2						
Total Library Load request wait time	00:00:00.00168						
Average Library Load request wait time	00:00:00.00083						
Current Waiting Library Load requests.	0						
Peak Waiting Library Load requests	1						
Times at Peak.	2	Average Not-In-Use program size.				9K	
CDSA		ECDSA					
Programs Removed by compression.	0	Programs Removed by compression.				0	
Time on the Not-In-Use Queue	00:00:00.00000	Time on the Not-In-Use Queue				00:00:00.00000	
Average Time on the Not-In-Use Queue	00:00:00.00000	Average Time on the Not-In-Use Queue				00:00:00.00000	
Programs Reclaimed from the Not-In-Use Queue . .	29	Programs Reclaimed from the Not-In-Use Queue . .				126	
Programs Loaded - now on the Not-In-Use Queue. .	7	Programs Loaded - now on the Not-In-Use Queue. .				14	
SDSA		ESDSA					
Programs Removed by compression.	0	Programs Removed by compression.				0	
Time on the Not-In-Use Queue	00:00:00.00000	Time on the Not-In-Use Queue				00:00:00.00000	
Average Time on the Not-In-Use Queue	00:00:00.00000	Average Time on the Not-In-Use Queue				00:00:00.00000	
Programs Reclaimed from the Not-In-Use Queue . .	251	Programs Reclaimed from the Not-In-Use Queue . .				78	
Programs Loaded - now on the Not-In-Use Queue. .	13	Programs Loaded - now on the Not-In-Use Queue. .				7	
RDSA		ERDSA					
Programs Removed by compression.	0	Programs Removed by compression.				0	
Time on the Not-In-Use Queue	00:00:00.00000	Time on the Not-In-Use Queue				00:00:00.00000	
Average Time on the Not-In-Use Queue	00:00:00.00000	Average Time on the Not-In-Use Queue				00:00:00.00000	
Programs Reclaimed from the Not-In-Use Queue . .	0	Programs Reclaimed from the Not-In-Use Queue . .				0	
Programs Loaded - now on the Not-In-Use Queue. .	2	Programs Loaded - now on the Not-In-Use Queue. .				16	
Program Storage							
Nucleus Program Storage (CDSA)	88K	Nucleus Program Storage (ECDSA).				344K	
Program Storage (SDSA)	108K	Program Storage (ESDSA).				92K	
Resident Program Storage (SDSA).	0K	Resident Program Storage (ESDSA)				0K	
Read-Only Nucleus Program Storage (RDSA)	8K	Read-Only Nucleus Program Storage (ERDSA). . . .				944K	
Read-Only Program Storage (RDSA)	40K	Read-Only Program Storage (ERDSA).				388K	
Read-Only Resident Program Storage (RDSA). . . .	0K	Read-Only Resident Program Storage (ERDSA) . . .				0K	
CDSA used by Not-In-Use programs. :	70K	13.61% of CDSA	ECDSA used by Not-In-Use programs :	204K	3.98% of ECDSA		
SDSA used by Not-In-Use programs. :	68K	26.72% of SDSA	ESDSA used by Not-In-Use programs :	81K	7.87% of ESDSA		
RDSA used by Not-In-Use programs. :	2K	0.32% of RDSA	ERDSA used by Not-In-Use programs :	108K	1.51% of ERDSA		
Applid DBDCCICS	Sysid CIC1	Jobname CICSICCF	Date 09/08/16	Time 08:55:19	CICS 02.01.00	PAGE	6

Figure B-4 STAT transaction output (Part 4 of 7)

Figure B-5 shows Part 5 of the output that is produced by the sample statistics program from a CICS TS partition.

Applid DBDCCICS	Sysid CIC1	Jobname CICSICCF	Date 09/08/16	Time 08:55:19	CICS 02.01.00	PAGE	13			
Programs										
Program Name	Data Loc	Exec Key	Times Used	Times Fetched	Total Fetch Time	Average Fetch Time	Times Newcopy	Times Removed	Program Size	Program Location
\$EDCTCPM	Below	USER	0				0	0		None
\$EDCTCPV	Below	USER	0				0	0		None
ARXITCPU	Below	USER	0				0	0		None
BSTADMII	Below	CICS	0				0	0		None
CEEBINT	Below	USER	1	1	00:00:00.00020	00:00:00.00020	0	0	8	ESDSA
..										
STOCKOR	Any	USER	0				0	0		None
STOCKPT	Below	USER	0				0	0		None
STOCKQU	Below	USER	0				0	0		None
STOCKRF	Below	USER	0				0	0		None
TELNET01	Below	USER	0				0	0		None
Totals			1,000	103						
Applid DBDCCICS	Sysid CIC1	Jobname CICSICCF	Date 09/08/16	Time 08:55:19	CICS 02.01.00	PAGE	44			
Program Totals										
Programs			1,555							
Assembler			1,460							
C			44							
COBOL			13							
LE/VSE			0							
PL1			5							
Other			33							
Maps			70							
Partitionsets			1							
Total			1,626							
CDSA Programs			8							
SDSA Programs			20							
RDSA Programs			3							
ECDSA Programs			24							
ESDSA Programs			8							
ERDSA Programs			37							
SVA Programs			2							
ESva Programs			25							
Unused Programs			11							
Not Located Programs			1,515							
Total			1,626							

Figure B-5 STAT transaction output (Part 5 of 7)

Figure B-6 shows Part 6 of the output that is produced by the sample statistics program from a CICS TS partition.

Applid DBDCCICS	Sysid CIC1	Jobname CICSICCF	Date 09/08/16	Time 08:55:19	CICS 02.01.00	PAGE	45
Temporary Storage							
Put/Putq main storage requests	265						
Get/Getq main storage requests	41						
Peak storage used for TS Main	3K						
Current storage used for TS Main	0K						
Put/Putq auxiliary storage requests . . .	125						
Get/Getq auxiliary storage requests . . .	104						
Times temporary storage queue created . .	276						
Peak temporary storage queues in use . .	102						
Current temporary storage queues in use .	4						
Items in longest queue	11						
Queue extension threshold	20						
Queue extensions created	0						
Control interval size	16,384						
Control intervals in the DFHTEMP dataset :	135						
Peak control intervals used	2						
Current control intervals in use	2						
Available bytes per control interval . .	16,320						
Segments per control interval	255						
Bytes per segment	64						
Writes bigger than control interval size :	0						
Largest record length written	2,762						
Times auxiliary storage exhausted	0						
Number Temporary storage compressions .	3						
Temporary storage strings	8						
Peak Temporary storage strings in use . .	0						
Temporary storage string waits	0						
Peak users waiting on string	0						
Current users waiting on string	0						
Temporary storage buffers	8						
Temporary storage buffer waits	0						
Peak users waiting on buffer	0						
Current users waiting on buffer	0						
Temporary storage buffer reads	0						
Temporary storage buffer writes	0						
Forced buffer writes for recovery	0						
Format writes	0						
I/O errors on the DFHTEMP dataset	0						
Applid DBDCCICS	Sysid CIC1	Jobname CICSICCF	Date 09/08/16	Time 08:55:19	CICS 02.01.00	PAGE	46
Transient Data							
Transient data reads	0						
Transient data writes	0						
Transient data formatting writes	0						
Control interval size	4,096						
Control intervals in the DFHNTRA dataset :	180						
Peak control intervals used	2						
Times NOSPACE on DFHNTRA occurred . . .	0						
Transient data strings	3						
Times Transient data strings in use . . .	0						
Peak Transient data strings in use	0						
Times string wait occurred	0						
Peak users waiting on string	0						
Transient data buffers	3						
Times Transient data buffer in use	26						
Peak Transient data buffers in use	1						
Peak buffers containing valid data	1						
Times buffer wait occurred	0						
Peak users waiting on buffer	0						
I/O errors on the DFHNTRA dataset	0						

Figure B-6 STAT transaction output (Part 6 of 7)

Figure B-7 shows Part 7 of the output that is produced by the sample statistics program from a CICS TS partition.

Applid DBDCCICS	Sysid CIC1	Jobname CICSICCF	Date 09/08/16	Time 08:55:19	CICS 02.01.00	PAGE	47						
LSR Pools													
Pool Number : 1 Time Created : 08:21:39.01568													
Maximum key length : 255													
Total number of strings : 17													
Peak concurrently active strings : 1													
Total requests waited for string : 0													
Peak requests waited for string. : 0													
Buffer Totals													
Data Buffers : 52													
Successful look asides : 420													
Buffer reads : 267													
User initiated writes. : 2													
Non-user initiated writes. . . : 0													
Index Buffers. : 0													
Successful look asides : 0													
Buffer reads : 0													
User initiated writes. : 0													
Non-user initiated writes. . . : 0													
Data and Index Buffer Statistics													
Size Buffers Look Asides Reads User Writes Writes													
512 8 0 0 0 0													
1024 4 409 3 0 0													
2048 6 0 0 0 0													
4096 28 11 264 2 0													
8192 3 0 0 0 0													
12288 3 0 0 0 0													
Applid DBDCCICS	Sysid CIC1	Jobname CICSICCF	Date 09/08/16	Time 08:55:19	CICS 02.01.00	PAGE	48						
Files													
Filename	Access Method	Type	LSR Pool	Str Max	Waits Total	Read Requests	Get Update Requests	Browse Requests	Add Requests	Update Requests	Delete Requests	Data EXCPs	Index EXCPs
BSTCNTL	VSAM		0	0	0	0	0	0	0	0	0	0	0
DFHCSD	VSAM		1	0	0	0	0	0	0	0	0	0	0
EZACACH	VSAM		1	0	0	0	0	0	0	0	0	0	0
EZACONF	VSAM		1	0	0	0	0	0	0	0	0	0	0
IESCNTL	VSAM	KSDS	1	0	0	26	0	0	0	0	0	26	26
IESLDUM	VSAM		1	0	0	0	0	0	0	0	0	0	0
..													
STOPTC4	VSAM		1	0	0	0	0	0	0	0	0	0	0
STOPTC5	VSAM		0	0	0	0	0	0	0	0	0	0	0
Totals						233	4	0					
Applid DBDCCICS	Sysid CIC1	Jobname CICSICCF	Date 09/08/16	Time 08:55:19	CICS 02.01.00	PAGE	49						
Data Tables - Requests													
Filename	Data Table	Type	Max Num recs	Successful Reads	Records Not Found	Adds via Read	Adds via API	Adds Rejected	Rewrite Requests	Delete Requests	Read Retries		

Figure B-7 STAT transaction output (Part 7 of 7)

Abbreviations and acronyms

ICCFTLT

ABEND	abnormal end	DITTO	Data Interfile Transfer, Testing & Operations utility
ACB	access control block	DL/I	Data Language 1
ACF/VTAM	Advanced Communications Facility/Virtual Telecommunications Access Method	DLBL	Disk Label
ACLR	Access Logging and ReportingDescription4	DM	Domain Manager Domain
ALT	application load table	DMF	Data Management Facility
AMA	Application Migration Aid	DNS	Domain Name Support
AMODE	addressing mode	DOR	Data-owning region
AOR	application-owning region	DOS	Disk operating system
AP	Application Domain	DS	Dispatcher Domain
API	Application Programming Interface	DSA	Dynamic Storage Area
APPC	Advanced Program-to-Program Communication	DTR	Dynamic Transaction Routing
APPL	application	DU	Dump Domain
APPN	Advanced Peer-to-Peer Networking	ECB	event control block
AR	Attention Routine	ECI	External Call Interface
ASI	Automatic System Initialization	EDSA	Extended Dynamic Storage
BG	background	EIP	EXEC interface program
BMS	Basic Mapping Support	EOJ	end of job
BSM	Basic Security Manager	ESA	Enterprise Systems Architecture
CDRM	Cross-Domain Resource Manager	ESDS	Entry Sequenced Data Set
CDSA	CICS Dynamic Storage Area	ESM	External Security Manager
CEMT	Master Terminal Transaction	EXCI	external CICS interface
CET	Central European Time	EXEC	Execute/Execution
CI	Control Interval	FCT	File Control Table
CICS	Customer Information Control System	FEPI	Front-End Programming Interface
CICS TS	Customer Information Control System Transaction Server	FOR	file-owning region
CICSVR	CICS VSAM Recovery	FORTTRAN	formula translation
CMF	CICS Monitoring Facility	FSU	Fast Service Upgrade
COBOL	common business-oriented language	FTP	File Transfer Program
CPC	central computer complex	GID	Group Identifier
CSA	Common Storage Area	GL	Global Catalog Domain
CSD	CICS System Definition	GLUE	global user exit
DASD	Direct Access Storage Device	GTF	generalized trace facility
DB2	DataBase2	HLASM	high-level assembler
DCT	Destination Control Table	HLL	high-level language
DD	Directory Manager Domain	HTML	Hypertext Markup Language
		HTTP	Hypertext Transport Protocol
		I/O	Input/Output
		IBM	International Business Machines Corporation

ICCF	Interactive Computing and Control Facility	PSP	preventive service planning
ID	identification/identifier	PTF	Program Temporary Fix
II	Interactive Interface	QOR	Queue-owning region
IPL	initial program load	IBM RACF®	Resource Access Control Facility
IRC	interregion communication	RC	return code
ISC	intersystem communication	RCF	Report Controller Facility
IT	Information Technology	RDO	Resource Definition Online
ITSO	International Technical Support Organization	RDSA	Read-only Dynamic Storage Area
JCT	journal control table	REXX	Restructured Extended Executor
KE	Kernel Domain	RMODE	Residency mode
KSDS	Key Sequenced Data Set	RPG	Report Program Generator
LAN	local area network	SAF	System Authorization Facility
LC	Local Catalog Domain	SAM	Sequential Access Method
LD	Loader Domain	SCSI	Small Computer Systems Interface
LE/VSE	Language Environment for z/VSE	SDT	Shared Data Table
LM	Lock Manager Domain	SIT	System Initialization Table
LPAR	Logically Partitioned mode	SM	Storage Manager Domain
LSR	Local Shared Resources	SMA	Security Migration Aid
LU	Logical Unit	SNA	Systems Network Architecture
ME	Message Domain	SNT	System Signon Table
MN	Monitoring Domain	SOAP	Simple Object Access Protocol
MPS	multiple partition support	SPI	System Programming Interface
MQ	Message Queue	ST	Statistics Domain
MQI	Message Queue Interface	STS	Security Transaction Server
MQ Series	Message Queue Series	SVA	Shared Virtual Area
MRO	multiregion operation	SYSRES	system residence file
N/A	not applicable	TCP/IP	Transmission Control Protocol/Internet Protocol
NLT	nucleus load table	TCT	Terminal Control Table
NSR	nonshared resources	TI	Timer Domain
OCO	object code only	TLT	Terminal List Table
PA	Parameter Manager Domain	TOD	time-of-day
PCT	Program Control Table	TOR	terminal-owning region
PG	Program Manager Domain	TR	Trace Domain
PL/I	Programming Language 1	TRUE	Task-related user exit
PLT	Program List Table	UDSA	User Dynamic Storage Area
PLTPI	Program List Table Post Initialization	URM	User-Replacable Module
PLTSD	Program List Table Shutdown	US	User Domain
POR	printer-owning region	VM	Virtual Machine
POWER	Priority Output Writers, Execution processor, and input Readers	VSAM	Virtual Storage Access Method
PPT	Processing Program Table	VSCR	Virtual Storage Constraint Relief
PSB	program specification block	VSE	Virtual Storage Extended
		VTAM	Virtual Telecommunications Access Method

VTOC	Volume Table of Contents
WWW	World Wide Web
XM	Transaction Manager Domain
XML	extensible Markup Language
XPI	exit programming interface
XRF	extended recovery facility
XS	Security Manager Domain
z/OS	IBM System z® Operating System
z/VM	IBM System z Hypervisor
z/VSE	IBM System z entry operating system

Related publications

The publications that are listed in this section are considered particularly suitable for a more detailed discussion of the topics that are covered in this book.

IBM Redbooks

The following IBM Redbooks publications provide more information about the topic in this book. Some publications that are referenced in this list might be available in softcopy only:

- ▶ *CICS Transaction Server from Start to Finish*, SG24-7952:
<http://www.redbooks.ibm.com/abstracts/sg247952.html>
- ▶ *The Complete Guide to CICS Transaction Gateway Volume 1 Configuration and Administration*, SG24-8160:
<http://www.redbooks.ibm.com/abstracts/sg248160.html>
- ▶ *Application Development for IBM CICS Web Services*, SG24-7126:
<http://www.redbooks.ibm.com/abstracts/sg247126.html>

You can search for, view, download, or order these documents and other Redbooks, Redpapers, Web Docs, draft and more materials, at the following web address:

ibm.com/redbooks

Other publications

For more information about IBM CICS TS for VSE/ESA and REXX see this website:

<https://ibm.biz/BdiYJd>

The following publications that are available at this site were referenced in this IBM Redbooks publication:

- ▶ *CICS TS for VSE/ESA Release Guide*, GC33-1645
- ▶ *CICS TS for VSE/ESA Migration Guide*, GC33-1646
- ▶ *CICS TS for VSE/ESA System Definition Guide*, SC33-1651
- ▶ *CICS TS for VSE/ESA Customization Guide*, SC33-1652
- ▶ *CICS TS for VSE/ESA Resource Definition Guide*, SC33-1653
- ▶ *CICS TS for VSE/ESA Operations and Utilities Guide*, SC33-1654
- ▶ *CICS TS for VSE/ESA Application Programming Reference*, SC33-1658
- ▶ *CICS TS for VSE/ESA Application Migration Aid Guide*, SC33-1943
- ▶ *CICS TS for VSE/ESA Performance Guide*, SC33-1667
- ▶ *CICS TS for VSE/ESA Shared Data Tables Guide*, SC33-1668
- ▶ *CICS TS for VSE/ESA Security Guide*, SC33-1942
- ▶ *CICS TS for VSE/ESA Report Controller Planning Guide*, GC33-1941
- ▶ *CICS TS for VSE/ESA Trace Entries Handbook*, SX33-6108

For more information about z/VSE, see *z/VSE V6R1 PDF Extended Shelf*, which is available at this website:

<http://ibm.co/2n3KJBq>

The following publications that are available at this site were referenced in this IBM Redbooks publication:

- ▶ *z/VSE V6R1.0.0 Planning*, SC34-2681
- ▶ *z/VSE V6R1.0.0 Installation*, SC34-2678
- ▶ *z/VSE V6R1.0.0 Administration*, SC34-2692
- ▶ *z/VSE V4R3.0.0 Operation*, SC33-8309
- ▶ *z/VSE V6R1.0.0 System Control Statements*, SC34-2679
- ▶ *z/VSE V4R3.0 System Utilities*, SC33-8336
- ▶ *z/VSE V5R1.0.0 e-business Connectors User's Guide*, SC34-2629
- ▶ *CICS TS for z/VSE Enhancement Guide*, SC34-2685

For more information, see the following resources:

- ▶ *IBM Data Language/I Virtual Storage Extended Release Guide Version 1 Release 12*, SC33-6211:
<http://ibm.co/2n3Sw1Y>
- ▶ *DB2 Server for VSE System Administration V7R5*, SC09-2981:
<http://ibm.co/2m6x10h>

Online resources

The following web sites provide relevant information sources:

- ▶ *CICS Developer Center* on IBM developerWorks, including blogs, videos, podcasts, samples, and support:
<https://developer.ibm.com/cics/>
- ▶ IBM Knowledge Center for *CICS Transaction Server for z/VSE*:
<https://www.ibm.com/support/knowledgecenter/SSE CAB>
- ▶ *IBM z/VSE* home page:
<http://www.ibm.com/systems/z/os/zvse/>

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Redbooks

Migration to IBM CICS Transaction Server for z/VSE

SG24-8390-00

ISBN 0738442461



(0.5" spine)

0.475" <-> 0.873"

250 <-> 459 pages



SG24-8390-00

ISBN 0738442461

Printed in U.S.A.

Get connected

