



CICS Automation Programmer's Reference and Operator's Guide



CICS Automation Programmer's Reference and Operator's Guide

Note!

Before using this information and the product it supports, read the information in "Notices" on page ix.

This edition applies to IBM Tivoli System Automation for z/OS (5698-SA3) Version 3 Release 2, an IBM licensed program, and to all subsequent releases and modifications until otherwise indicated in new editions.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates.

A form for readers' comments appears at the back of this publication. If the form has been removed, address your comments to:

IBM Deutschland Entwicklung GmbH
Department 3248
Schoenaicher Strasse 220
D-71032 Boeblingen
Federal Republic of Germany

FAX: (Germany) 07031-16-3456
FAX: (Other countries) (+49)+7031-16-3456

Internet: s390id@de.ibm.com

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 1990, 2007. All rights reserved.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Figures	v
--------------------------	----------

Tables	vii
-------------------------	------------

Notices	ix
--------------------------	-----------

Programming Interface Information.	x
Trademarks	xi

Accessibility	xiii
--------------------------------	-------------

Using assistive technologies	xiii
Keyboard navigation of the user interface	xiii
z/OS information	xiii

About This Book	xv
----------------------------------	-----------

Who Should Use This Book	xv
What's in This Book	xv
Related Publications	xv
The System Automation for z/OS Library	xv
Related Product Information	xvi
Using LookAt to look up message explanations	xvi
Summary of Changes for SC33-8267-04.	xvii
Changed Information	xvii
Moved Information	xvii
Deleted Information	xvii

Part 1. Introducing CICS Automation 1

Chapter 1. Functions of CICS Automation 3

Link Monitoring	3
Health Checking	3
Recovery	3
Program-to-Program Interface	3
NetView Components	4
CICS Components	4
Communication Components	4
CICS Message Processing	4

Part 2. Customizing CICS Automation. 7

Chapter 2. Customizing CICS Automation 9

*CICS Best Practices Policy	9
Customizing CICS Definitions	9
Step 1: Basic CICS Automation Common Policy Definitions	9
Step 2: Basic CICS Application Definitions	9
Step 3: Installing CICSplex SM REXX API	10
Step 4: Health Check Programs	11
CICS Automation Definitions for CICSplex System Manager	11

Automating Coordinating Address Space Startup and Shutdown	11
Automating CICSplex SM Address Space Startup and Shutdown	11
Using the CICS Automation Message Exit	12
Defining CICS Messages	12
Refreshing Policy Data	16
Partial Message IDs and Performance.	16
Migration and Coexistence	17
Migration	17
Coexistence	17

Chapter 3. How to Set Up the Functions of CICS Automation 19

Automating Recovery For Transactions	19
How to Define Transaction Recovery	20
How to Set Up Health Checking	21
How to Set Up Link Monitoring	22
Adding Local Applications to the CICS Automation Operator Interface	22
Using Linemode Functions	24
Health Checking.	24
System Init Table Override	24
Message Options	24
CEMT PPI	25
How to Implement Remote Site Recovery for VSAM RLS (CICS TS Function Only)	25

Chapter 4. MESSAGES/USER DATA Entries for CICS Automation 27

CICS-Specific MESSAGES/USER DATA Keywords	27
ABCODESYSTEM: System Abend Recovery	28
ABCODETRAN: Transaction Abend Recovery	29
CICSINFO: Display Information	31
HEALTHCHK: Health Checking	32
LISTSHUT: Transaction Purging During Shutdown	33
RCVRSOS: Short-On-Storage Handling	35
RCVRTRAN: Transaction Recovery	36

Chapter 5. CICS Automation Routines and Commands 37

Operator Commands	37
CEMT PPI: CEMT PPI Short Syntax	38
CICSHLTH: Linemode Health Checking.	39
CICSOVRD: Linemode SIT Override	40
Automation Policy Commands	41
CICSPURG: Purge Transactions.	42
CICRSYC: CICS Resync	43
CICSSHUT: Shutdown Processor	44
EVEERDMP: CICS Dump	45
CMASSHUT: CICSplex SM Address Space Shutdown	46
Application Programming Interfaces	47

CICSQRY: Name Lookup	48
CICSRCMD: Request a CICS Function	52

Part 3. Using CICS Automation 53

Chapter 6. Working with CICS

Resources 55

Using CICS Automation Panels	55
Using the Main Menu	55
Starting and Stopping Resources	56
Startup	56
Shutdown	59
Monitoring Your CICS Subsystems	59
Link Monitoring	59
Health Checking	62
Broadcasting Messages	63

Chapter 7. The Status Display Facility 65

Appendix. CICS Automation and the Program-to-Program Interface. 67

Program-to-Program Interface Components in NetView and CICS	67
NetView Requests Using the Program-to-Program Interface	68
CONVERSE from NetView	68
SEND from NetView	70

CANCEL from NetView	71
CICS Requests Using the Program-to-Program Interface	72
CONVERSE from CICS	72
SEND from CICS	73
Programming Interface	74
EVESNCCI: NetView to CICS Communication Interface	76
EVESNRSP: Common Response Handler from CICS	81
EVESCCCI: CICS to NetView Communication Interface	82
EVEMPINT: EVESCCCI Parameter List Copy Book	85
Customizing CICS Definitions	90
Step 1: Modifications to Program-to-Program Interface Initialization	90
Step 2: Define a NetView PPI Receiver Task	90
Step 3: Define a CICS PPI Receiver Task.	90
Definition Members	90
EVESPINM: CICS PPI Initialization Member	91
EVENTASK: NetView PPI Initialization Member	92
Security Checking Using CICS	93

Glossary of CICS and Other Terms 95

Index 99

Figures

1.	Thresholds Definitions Panel	20	8.	Program-to-Program Interface Components in NetView and CICS	68
2.	Code Processing Panel	21	9.	An EVESNCCI CONVERSE Request	70
3.	CICS Automation Main Menu	55	10.	An EVESNCCI SEND Request	71
4.	Verification Panel for INGREQ	58	11.	An EVESCCCI CONVERSE Request	73
5.	Event-based Monitoring Infrastructure	60	12.	An EVESCCCI SEND Request	74
6.	Health Checking Panel.	62			
7.	Broadcast Messages Panel.	63			

Tables

1. System Automation for z/OS Library xv	6. EVESCCCI Fields Returned to Caller from CONVERSE Request 83
2. Keywords in CICS Message Definitions 12	7. EVESCCCI SEND Request Parameter List 84
3. CICSQRY Return Codes 49	
4. CICSRCMD Return Codes 52	
5. EVESCCCI CONVERSE Request Parameter List 82	

Notices

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

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

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
USA

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Deutschland Entwicklung GmbH
Department 3248
Schoenaicher Strasse 220
D-71032 Boeblingen
Federal Republic of Germany

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

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

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

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

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Programming Interface Information

This book documents programming interfaces that allow the customer to write programs to obtain the services of IBM Tivoli System Automation for z/OS.

Trademarks

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

CICS	MVS/ESA	SP
CICSplex	NetView	Tivoli
DB2	OS/390	VTAM
IBM	RACF	z/OS
IMS	Resource Link	
MVS	S/390	

Accessibility

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully. The major accessibility features in z/OS™ enable users to:

- Use assistive technologies such as screen readers and screen magnifier software
- Operate specific or equivalent features using only the keyboard
- Customize display attributes such as color, contrast, and font size

Using assistive technologies

Assistive technology products, such as screen readers, function with the user interfaces found in z/OS. Consult the assistive technology documentation for specific information when using such products to access z/OS interfaces.

Keyboard navigation of the user interface

Users can access z/OS user interfaces using TSO/E or ISPF. Refer to *z/OS TSO/E Primer*, *z/OS TSO/E User's Guide*, and *z/OS ISPF User's Guide Vol I* for information about accessing TSO/E and ISPF interfaces. These guides describe how to use TSO/E and ISPF, including the use of keyboard shortcuts or function keys (PF keys). Each guide includes the default settings for the PF keys and explains how to modify their functions.

z/OS information

z/OS information is accessible using screen readers with the BookServer/Library Server versions of z/OS books in the Internet library at:

<http://www.ibm.com/servers/eserver/zseries/zos/bkserv/>

About This Book

This book describes how to customize and operate CICS® Automation. CICS Automation provides a simple and consistent way to monitor and control all of the CICS regions, both local and remote, within your organization. This automates, simplifies, and standardizes console operations and the management of component, application, and production related tasks.

Who Should Use This Book

This book is intended for the following users:

- System programmers, system designers, and application designers who will automate CICS using CICS Automation.

For these users, all three parts of the book are of interest.

Installing and customizing CICS Automation requires a programmer's understanding of NetView®, CICS, SA z/OS, and CICS Automation, because most of the definitions are done in these programs. Also, you will modify JCL, command lists, and programs for some of the automation functions.

- Operators and administrators who manage and monitor CICS subsystems.

These users mainly need part 1 and part 3.

For operators, a working knowledge of CICS will be assumed.

What's in This Book

This book contains the following:

Part 1, "Introducing CICS Automation"

Explains the concepts of SA z/OS and describes the functions of CICS Automation.

Part 2, "Customizing CICS Automation"

Describes the customization of CICS Automation and contains reference sections for MESSAGES policy items and for the programming interface.

Part 3, "Using CICS Automation"

Describes the operator interface of CICS Automation.

Related Publications

The System Automation for z/OS Library

The following table shows the information units in the System Automation for z/OS library:

Table 1. System Automation for z/OS Library

Title	Order Number
<i>IBM Tivoli System Automation for z/OS Planning and Installation</i>	SC33-8261
<i>IBM Tivoli System Automation for z/OS Customizing and Programming</i>	SC33-8260
<i>IBM Tivoli System Automation for z/OS Defining Automation Policy</i>	SC33-8262
<i>IBM Tivoli System Automation for z/OS User's Guide</i>	SC33-8263

Table 1. System Automation for z/OS Library (continued)

Title	Order Number
<i>IBM Tivoli System Automation for z/OS Messages and Codes</i>	SC33-8264
<i>IBM Tivoli System Automation for z/OS Operator's Commands</i>	SC33-8265
<i>IBM Tivoli System Automation for z/OS Programmer's Reference</i>	SC33-8266
<i>IBM Tivoli System Automation for z/OS CICS Automation Programmer's Reference and Operator's Guide</i>	SC33-8267
<i>IBM Tivoli System Automation for z/OS IMS Automation Programmer's Reference and Operator's Guide</i>	SC33-8268
<i>IBM Tivoli System Automation for z/OS TWS Automation Programmer's Reference and Operator's Guide</i>	SC23-8269
<i>IBM Tivoli System Automation for z/OS End-to-End Automation Adapter</i>	SC33-8271
<i>IBM Tivoli System Automation for z/OS Monitoring Agent Configuration and User's Guide</i>	SC33-8337

The System Automation for z/OS books are also available on CD-ROM as part of the following collection kit:

IBM® Online Library z/OS Software Products Collection (SK3T-4270)

SA z/OS Home Page

For the latest news on SA z/OS, visit the SA z/OS home page at <http://www.ibm.com/servers/eserver/zseries/software/sa>

Related Product Information

You can find books in related product libraries that may be useful for support of the SA z/OS base program by visiting the z/OS Internet Library at <http://www.ibm.com/servers/eserver/zseries/zos/bkserv/>

Using LookAt to look up message explanations

LookAt is an online facility that lets you look up explanations for most of the IBM messages you encounter, as well as for some system abends and codes. Using LookAt to find information is faster than a conventional search because in most cases LookAt goes directly to the message explanation.

You can use LookAt from these locations to find IBM message explanations for z/OS elements and features, z/VM®, VSE/ESA™, and Clusters for AIX® and Linux™:

- The Internet. You can access IBM message explanations directly from the LookAt Web site at <http://www.ibm.com/servers/eserver/zseries/zos/bkserv/lookat/>.
- Your z/OS TSO/E host system. You can install code on your z/OS or z/OS.e systems to access IBM message explanations using LookAt from a TSO/E command line (for example: TSO/E prompt, ISPF, or z/OS UNIX® System Services).
- Your Microsoft® Windows® workstation. You can install LookAt directly from the *z/OS Collection* (SK3T-4269) or the *z/OS and Software Products DVD Collection* (SK3T4271) and use it from the resulting Windows graphical user interface

(GUI). The command prompt (also known as the DOS > command line) version can still be used from the directory in which you install the Windows version of LookAt.

- Your wireless handheld device. You can use the LookAt Mobile Edition from <http://www.ibm.com/servers/eserver/zseries/zos/bkserv/lookat/lookatm.html> with a handheld device that has wireless access and an Internet browser (for example: Internet Explorer for Pocket PCs, Blazer or Eudora for Palm OS, or Opera for Linux handheld devices).

You can obtain code to install LookAt on your host system or Microsoft Windows workstation from:

- A CD-ROM in the *z/OS Collection* (SK3T-4269).
- The *z/OS and Software Products DVD Collection* (SK3T4271).
- The LookAt Web site (click **Download** and then select the platform, release, collection, and location that suit your needs). More information is available in the LOOKAT.ME files available during the download process.

Summary of Changes for SC33-8267-04

This document contains information previously presented in System Automation for z/OS V3R1.0 CICS Automation Programmer's Reference and Operator's Guide, SC33-8267-03, which supports z/OS Version 1 Release 7.

Changed Information

The following sections have been changed:

- “*CICS Best Practices Policy” on page 9.
- “Step 1: Basic CICS Automation Common Policy Definitions” on page 9 no longer contains a substep to make definitions for CICS state/action tables, and the steps for PPI definitions have been moved to “CICS Automation and the Program-to-Program Interface,” on page 67.
- “Defining CICS Messages” on page 12.
- “Migration and Coexistence” on page 17.
- Due to changes in link monitoring, the following have been updated:
 - “Link Monitoring” on page 3
 - “How to Set Up Link Monitoring” on page 22
 - “Link Monitoring” on page 59

Moved Information

The following sections have been moved to “CICS Automation and the Program-to-Program Interface,” on page 67:

- “Customizing CICS Definitions” on page 90
- “Definition Members” on page 90
- “Security Checking Using CICS” on page 93

Deleted Information

The following sections have been deleted:

- “State/Action Tables” in Chapter 1, “Functions of CICS Automation,” on page 3
- “How to Set Up the State/Action Tables” in Chapter 3, “How to Set Up the Functions of CICS Automation,” on page 19

- “RCVRSOS—Short-On-Storage Handling” in Chapter 4, “MESSAGES/USER DATA Entries for CICS Automation,” on page 27 and references to this keyword
- “EVEEY00S—Common State Handler for State/Action Tables” in Chapter 5, “CICS Automation Routines and Commands,” on page 37 and references to it
- The chapter “NMC Display Support”

You may notice changes in the style and structure of some content in this document—for example, headings that use uppercase for the first letter of initial words only, and procedures that have a different look and format. The changes are ongoing improvements to the consistency and retrievability of information in our documents.

This document contains terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

Part 1. Introducing CICS Automation

This part describes the concepts of SA z/OS, including some NetView-related information, and gives an overview of the functions provided by CICS Automation. It consists of the following chapter:

- Chapter 1, “Functions of CICS Automation,” on page 3

Chapter 1. Functions of CICS Automation

CICS Automation is integrated into SA z/OS. Thus, CICS regions must be generated in the policy database as subsystems by linking CICS applications to systems, in order to be available to CICS Automation. Triggers and service periods for CICS regions are also defined as for any other application.

Link Monitoring

CICS communicates with other systems or subsystems using either intersystem communication (ISC) or interregion communication (IRC). Link monitoring verifies that the IRCs and ISCs are active. This is done using SA z/OS monitor resources that are defined in the customization dialog. These monitor resources register for events that are issued by CICSplex[®] System Manager (CICSplex SM) whenever there is a problem with the monitored link. When an event is received, INGMON is called to map the event to a health status, update the monitor resource and trigger recovery action, if it has been defined in the automation policy.

Health Checking

Through health checking, you execute programs that check the state of certain components of an application running under CICS. CICS Automation initiates a health check program by sending a request across the program-to-program interface to CICS. CICS invokes the program, and sends back an Acknowledgement (ACK), indicating that the program executed as expected; or it sends a Negative Acknowledgement (NACK), indicating that the program did not execute as expected. A NACK includes data describing the error. Up to ten health check programs can be associated with any CICS application that is defined to SA z/OS.

Health check routines are usually executed automatically at timed intervals. When CICS Automation receives an abnormal response from CICS (NACK) or when a timeout occurs, CICS Automation sends out notification messages.

Note that, if you intend to perform health checking, you will need to use the NetView program-to-program interface.

Recovery

You can automate recovery for transactions and for certain problems. Transaction recovery can be automated globally and for individual transactions. This is achieved by combining some of the basic functions of the product with CICS-specific policy items and several CICS-specific message IDs.

Program-to-Program Interface

Note that you only need to use the NetView program-to-program interface for health checking and link monitoring as implemented in previous releases.

NetView's program-to-program interface provides the ability to communicate between a NetView application and other address spaces on the same host, such as CICS and IMS[™]. CICS Automation uses this program-to-program interface to:

- Initiate, from NetView, the execution of a CICS program.
- Process a response from this CICS program.
- From CICS initiate, the execution of a command list or command processor in NetView.
- Process a response from this command list or command processor.

There are CICS Automation program-to-program interface components in CICS as well as in NetView. Chapter 2, “Customizing CICS Automation,” on page 9 provides you with step-by-step procedures that tell you how to install these components so that the interface can be implemented.

If you want to use the CICS Automation program-to-program interface code for your own purposes, refer to “CICS Automation and the Program-to-Program Interface,” on page 67 for further information.

NetView Components

There is an optional task (EVENTASK), an initialization member for this optional task, and command processors. The initialization member is described in “EVENTASK: NetView PPI Initialization Member” on page 92. The command processors are described in “EVESNCCI: NetView to CICS Communication Interface” on page 76, and “EVESNRSP: Common Response Handler from CICS” on page 81.

CICS Components

There is a long-running transaction COPC, as well as start and stop transactions to start and stop the CICS program-to-program interface component; there is also a subroutine which is described in “EVESCCCI: CICS to NetView Communication Interface” on page 82, and an initialization member which is described in “EVESPINM: CICS PPI Initialization Member” on page 91.

Communication Components

An ID (RECEIVERID) is defined at both ends of the program-to-program interface, so CICS Automation knows which NetView to sign on to. This RECEIVERID is contained in the initialization members described above. The VTAM[®] applid is used by CICS to sign on to the program-to-program interface. NetView determines which applid relates to which subsystem from the **APPLid** field of the CICS CONTROL policy item.

The CICS Automation program-to-program interface cannot function if the RECEIVERIDs in the initialization members do not match, or if the VTAM applid of the **APPLid** field of the CICS CONTROL policy item does not match the VTAM applid. The customization sections describe how to define the RECEIVERID and the VTAM applid.

CICS Message Processing

SA z/OS can only automate messages that are issued via WTO. Most CICS system messages that are important are WTO'd. However, there are many CICS messages that are only logged to the CICS external log. Some of these messages may be useful in automation situations. To enable SA z/OS to process these messages, exits are installed to WTO messages that would not be WTO'd by CICS.

Note:

Only CICS messages that are *not* WTO'd by CICS are processed.

Messages that would be WTO'd by CICS by default can be specified in the exit policy, however the exit will take no actions for these messages. More specifically, it will not suppress a message that CICS has already determined to send to the console via WTO.

In addition, user code might produce messages that are written to CICS Transient Data Queues. Some of these messages might be of interest in automation situations. SA z/OS installs an exit to WTO these messages.

Note:

To avoid duplication, *no* messages starting with "DFH" are checked in the Transient Data Exit. Such messages are handled by the Message Exit.

Part 2. Customizing CICS Automation

This part describes how to customize and set up CICS Automation. It also contains reference information for CICS-specific MESSAGES keywords and for common routines which request information or perform tasks associated with CICS Automation. It consists of the following chapters:

- Chapter 2, “Customizing CICS Automation,” on page 9
- Chapter 3, “How to Set Up the Functions of CICS Automation,” on page 19
- Chapter 4, “MESSAGES/USER DATA Entries for CICS Automation,” on page 27
- Chapter 5, “CICS Automation Routines and Commands,” on page 37

Chapter 2. Customizing CICS Automation

This chapter explains how to customize NetView, CICS and SA z/OS for CICS Automation. The customization consists of the following steps:

1. Define CICS Automation to CICS.
2. CICS Automation definitions in NetView. In this step you define the policy objects in the SA z/OS policy database that are necessary for CICS Automation.

*CICS Best Practices Policy

For the definitions to be made in NetView, refer to *IBM Tivoli System Automation for z/OS Planning and Installation*.

SA z/OS supplies a sample policy, *CICS, that provides best practice definitions for running CICS Automation. It provides you with a number of sample classes and instances of entry type APL that model CICS applications. Use these samples as a guideline in defining your CICS Automation policies.

For further policy details you can create a new policy database using *EMPTY as the basis and selecting *CICS as the add-on policy, and then create an enterprise report.

Diagrams of the sample policies are provided as PDF files that are located in the USS installation path. The default for this path is: /usr/lpp/ing/doc/policies/.

Customizing CICS Definitions

Step 1: Basic CICS Automation Common Policy Definitions

For each NetView domain, set up the CICS Automation environment according to the following list:

1. Verify that the system environment is defined as described in the SA z/OS documentation.
2. Make sure that the Automation Operators that are used for CICS are defined in the AOP entry type. The following entries (all delivered in the *CICS add-on sample policy database) are required:

Automated Function	Operator ID	Message Classes
CICSMSTR	AUTCICS	EVE*

3. Define link monitoring. See "How to Set Up Link Monitoring" on page 22 for details.

Step 2: Basic CICS Application Definitions

For each CICS application, set up the CICS Automation specifications in the customization dialogs according to the following list:

1. Specify the basic information for the CICS applications (APL entry type). The **Application Type** field of the **Define New Entry** panel must be set to CICS for CICS applications.

*CICS Best Practices Policy

2. Check the CICS CONTROL policy item and enter any required values. You must specify a value for the APPLid field.
3. Specify the automation flags in the AUTOMATION FLAGS policy item.
4. Specify the thresholds of the application in the THRESHOLDS policy item.
5. Specify the startup commands in the STARTUP policy item. Note that the possible startup types depend on the CICS version as follows:

Start Type	Explanation	Valid for
AUTO	Uses the restart data set to determine the startup type.	All releases
COLD	Initiates a cold start.	All releases
INITIAL	Initiates a cold start with additional processing for CICS TS resources.	CICS TS V1R1 and above
NORM	This is the same as AUTO.	
STANDBY	Initiates a start for an XRF backup.	All releases

6. Specify the shutdown commands in the SHUTDOWN policy item.
7. If you want to use service periods or triggers, link these to the application in the SERVICE PERIOD or TRIGGER policy item.
8. If you want to use link monitoring, refer to the description of link monitoring in “How to Set Up Link Monitoring” on page 22.
9. Specify the ACORESTART keyword in the MESSAGES/USER DATA item of the APPLICATION policy object. The command associated with this keyword must be CICSRSYC, see “CICSRSYC: CICS Resync” on page 43.
10. If you want to use the health check function, specify the HEALTHCHK keyword in the MESSAGES/USER DATA policy item. For details, see “HEALTHCHK: Health Checking” on page 32.
11. Review the supplied CICS sample classes and instances for the message keywords described in “CICS-Specific MESSAGES/USER DATA Keywords” on page 27. Customize these entries as required.
12. If you need thresholds for the RCVR SOS or RCVRTRAN (RCVRTRAN.*tranid*) message keywords (see “CICS-Specific MESSAGES/USER DATA Keywords” on page 27 for these keywords), define resource thresholds for them with the names of SOS or TRAN (TRAN.*tranid*) in the application's MINOR RESOURCE THRES policy item. For more details on recovery, see “Automating Recovery For Transactions” on page 19.

Step 3: Installing CICSplex SM REXX API

To manage CICSplex SM CMAS address spaces the CPSM REXX API is required. This can be installed in NetView by adding the CICSplex SM library SEYUAUTH before the library that contains module IRXFLOC. If there are no existing IRXFLOC modules the library can be placed at the end of the //STEPLIB concatenation.

Alternatively, module IRXFLOC can be customized according to the instructions in the section “Installing the REXX function package” in manual *CICS Transaction Server for z/OS Installation Guide*.

The version of SEYUAUTH that is accessible to SA z/OS NetView (via either the NetView STEPLIB or the linklist) must be at the same level as the SEYUAUTH library that is accessed by the CMAS that SA z/OS will connect to.

Step 4: Health Check Programs

Note: This step assumes that the health check routines have already been written and that you know the transaction name, program name, and language of each of them. If you need more information, read “How to Set Up Health Checking” on page 21 before performing this step.

If you want to use the health check feature, define the program name and language for each health check routine (there can be up to ten for each CICS subsystem) as follows:

```
DEFINE PROGRAM(program) LANGUAGE(language)
```

Note: The program name must correspond to a program name defined in the **Data** field in the HEALTHCHK keyword in the MESSAGES/USER DATA policy item message ID for the respective subsystem. For more information on HEALTHCHK, see “HEALTHCHK: Health Checking” on page 32.

CICS Automation Definitions for CICSplex System Manager

This section helps you define CICSplex System Manager components to Automation. For definitions that are required for CICS, see the add-on policy.

Automating Coordinating Address Space Startup and Shutdown

The Coordinating Address Space (CAS) applications are not CICS systems. They should be defined in the policy database with application type STANDARD, not CICS.

Automating CICSplex SM Address Space Startup and Shutdown

The CICSplex SM Address Space (CMAS) regions are CICS regions. They should be defined in the policy database with application type CICS and subtype CMAS. The CMAS regions execute only CICSplex SM code. (CICSplex SM recommends that only CICSplex code be run in CMAS regions. User transactions should not be run in CMAS regions.)

The Automation required to manage the CMAS regions is less than a normal CICS region because there is no need to have PPI and other CICS-monitoring functions. However, CMAS CICS regions need to be shut down via the CICSplex SM SHUTDOWN command. To allow CICS Automation to shut down the CMAS, the COSD command is used. Use it instead of the normal CICS SHUT command in the policy definition for CMAS regions.

Note: The use of CEMT PERFORM SHUTDOWN is not recommended for CMAS regions.

CPSM recommends that CMAS regions be started prior to MAS regions they manage. This can be achieved via relationships as supported by System Automation for z/OS.

Using the CICS Automation Message Exit

To enable the CICS Message and Transient Data Queue exits to process messages, the messages must be defined in the policy database. The MESSAGES/USER DATA policy is used to define the messages. Messages IDs that are added to the MESSAGES/USER DATA policy for CICS Message exit processing must not contain special characters. Each message ID must be a single alphanumeric word.

If there are no messages defined to use the exits, the exits are disabled. The exits are enabled when message or transient data policy is defined.

Message policy data is processed on a subsystem basis, however, you can put the policy information in a CLASS and link the class to the subsystems to save on data entry.

Each subsystem has its own policy and does not share policy information, except for CLASS information, with other subsystems. This means you can update the policy information for a subsystem or set of subsystems and not affect other subsystems.

Defining CICS Messages




CICS messages that are not already WTO'd can be WTO'd by specifying the message in the customization dialog. In addition you can control the transient data queues and routing codes for any messages beginning with DFH.

There are several special keywords that, when added to the message with the USR command, cause the message information to be loaded into the appropriate exit for processing. The keywords and their descriptions are described in Table 2.

Table 2. Keywords in CICS Message Definitions

Keyword	Description
OFFSET	This keyword is required to load the message into the exit. It specifies the word number in the message that represents the message ID. Its format is a single integer number.

Table 2. Keywords in CICS Message Definitions (continued)

Keyword	Description
INSERT	<p>This keyword specifies the matching Insert values for CICS messages. It is optional and allows you to make decisions to control the message based on its content. Its format is:</p> <div style="margin-left: 20px;">  <pre> (Tokens CICS Actions) </pre> </div> <p>Tokens:</p> <div style="margin-left: 20px;">  <pre> (number, word_text) </pre> </div> <p>CICS Actions:</p> <div style="margin-left: 20px;">  <pre> NONE AUTOMATE AUTO SUPPRESS </pre> </div> <p>Where <i>number</i> is the insert number to be checked and <i>word_text</i> is the text to be checked.</p> <p>Inserts are described in <i>CICS Messages and Codes</i> for each of the messages that have them. The <i>word_text</i> specification is checked for the length specified. This allows for trailing data to be ignored. The format of the number is a single integer.</p> <p>The format for <i>word_text</i> is a single alphanumeric word; no blanks or special characters are allowed. The case of the value is ignored.</p> <p>The actions that are available are:</p> <p>NONE No actions are to be taken.</p> <p>AUTOMATE AUTO WTO the message for SA z/OS to trap.</p> <p>SUPPRESS The message is suppressed from z/OS consoles and also all TD queues. CICS reserves the right not to suppress messages that it determines are critical.</p> <p>The default action is no action if the MSGDISP keyword is specified. This allows for the removal of an action based on message content. If no MSGDISP keyword is specified the default action is AUTOMATE.</p>
TDQUEUE	<p>This keyword specifies the matching Transient Data Queue.</p> <p>Do not specify DFH messages with a TDQUEUE value.</p> <p>The format of the data is a single alphanumeric word up to 4 bytes in length. The specification of this keyword in a message policy causes the message information to be matched against all messages written to the Transient Data Queue that is specified. If a match is made, the message may be WTO'd depending upon any other keywords that are specified.</p>

Using the CICS Automation Message Exit

Table 2. Keywords in CICS Message Definitions (continued)




Keyword	Description
TOKEN	<p>This keyword specifies the matching token values for user messages. It is optional, but if present, only messages that match the values specified are WTO'd. Its format is:</p>  <p>Tokens:</p>  <p>TDQ Actions:</p>  <p>Where <i>number</i> is the word number to be checked and <i>word_text</i> is the text to be checked.</p> <p>Words are counted from the beginning of the message starting from 1. A word is considered to be a contiguous set of alphanumeric characters. Any non-alphanumeric characters are treated as delimiters.</p> <p>The <i>word_text</i> specification is checked for the length specified. This allows for trailing data to be ignored. The format of the number is a single integer. The format for <i>word_text</i> is a single alphanumeric word; no blanks or special characters are allowed. The case of the value is ignored.</p> <p>The actions that are available are:</p> <p>NONE Specifies that no action is to occur for the matching message.</p> <p>AUTOMATE AUTO WTO the message for SA z/OS to trap. The default action is AUTOMATE.</p>

Table 2. Keywords in CICS Message Definitions (continued)

Keyword	Description
MSGDISP	<p>This keyword specifies the default actions for a message ID. The format of the data is dependent on the type of message being processed and is as follows:</p> <div style="margin-left: 20px;"> </div> <p>CICS Actions:</p> <div style="margin-left: 20px;"> </div> <p>TDQ Actions:</p> <div style="margin-left: 20px;"> </div> <p>The actions that are available are:</p> <p>NONE The message is not acted on by the exit. Use this option to set a default of no actions when used with the INSERT or TOKEN keywords.</p> <p>AUTOMATE AUTO WTO the message for SA z/OS to trap.</p> <p>SUPPRESS The message is suppressed from z/OS consoles and also all TD queues. CICS reserves the right not to suppress messages that it determines are critical.</p> <p>The default action is AUTOMATE if no INSERTs or TOKENs are present. If either INSERTs or TOKENs are present the default action is NONE.</p>

To define a CICS message to be WTO'd, do the following:

1. Specify the message ID in the MESSAGES/USER DATA policy item.
2. Enter the USR command against the message ID. Specify OFFSET in the **Keyword** field and a data value of 1 in the **Data** field.
3. Optionally enter the USR command against the message ID and specify INSERT in the **Keyword** field. Optionally specify the actions for the INSERT contents.

Note: You can specify several INSERT keywords.

4. Optionally enter the USR command against the message ID and specify MSGDISP in the **Keyword** field to detail the actions desired.
5. Specify either the AUT or the CMD command, or both. Use the AUT command to specify capturing the message or other automation functions. Use the CMD command to specify commands that are to be executed when the message occurs.

To define a user message to be WTO'd from a Transient Data Queue, do the following:

1. Specify the message ID in the MESSAGES/USER DATA policy item.

Using the CICS Automation Message Exit

2. Enter the USR command against the message ID. Specify OFFSET in the **Keyword** field and the word number that the message ID occurs at in the message in the **Data** field.
3. Enter the USR command against the message ID. Specify TDQUEUE in the **Keyword** field and the name of the Transient Data Queue that the message is written on in the **Data** field.
4. Optionally, enter the USR command against the message ID and specify TOKEN in the **Keyword** field, and the words and their values that are to be matched in the **Data** field.
5. Optionally enter the USR command against the message ID and specify MSGDISP in the **Keyword** field to detail the actions desired.

Refreshing Policy Data

To load the information into CICS address spaces issue the INGAMS REFRESH command. As a part of the ACF load, each CICS that had changes to MESSAGES/USER DATA policy will be reloaded with any changes. If you want to disable the exits, either delete the messages out of the policy or change the OFFSET keyword on every message to some other name, for example, UFFSET, and rebuild and reload the ACF.

Partial Message IDs and Performance

A partial message ID is one where only a part of the message ID is specified. The partial message ID is restricted to the leftmost part of the message ID. You cannot specify a partial message ID, such as XXX**YY, only XXX would be acceptable. For messages beginning with DFH, the exit does not support partial message IDs and the full message must always be specified. For messages in the TD queues, the first three bytes of the message ID are the minimum required to specify a message to trap in the exit. You can specify a partial message ID within this limitation.

It is recommended that the TOKEN keyword not be used with partial message IDs for messages. This causes a performance degradation because the token data must be matched for the partial message and then again for any specific message.

You can specify MSGDISP policy information on the partial message IDs. This sets a default for actions for all specific message IDs that match the partial message ID. You do not have to specify more specific message IDs. Instead a partial message ID and a matching set of TOKEN keywords can be used to specify the actions for a message. Be aware that token scanning is much less efficient than message ID scanning.

You can use the partial message IDs to improve exit performance. An extreme example might be if you are trapping messages ABC000-ABC100. You might choose to create partial message IDs for ABC00-ABC10. The partial message IDs would specify MSGDISP=(NONE) to prevent them supplying default actions. This would cause the exit to first scan the set of partial message IDs ABC00-ABC10 or 11 checks. Then, assuming it found a match on ABC10, it would scan the specific messages ABC110-ABC119 for a total of 21 checks as opposed to 101 for the original set. This would require 112 MESSAGES/USER DATA policy definitions. 101 for messages ABC000-ABC100 and 11 for messages ABC00-ABC10. In normal circumstances you can ignore the usage of partial message IDs for performance reasons and use them for functional reasons.

Migration and Coexistence

This section contains information on coexistence of and migration from SA z/OS 3.1 or SA z/OS 2.3 to SA z/OS 3.2.

Migration

The general migration process is described in the “*Data Management*” chapter in *IBM Tivoli System Automation for z/OS Defining Automation Policy*.

The major changes in this release from previous releases are:

- Link monitoring is now implemented by exploiting CICSplex SM capabilities.

Coexistence

The new link monitoring is only available with SA z/OS 3.2. Note that the SA z/OS 3.2 configuration data cannot contain any definitions for the new link monitoring function in SA z/OS 3.2 if it is to be made available to a SA z/OS 2.3 or SA z/OS 3.1 system.

Migration and Coexistence

Chapter 3. How to Set Up the Functions of CICS Automation

This chapter explains how to set up the functions of CICS Automation for your specific needs. For the setup of base functions, for example, starting and stopping subsystems, see the SA z/OS documentation.

Automating Recovery For Transactions

CICS Automation provides automation for short-on-storage conditions.

You can control automated recovery through the following policy items of the APPLICATION object:

MINOR RESOURCE FLAGS

With these flags, you can switch automated recovery on and off for transactions or for a certain problem area. To do this, define a minor resource and set its **Recovery** flag as required. For the definition of minor resources, see *IBM Tivoli System Automation for z/OS Defining Automation Policy*. The names of these minor resources must be as follows:

Problem area	Minor resource name
Short-on-storage conditions	SOS
Transaction recovery	TRAN[.trans_id]

For transaction recovery, you can also define second-level minor resources by suffixing TRAN with the transaction name. The recovery flag of the TRAN minor resource applies to all transactions of the respective application; TRAN.trans_id only applies to the trans_id transaction. The transaction-specific recovery flag overrides the general TRAN flag.

When no minor resources are defined, CICS Automation acts according to the recovery setting of the application (AUTOMATION FLAGS policy item). When no second-level minor resource is defined for a transaction, the TRAN minor resource is applied. If that does not exist either, the application setting is applied. You only need to define minor resources when the recovery setting for a lower level is to be different from the next higher level.

MESSAGES/USER DATA

For every recovery type, there are one or more keywords that are used to specify how recovery is to proceed. These keywords are:

Problem area	Keywords
Short-on-storage conditions	RCVRSOS (see page 35)
Transaction recovery	ABCODETRAN (see page 29), RCVRTRAN (see page 36)

Transaction recovery is the most complex of these recovery types. Therefore this type is used to explain recovery configuration in more detail.

How to Define Transaction Recovery

Customization of transaction recovery consists of:

1. Identifying the transactions that will have recovery automation.
2. Identifying the error threshold level when recovery should be stopped.
3. Identifying specific abend codes when you want recovery procedures to take place (there are probably several that you would want to ignore).
4. Specifying the recovery procedure, which usually consists of invoking a command, a routine, or sending notification to an operator, or all three.

The recovery itself is typically triggered from the AT by calling the EVEERTRN routine when certain messages arrive at NetView. EVEERTRN then consults the ACF in order to learn what it has to do for recovery.

The following sections illustrate the configuration process by an example.

Specifying the Transactions to Be Recovered

If recovery is enabled for the CICS1 application on the application level, and you want to enable it also for transactions PAYR, DBTS, and BLNG, but not for any other transaction, you must define four minor resource flags for CICS1 in the customization dialogs:

- TRAN
- TRAN.BLNG
- TRAN.DBTS
- TRAN.PAYR

Set the recovery automation flag to NO for TRAN and to YES for the three second level minor resources. For more information, see *IBM Tivoli System Automation for z/OS Defining Automation Policy*.

Defining Recovery Thresholds

You can specify that recovery is to be stopped when the number of abends within a certain time interval reaches a certain threshold. To do this, define thresholds in the MINOR RESOURCE THRES policy item of the APPLICATION policy object. The thresholds must have the name TRAN or TRAN.*tranid*, where the values of the TRAN thresholds will be used for all transactions *tranid* for which no TRAN.*tranid* thresholds exist. The **Critical** value of the thresholds will be used.

If you want to stop recovery specifically for PAYR if two or more abends occur within one hour, you must enter the values on the Thresholds Definitions panel as follows:

Resource	Levels					
	Critical		Frequent		Infrequent	
	Number	Interval (hh:mm)	Number	Interval (hh:mm)	Number	Interval (hh:mm)
CICS1.TRAN.PAYR	2	01:00	2	05:00	2	24:00

Figure 1. Thresholds Definitions Panel

For more details, see *IBM Tivoli System Automation for z/OS Defining Automation Policy*.

Selecting the Abend Codes

The abend codes for which recovery is to take place are specified through the ABCODETRAN keyword in the MESSAGES/USER DATA policy item for CICS1. If you want to initiate recovery for transaction PAYR only when the abend code is AEI0 or AKC3, you must create the ABCODETRAN entry in the Message Processing panel and associate codes with this entry as displayed in Figure 2:

COMMANDS HELP

Code Processing Row 1 to 6 of 21

Command ==> SCROLL==> PAGE

Entry Type : Application PolicyDB Name : SCENARIO
Entry Name : CICS1 Enterprise Name : TEST

Subsystem : CICS1
Message ID : ABCODETRAN.PAYR

Enter the value to be passed to the calling CLIST when this resource issues the selected message and the following codes are contained in the message.

Code 1	Code 2	Code 3	Value Returned
_____	AEI0_____	_____	INCLUDE_____
_____	AKC3_____	_____	INCLUDE_____
_____	*_____	_____	EXCLUDE_____
_____	_____	_____	_____
_____	_____	_____	_____

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

Figure 2. Code Processing Panel

For more details, see “ABCODETRAN: Transaction Abend Recovery” on page 29.

Specifying Recovery Actions

For transaction recovery the following messages must be defined:

For more details, see “RCVRTRAN: Transaction Recovery” on page 36.

How to Set Up Health Checking

Health checking allows you to execute programs that check the health of an application running under CICS. CICS Automation initiates a health check program by sending a request across the program-to-program interface to the target CICS. CICS receives the request, processes the program, and sends the results back to NetView.

Note: Sample health check code is provided with CICS Automation. This code can be modified and used to check the availability of critical resources, such as DB2® or IMS.

Health checking is set up in the following manner:

Step 1: The health check program is written

The actual health check program is executed on the target CICS. According to the health check protocol, one of the following is returned:

- An acknowledgment (ACK) stating that the program completed successfully.

How to Set Up Health Checking

- A negative acknowledgment (NACK) stating that the program did not complete successfully. If a NACK response is given, data can be passed back to NetView describing the error condition.

The ACK or NACK are returned through the use of a DFHCOMMAREA. The user-written health check program is linked to by a CICS Automation health check program, which passes the 104-byte DFHCOMMAREA. The first four bytes are reserved for the characters ACK or NACK. The last 100 bytes can be used for a NACK message if the user-written program encounters an error. Refer to the samples for the format of the DFHCOMMAREA, and for an example of how to use the DFHCOMMAREA to return the response.

Step 2: The health check programs are defined to CICS

Use *CICS Transaction Server for z/OS Resource Definition (Online)* to do this.

Step 3: The health check program is defined to CICS Automation

The health check program is defined to CICS Automation in the MESSAGES/USER DATA policy item HEALTHCHK (see “HEALTHCHK: Health Checking” on page 32 on the **User Defined Data** panel.

How to Set Up Link Monitoring

CICS communicates with other systems or subsystems using either intersystem communication (ISC) or interregion communication (IRC). In either ISC or IRC, communication between different systems or subsystems takes place across predefined sessions. Sessions are logical links that are allocated whenever there is a need to communicate.

To activate link monitoring, perform the following steps:

1. Create a monitor resource (MTR) for each connection.
2. Enter the monitored object according to your naming conventions (for example, CPSM.CICSTOR1.CONNECT.CT12)
3. Define code processing for the message ID ING150I in the monitor resource's MESSAGE/USER DATA policy item to map the CPSM severities to valid health states. Use the first word of the Value Returned field to define a recovery command to be executed.

For more details, see “Passive, Event-Based Health Monitoring” in *IBM Tivoli System Automation for z/OS Defining Automation Policy*.

Adding Local Applications to the CICS Automation Operator Interface

Option 99, Local Functions, from the CICS Automation main menu, provides you with a way to add your local applications to the CICS Automation interface.

To do this, write a module named EVEEU000 using the programming notes described below. This is the module that is called when option 99 is selected.

These programming notes assume that you understand how to write a NetView panel handler exec. These notes clarify unique functions or conventions used with CICS Automation. For your panel to be logically consistent with the CICS Automation interface, incorporate these functions.

Programming notes:

1. To exit CICS Automation (PF2) or to return to the main menu (PF4), code the following after displaying your panel and accepting the input:

Adding Local Applications to the CICS Automation Operator Interface

```
WHEN VIEWAID = 'PF2' | VIEWAID = 'PF14' THEN
DO
  EVE_PF2 = 'YES'
  'GLOBALV PUTT EVE_PF2'
  EXIT 0
END
```

and

```
WHEN VIEWAID = 'PF4' | VIEWAID = 'PF16' THEN
DO
  EVE_PF4 = 'YES'
  'GLOBALV PUTT EVE_PF4'
  EXIT 0
END
```

2. When you call a module and you return from that module, you should exit if the called module displays a panel and PF2 or PF4 was pressed. To check for this, code the following after the call.

```
'GLOBALV GETT EVE_PF2'
IF EVE_PF2 = 'YES' THEN
DO
  EXIT 0
END
```

and

```
'GLOBALV GETT EVE_PF4'
IF EVE_PF4 = 'YES' THEN
DO
  EXIT 0
END
```

3. To handle a fast-path command entered on your panel:
 - a. Add the following to the beginning of the program:
'SIGNAL ON HALT'
 - b. Add the following routine into the program:

```
HALT:
  EVE_PF2 = 'YES'
  'GLOBALV PUTT EVE_PF2'
  EXIT 0
```

- c. Add the following code after displaying your panel and accepting input:

```
WHEN VIEWAID = 'ENTER' & CMD ^= '' THEN
DO
  IF SUBSTR(CMD,1,1) = '=' THEN
  DO
    PARSE VAR CMD '=' REST
    CMD = 'EVEE0000 ' || REST
  END
  'CMD HIGH 'CMD
END
```

Note: In this code, CMD is the command line on the NetView panel.

4. If you code a menu panel, add the following code to check for fast-path when your program is entered:

```
'GLOBALV GETT EVE_SELECTION'
IF EVE_SELECTION ^= ''
DO
  PARSE EVE_SELECTION MYSELECTION '.' EVE_SELECTION
  'GLOBALV PUTT EVE_SELECTION'
END
```

Adding Local Applications to the CICS Automation Operator Interface

5. On entry, or returning from a called program, to get the CICS subsystem name (if the previous program had a valid name and saved it) code the following:

```
'GLOBALV GETT EVESELNM'  
MYNAME = EVESELNM
```

6. Always validate a new CICS name before storing it for other programs to use. The following is an example of validation:

```
'CICSQRY REQ=VALIDATE,TYPE=CICS,NAME='MYNAME  
IF RC ^= 0  
  DO  
    write your error message  
  END  
ELSE  
  EVESELNM=MYNAME  
  'GLOBALV PUTT EVESELNM'
```

Using Linemode Functions

Linemode functions allow the operator or user-written routines to access the following special CICS Automation functions without using the CICS Automation panels:

- Health checking
- System Init Table override
- Message options
- CICSPOST
- CEMTPPI

The linemode routines allow the extension of automation from user-written routines. The user-written routine issues the linemode command during NetView initialization or at a specific time or day. A message and return code is given to the calling routine to verify that the requested operation was successful.

Health Checking

Linemode health checking makes it possible to manipulate health-check routines from a user-written command. A health-check program is a user-written routine which executes periodically to ensure that a critical application is capable of supporting its users. The actions supported include suspending and resuming the health-check program. Other actions are supported.

System Init Table Override

Linemode System Init Table (SIT) overrides give user-written routines the capability of setting the SIT overrides, which can then be used by CICS Automation to control the startup of the CICS. A typical use of this linemode command will be to enable automation to perform cold startups on a given day of the week. For example, using SA z/OS timer facilities, a user could set a timer to set the overrides to cold-start every Monday morning. Then, using service periods, the CICS system could be recycled, and a cold start would be performed.

Message Options

Linemode message options enables a user-written routine to change the message header options which display on the operator panel. Typical use of this command is during NetView initialization, when a user-written routine would set the domain-wide defaults.

CEMTPPI

CEMTPPI allows you to code a CEMT command:

1. In your own automation routines.
2. In the NetView automation table.
3. In the policy database in certain items of the APPLICATION policy object such as STARTUP or SHUTDOWN, or in certain message IDs, for example RCVRSOS.

Refer to “CEMTPPI: CEMT PPI Short Syntax” on page 38 for details.

It accepts CEMT input as data, issues an MVS™ Modify command on a console, and sends a response back to the originating task.

How to Implement Remote Site Recovery for VSAM RLS (CICS TS Function Only)

CICS TS provides support for remote site recovery where VSAM data sets are used in RLS mode at the primary site. Using this RLS support for remote recovery, you can switch over to the remote site without suffering indeterminate or unreported loss of data integrity.

To invoke CICS RLS support for off-site recovery, you must start CICS systems with INGREQ using start type AUTO and specifying OFFSITE=YES in the **Appl Parm**s field of the INGREQ input panel. See “Startup” on page 56.

With RLS recovery in operation during an emergency restart, CICS prevents any data sets from being accessed in RLS mode until CICS has completed all outstanding RLS recovery work and it has received a ‘GO’ response to WTOR DFHFC0575.

The operator should reply ‘GO’ to the message only when all the CICS regions being restarted with OFFSITE=YES have issued message DFHFC0575 indicating that they have completed their RLS recovery.

CICS TS provides a sample REXX exec DFH\$OFAR to be used to automatically reply ‘GO’ to the WTOR for each participating CICS system, when appropriate.

To be able to use the CICS TS-provided sample REXX exec DFH\$OFAR, you will need to copy it from the CICS TS DFHSAMP library into a DSICLD concatenated library. Refer to the CICS TS documentation for more information.

DFH\$OFAR requires that a unique control file (a sequential data set) be defined containing all the participating CICS systems. This control file must be accessible from any participating MVS image within the sysplex. Refer to the prolog in the REXX exec DFH\$OFAR for more detailed information.

CICS Automation provides the AT entries required to drive the CICS TS-provided REXX exec DFH\$OFAR. Merge these entries into your own AT to be able to use this function.

How to Implement Remote Site Recovery for VSAM RLS

Chapter 4. MESSAGES/USER DATA Entries for CICS Automation

You must specify any information for CICS Automation in the SA z/OS policy database through the customization dialog. In most cases, the customization dialog itself restricts you to the format in which this information must be entered. There are, however, a number of CICS application-specific automation parameters that must be specified as entries in the MESSAGES/USER DATA policy item of the appropriate application. For further information about the MESSAGES/USER DATA policy item, see *IBM Tivoli System Automation for z/OS Defining Automation Policy*.

The following chapter contains detailed descriptions of these automation entries. However, a general understanding of the MESSAGES/USER DATA policy item is assumed.

CICS-Specific MESSAGES/USER DATA Keywords

The following keywords are specific for CICS Automation. They need to be entered in the MESSAGE ID field on the Message Processing panel.

Entry	Description
"ABCODESYSTEM: System Abend Recovery" on page 28.	Use this ID to define actions to be taken for specific abend codes.
"ABCODETRAN: Transaction Abend Recovery" on page 29.	Use this ID to define actions to be taken for transaction abend codes.
"CICSINFO: Display Information" on page 31.	Use this ID to define the effect of INGCICS REQ=INFO
"HEALTHCHK: Health Checking" on page 32.	This ID is used to define the health check routines.
"LISTSHUT: Transaction Purging During Shutdown" on page 33.	Use this ID to define those transactions running under this CICS subsystem that should or should not be purged during a shutdown.
"RCVRSOS: Short-On-Storage Handling" on page 35.	Use this ID if you want CICS Automation to take action for short on storage conditions.
"RCVRTRAN: Transaction Recovery" on page 36.	Use this ID to define actions to be taken when this specific transaction has abended.

ABCODESYSTEM: System Abend Recovery

Use this keyword to either include specific abend codes in recovery or exclude them from recovery.

Use Code Processing and enter the following data:

Code 1	Code 2	Code 3	Value Returned
<i>msg</i>	<i>abend1</i>	<i>abend2</i>	RESTART or NORESTART

Keyword and Parameter Definitions

CODE

Defines which abends are restartable, as shown in the following descriptions:

msg

The abend message ID.

abend1 and *abend2*

The specific abend codes or qualifiers.

RESTART|NORESTART

Indicates whether or not to initiate a restart for this subsystem when this specific message/abend code(s) occur(s).

Comments and Usage Notes

1. Abend qualifiers vary depending on the AT. Refer to sample tables to determine the qualifiers for each message.
2. If a CICS message (DFHxxxxx) is trapped and included in the ABCODESYSTEM table, then it is not usually necessary to code the corresponding IEF450I message with the same user abend code (Uxxxx) in the table. An exception to this may be DFHKE1800, which is issued so closely in time to IEF450I that it may be processed before the DFHKE1800. It is therefore recommended that you either:
 - Add both DFHKE1800 and IEF450I with U1800 to the table, or
 - Exclude DFHKE1800 from the table and from the AT as well.
3. When CICS issues one of the following abend messages a restart of CICS requires INITIAL to be specified as the startup type:

DFHLG0736	DFHLG0738	DFHLG0740	DFHRM0134	DFHRM0136
DFHRM0144	DFHRM0401	DFHDM0106	DFHSI1542	

In these cases CICS Automation prevents a restart by ARM. Rather, the restart policy must be defined with the ABCODESYSTEM entry.

Examples of Usage

In the following example, a restart will be initiated for message IEF450I if the qualifier is S222. If messages DFH607 or DFH3784 are issued, restarts will be initiated. No restart will be initiated for message DFH0408.

Code 1	Code 2	Code 3	Value Returned
IEF450I _____	S222 _____	* _____	RESTART _____
DFH0607 _____	* _____	* _____	RESTART _____
DFH3784 _____	* _____	* _____	RESTART _____
DFH0408 _____	* _____	* _____	NORESTART _____

ABCODETRAN: Transaction Abend Recovery

Use this keyword to define actions to be taken for transaction abend codes.

Use Code Processing and enter the following data:

Code 1	Code 2	Code 3	Value Returned
<i>tran</i>	<i>abend</i>	<i>pgm</i>	INCLUDE or EXCLUDE

Keyword and Parameter Definitions

ABCODETRAN[.*tran*]

You can add the name of a transaction as a suffix to the keyword. In this case the specifications of the CODE attribute(s) will only apply to this transaction.

CODE

Defines which abends are recoverable, as shown in the following descriptions:

tran

The transaction ID.

abend

The abend code.

pgm

The program that abended.

INCLUDE|EXCLUDE

Indicates whether or not to initiate a recovery for this transaction, abend code, and program. Use INCLUDE to initiate a recovery and EXCLUDE if you do not want a recovery initiated.

Comments and Usage Notes

1. The transaction name is specified as either ABCODETRAN.*tran* or the first value of the CODE attribute. Use ABCODETRAN.*tran* when you want all of the specifications to apply to one specific transaction. Use the CODE attribute when you want to code several transactions.
2. Ensure that the abend transaction message is issued as a WTO. For more details, see "Defining CICS Messages" on page 12

Examples of Usage

In the following example, recovery will not take place for transaction CSFE if the abend code is ATNI or AKC3. Recovery will take place for all other transaction and abend codes.

Code 1	Code 2	Code 3	Value Returned
CSFE	ATNI	*	EXCLUDE
CSFE	AKC3	*	EXCLUDE
*	*	*	INCLUDE

The following example shows the user definitions that are needed to issue a WTO for an abend transaction message:

ABCODETRAN: Transaction Abend Recovery

Keyword

Data

TDQUEUE _____

CSMT _____

OFFSET _____

1 _____

CICSINFO: Display Information

These commands are issued when the INGCICS REQ=INFO command is used to display the state of the selected CICS region. The commands are issued via the MODIFY subsystem ID on an MVS EMCS console and the resulting messages are either displayed on the INGCICS panel or written to the users NetView console.

For further information about the INFO request, see the description of the INGCICS command in *IBM Tivoli System Automation for z/OS Operator's Commands*.

Use User-Defined Processing and enter the following data:

Keyword	Data
CICSCMD	(<i>description,CICS command</i>)

Keyword and Parameter Definitions

description

The description is text that will be placed before the output of the CICS command. This can be used to identify the command output in the output stream. The description can be any string, but must be enclosed in quotes.

CICS command

The CICS command is the command to be executed. This command will be appended to a MODIFY subsystem and issued as an MVS command to an EMCS console. The output will be collected and displayed. The command can be any valid CICS transaction that will write to an MVS console. The command must be enclosed in quotes.

You may code multiple CICS commands, separated by a comma, in order to group results under a common description.

Comments and Usage Notes

This policy is required for correct operation of the INGCICS command and also PF10 of the DISPINFO panel.

Examples of Usage

```
Keyword
Data
CICSCMD
('DISPLAY ACTIVE TASKS','CEMT I TA')

CICSCMD
('DISPLAY SYSTEM INFORMATION', 'CEMT I SYS')
```

HEALTHCHK: Health Checking

Use this keyword to define the health check routines.

Use User-Defined Processing and enter the following data:

Keyword	Data
FUNCTION	(<i>n</i> , <i>pgm</i> , <i>int</i> , <i>resp</i> ,AUTO NOAUTO, ' <i>desc</i> ')

Keyword and Parameter Definitions

n Up to 10 health check entries can be specified for each CICS subsystem. 0 to 9 identifies a health check entry.

pgm

The program name as defined to the CICS region.

int

The interval, expressed in hours, minutes, and seconds, after which this health-check program is to be rerun. The format is *hh:mm:ss*. The maximum is 99:59:59.

Note: The interval must be greater than the response time limit.

resp

How long to wait for a response before sending an alert to the operator. This is expressed in seconds. The maximum is 120 seconds.

AUTO|NOAUTO

Specify NOAUTO if you only want this health check routine to be activated through the operator interface. The default AUTO activates the routine automatically when the CICS subsystem status is changed to UP, and deactivates it when the CICS subsystem terminates.

desc

The description of this health check routine as it appears on the CICS Automation Health Checking panel (see "Health Checking" on page 62). Up to 20 characters can be used.

Comments and Usage Notes

Refer to "How to Set Up Health Checking" on page 21 for more information about health checking.

Examples of Usage

In the following example, there are two health check programs that will be run, HCPAY1 and IMS, and these will be run every 2 minutes. If no response arrives within 15 seconds, the operator is notified.

```
Keyword
Data
FUNCTION
(0,HCPAY1,00:02:00,15,, 'Check payroll database')
_____
_____

FUNCTION
(1,IMS,00:02:00,15,, 'Check IMS001')
_____
_____
```

LISTSHUT: Transaction Purging During Shutdown

Use this keyword to define those transactions running under this CICS application that should or should not be purged during a shutdown.

Use User-Defined Processing and enter the following data:

Keyword	Data
EXCLUDE	* <i>transid</i>
INCLUDE	* <i>transid</i> (<i>transid</i> ,FORCE)

Keyword and Parameter Definitions

EXCLUDE

Do not purge this transaction during a shutdown.

* Specifies any transaction name.

transid Specifies a specific transaction name.

Any number of transactions may be specified by repeating the EXCLUDE keyword and its data.

INCLUDE

Purge this transaction during a shutdown.

* Specifies any transaction name.

transid Specifies a specific transaction name.

(*transid*,FORCE)

Specifies that the transaction named is to be FORCE purged.

Any number of transactions may be specified by repeating the INCLUDE keyword and its data.

Comments and Usage Notes

1. If this entry is not used, no transactions are purged.
2. When the LISTSHUT entry is used for this application, the CICSPURG command list must be coded in the shutdown policy for this application. For information on CICSPURG, see "CICSPURG: Purge Transactions" on page 42.
3. For CICS transactions that are not purgeable (SPURGE=N) the FORCE option must be used to purge the transaction. CICS will return PURGE FAILED for the purge of a non-purgeable transaction.

Examples of Usage

The following example specifies that:

1. The PAYR transaction is specifically not purged during CICS shutdown
2. BAT1 is purged
3. BAT2 is force purged

LISTSHUT: Transaction Purging During Shutdown

Keyword

Data

EXCLUDE _____

PAYR _____

INCLUDE _____

BAT1 _____

INCLUDE _____

(BAT2, FORCE) _____

RCVRSOS: Short-On-Storage Handling

Use this keyword to notify the operator and optionally issue a command when a short-on-storage condition exceeds the specified time limit.

- Use User-Defined Processing and enter the following data:

Keyword	Data
TIMELIMIT	<i>mm:ss</i>

- Optionally, use Command Processing and enter the following data:

Pass/Selection	Automated Function/'*'	Command Text
—	—	<i>cmd</i>

Keyword and Parameter Definitions

TIMELIMIT

Specifies the time limit that a short-on-storage condition can exist before the commands specified by the CMD attribute are executed.

cmd

The command or commands to be issued when the short-on-storage condition exceeds the time limit specified with the TIMELIMIT attribute.

Comments and Usage Notes

1. There is a sample command list, EVEERDMP, that can be specified for the CMD attribute when a dump is to be produced.
2. The **Critical** value of the SOS thresholds of the subsystem is used to determine how often the specified commands are allowed to be issued within a specific time frame. The SOS thresholds must be defined in the customization dialog with the application's MINOR RESOURCE THRES policy item.

Examples of Usage

The following example shows how the TIMELIMIT attribute can be specified:

```
Keyword
Data
TIMELIMIT _____
00:30 _____
```

The following shows how the CMD attribute can be specified:

```
Pass/Selection Automated Function/'*'
Command Text
EVEERDMP CICS1 _____
```

The example specifies that the operator is notified and a dump produced if CICS is short on storage for more than 30 seconds.

RCVRTRAN: Transaction Recovery

Use this keyword to define actions to be taken when transaction abends occur.

Use Command Processing and enter the following data:

Pass/Selection	Automated Function/'*'	Command Text
—	—	<i>cmd</i>

Note: The RCVRTRAN keyword may be followed with a dot and a transaction ID *tran*.

Keyword and Parameter Definitions

tran

A specific transaction. If this is not used, then the commands apply to all transactions.

cmd

The command or commands to be issued when the transaction abends.

Comments and Usage Notes

1. The **Critical** value of the TRAN or TRAN.*tranid* thresholds for the subsystem is used to indicate how many abends can occur before automation is stopped. The thresholds must be defined in the customization dialog with the application's MINOR RESOURCE THRES policy item.
2. The SA z/OS variable EHKVAR1 is set with the name of the transaction. This allows you to tailor your commands using the transaction name, such as disabling the transaction. Also, when available, EHKVAR2 will be set to the abend code and EHKVAR3 set to the program name.
3. The RCVRTRAN entry is used by the EVEERTRN routine that is delivered with CICS Automation. EVEERTRN is typically called from the AT.
4. Ensure that the abend transaction message is issued as a WTO. For more details, see "Defining CICS Messages" on page 12

Examples of Usage

The following entry states that the command shown is to be executed for all transactions that do not have specific entries for them. It is the default command.

Pass/Selection	Automated Function/'*'	Command Text
		MSG OP1,TRAN &EHKVAR1 FAILED _____ _____

The following example shows the user definitions that are needed to issue a WTO for an abend transaction message:

Keyword	
Data	
TDQUEUE	_____
CSMT	_____

OFFSET	_____
1	_____

Chapter 5. CICS Automation Routines and Commands

This section is intended to help system and application programmers write programs that use the CICS Automation function. The following are described:

Operator Commands

The following commands are operator commands and can be invoked by operators or via PIPES:

- CEMTPPI** Issue CICS CEMT commands, see “CEMTPPI: CEMT PPI Short Syntax” on page 38.
- CICSHLTH** Line mode Health Check, see “CICSHLTH: Linemode Health Checking” on page 39.
- CICSOVRD** Specify startup and shutdown options dynamically, see “CICSOVRD: Linemode SIT Override” on page 40.

Automation Policy Commands

The following commands are meant to be executed from the automation policy:

- CICSPURG** Purge transactions at CICS shutdown, see “CICSPURG: Purge Transactions” on page 42.
- CICSRSYC** Refresh CICS information at Agent startup. Run from the ACORESTART policy, see “CICSRSYC: CICS Resync” on page 43.
- CICSSHUT** Shuts a CICS subsystem down, see “CICSSHUT: Shutdown Processor” on page 44.
- EVEERDMP** Creates a system DUMP of the CICS address space, see “EVEERDMP: CICS Dump” on page 45.
- CMASSHUT** Shuts a CICS CMAS address space, see “CMASSHUT: CICSplex SM Address Space Shutdown” on page 46.

Application Programming Interfaces

The following commands are to be invoked from user REXX programs:

- CICSQRY** Get information about a CICS subsystem, see “CICSQRY: Name Lookup” on page 48.
- CICSRCMD** Route a command to an Agent that is controlling a CICS subsystem, see “CICSRCMD: Request a CICS Function” on page 52.

Operator Commands

The following commands are operator commands and can be invoked by operators or via PIPES:

- CEMTPPI** Issue CICS CEMT commands, see “CEMTPPI: CEMT PPI Short Syntax” on page 38.
- CICSHLTH** Line mode Health Check, see “CICSHLTH: Linemode Health Checking” on page 39.
- CICSOVRD** Specify startup and shutdown options dynamically, see “CICSOVRD: Linemode SIT Override” on page 40.

CEMTPPI: CEMT PPI Short Syntax

CEMTPPI allows you to code a CEMT command:

1. In your own automation routines.
2. In the AT.
3. In the **CMD Processing** panel of the customization dialogs (for example, SHUTDOWN or MESSAGES policy items).

It accepts CEMT input as data, issues an MVS Modify command on a console, and sends a response back to the originating task.

Format

CEMTPPI *subsys cent-command-stream*

Keyword and Parameter Definitions

subsys

The symbolic name by which this CICS subsystem is known to SA z/OS.

cent-command-stream

The CEMT command stream, such as SET TASK DISABLE. Do not prefix the command with CEMT.

Comments and Usage Notes

1. This command can only be used if PPI is set up accordingly.
2. This command can route across domains.
3. The CEMT PERFORM SHUTDOWN option is not allowed across the program-to-program interface. Therefore, it cannot be issued with CEMTPPI.

CICSHLTH: Linemode Health Checking

CICSHLTH allows an operator or user-written routine to control health checking without using the CICS Automation operator interface.

Format

```
CICSHLTH NAME=subsys, {ACTION=START, PROGRAM=programe |
                        ACTION=STATUS, PROGRAM=programe |
                        ACTION=RESUME, PROGRAM=programe |
                        ACTION=SUSPEND, PROGRAM=programe |
                        ACTION=STOP, PROGRAM=programe |
                        ACTION=CHECK, PROGRAM=programe}
```

Keyword and Parameter Definitions

subsys

The name of the CICS subsystem.

programe

The name of the user-written health check program.

ACTION=START

Used to initiate health check processing.

ACTION=STATUS

Used to determine if the health check program is active or inactive, and normal or abnormal regarding its most recent execution.

ACTION=RESUME

Used to continue a process that was temporarily suspended.

ACTION=SUSPEND

Used to temporarily stop a process.

ACTION=STOP

Used to stop health check processing.

ACTION=CHECK

Used to submit a status check regardless of the status and scheduled time interval of the health check program.

Comments and Usage Notes

Actions can only be performed using programs that are specified under the HEALTHCHK keyword in the MESSAGES/USER DATA item of the APPLICATION policy object for the subsystem.

Examples of Usage

Example 1: Status of a Health Check Program: Command:

```
CICSHLTH NAME=CICS01A, ACTION=STATUS, PROGRAM=EVECHLTH
```

Message Response:

```
EVE441I HEALTH CHECK PROGRAM EVECHLTH STATUS INACTIVE
EVE445I ABNORMAL RESPONSE ON 10/25/04 09:51:35 -
```

Example 2: Start of a Health Check Program: Command

```
CICSHLTH NAME=CICS01A, ACTION=START, PROGRAM=EVECHLTH
```

Message Response:

```
EVE436I CICS01A HEALTH START FOR EVECHLTH SUCCESSFUL.
```

CICSOVRD: Linemode SIT Override

CICSOVRD allows you to set CICS SIT override conditions prior to CICS startup. (Otherwise, CICS Automation only allows you to set override conditions through the INGREQ input panel; see “Starting and Stopping Resources” on page 56.)

Format

```
CICSOVRD NAME=subsys,STARTTYPE=type,  
          ACTION=SET,OVERRIDE=delimdatadelim,  
          KEYPOINT=[REQuired|OPTional]
```

```
ACTION=STATUS
```

Keyword and Parameter Definitions

type

The type of startup.

subsys

The name of the CICS subsystem.

ACTION=SET

Used to change the SIT override for a CICS subsystem.

ACTION=STATUS

Used to inquire about the SIT options for a CICS subsystem.

delim

The character that is used to delimit the override data. The first character after the '=' sign is taken to be this delimiter.

data

The override data to be used to override the SIT options.

KEYPOINT=[REQuired|OPTional]

Used to specify if a warm keypoint is required for the CICS subsystem, before these overrides can be used.

Comments and Usage Notes

1. The CICSOVRD command does not start the named CICS. The overrides are saved and used for subsequent starts of the CICS. The SIT overrides can be displayed by using the ACTION=STATUS option.
2. This linemode command returns the following message to the invoking routine:

```
EVE556I  CICSOVRD Completed successfully
```

To clear an override, enter:

```
CICSOVRD NAME=subsys,ACTION=SET,OVERRIDE=%%
```

Examples of Usage

Example 1: Add an Override for the PLTPI for CICS01A: Command:

```
CICSOVRD NAME=CICS01A,ACTION=SET,OVERRIDE=%PLTPI=02%
```

Message Response:

```
EVE556I  CICSOVRD Completed successfully
```

Example 2: Change the Keypoint Option for CICS01A: Command:

```
CICSOVRD NAME=CICS01A,ACTION=SET,KEYPOINT=OPT
```

Message Response:

```
EVE556I  CICSOVRD Completed successfully
```

Automation Policy Commands

The following commands are meant to be executed from the automation policy:

- CICSPURG** Purge transactions at CICS shutdown, see “CICSPURG: Purge Transactions” on page 42.
- CICRSYC** Refresh CICS information at Agent startup. Run from the ACORESTART policy, see “CICRSYC: CICS Resync” on page 43.
- CICSSHUT** Shuts a CICS subsystem down, see “CICSSHUT: Shutdown Processor” on page 44.
- EVEERDMP** Creates a system DUMP of the CICS address space, see “EVEERDMP: CICS Dump” on page 45.
- CMASSHUT** Shuts a CICS CMAS address space, see “CMASSHUT: CICSplex SM Address Space Shutdown” on page 46.

CICSPURG: Purge Transactions

This command will purge running transactions at the time it is issued. It will consult the LISTSHUT MESSAGES/USER DATA policy item (see “LISTSHUT: Transaction Purging During Shutdown” on page 33) to determine which transactions are eligible for purge.

CICSPURG should be placed in either the pre-shutdown policy or as the first phase of the shutdown policy. Under normal circumstances the default action that CICS takes to purge transactions at shutdown will suffice to get the CICS subsystem to shutdown.

Format

CICSPURG [*subsys*]

Keyword and Parameter Definitions

subsys

The symbolic name by which this CICS subsystem is known to SA z/OS.

Comments and Usage Notes

If a subsystem name is not specified, the TGLOBAL SUBSAPPL, which is set by AOCQRY, is used.

Examples of Usage

In this example, CICSPURG is used on the second attempt to shutdown this subsystem.

```
Pass/Selection Automated Function/'*'
Command Text
PASS1 _____
CICSSHUT NORMAL _____

PASS2 _____
CICSPURG _____
```

CICRSY: CICS Resync

The purpose of this routine is to resynchronize CICS information with what is currently operational in the system (such as the VTAM ACB status). If a CICS subsystem should be active, and health checking and link monitoring are defined for this subsystem, CICRSY will activate these monitoring functions. When you define the ACORESTART keyword under the MESSAGES policy item for a CICS application, you must specify CICRSY as the command. For ACORESTART, see *IBM Tivoli System Automation for z/OS Defining Automation Policy*.

The format is:

Format

CICRSY *subsys*

Keyword and Parameter Definitions

subsys

The symbolic name by which this CICS subsystem is known to SA z/OS.

Comments and Usage Notes

If *subsys* is not specified, CICRSY will access task global SUBSAPPL, which will probably not contain the correct value. The resync process may therefore be attempted on the wrong subsystem.

Examples of Usage

In this example, the command is used with the ACORESTART keyword to determine whether or not CICS1 should be active.

```
Pass/Selection Automated Function/'*'
Command Text
CICRSY CICS1 _____
```

CICSSHUT: Shutdown Processor

This is an extended command list that determines whether or not this CICS subsystem is running with XRF, so that the proper shutdown command can be called for the shutdown invocation.

Format

CICSSHUT {NORMAL|IMMED|TAKEOVER|DUMP} [*cicsname*] [SDTRAN=*tranid*|NONE]

Keyword and Parameter Definitions

cicsname

The job name or subsystem name of the CICS. *cicsname* is an optional parameter.

SDTRAN=*tranid*|NONE

tranid is the name of a CICS transaction that is to run at shutdown. The specified transaction overrides the SIT SDTRAN= specification, or the default CICS-supplied shutdown assist transaction CESD. If 'NONE' is specified, it will be translated into 'NOSDTRAN', meaning that no shutdown assist transaction is to run at shutdown.

The SDTRAN= parameter is optional and valid only for CICS TS for OS/390 V1R1 and higher versions. It is ignored for lower releases of CICS.

This routine invokes CICS transactions and the parameters perform the shutdown as described in the CICS operator manuals. If you are running in XRF and the backup system is active, CEBT is used to perform the shutdown. Otherwise, CEMT is used.

Comments and Usage Notes

1. The CEMT PERFORM SHUTDOWN types are passed as parameters to this command.
2. CICSSHUT is recommended for all shutdown policies for subsystems automated by CICS Automation.

Examples of Usage

Pass/Selection Automated Function/'*'

Command Text

PASS1 _____

CICSSHUT NORMAL _____

PASS2 _____

CICSPURG _____

EVEERDMP: CICS Dump

EVEERDMP will create a dump for specific CICS problems. It dumps the associated MVS region using the MVS DUMP command. It can be used for situations such as short on storage conditions if you want an MVS dump instead of a CICS internal dump.

Format

```
EVEERDMP {jobname | subsys}
```

Keyword and Parameter Definitions

jobname

The jobname for this CICS.

subsys

The name by which this CICS subsystem is known to SA z/OS.

Examples of Usage

```
Pass/Selection Automated Function/'*'  
Command Text
```

```
EVEERDMP &SUBSAPPL_____
```

CMASSHUT: CICSplex SM Address Space Shutdown

| This is a command list that determines whether this CICS subsystem is running a
| CICSplex SM Address Space (CMAS).

Format

CMASSHUT [*cmasname*]

Keyword and Parameter Definitions

cmasname

The job name or subsystem name of the CICSplex SM Address Space (CMAS).
cmasname is an optional parameter. If *cmasname* is not specified, the SUBSAPPL
task global value is used.

Comments and Usage Notes

CMASSHUT is intended as a shutdown command that is to be defined as a
shutdown pass in the policy database. It is recommended that this be used to
shutdown CICSplex SM Address Space (CMAS) subsystems.

Examples of Usage

A normal shutdown of the CMAS is requested with the first pass.

Pass/Selection Automated Function/'*'

Command Text

PASS1 _____

CMASSHUT _____

PASS2 _____

MVS C &SUBSAPPL _____

Application Programming Interfaces

The following commands are to be invoked from user REXX programs:

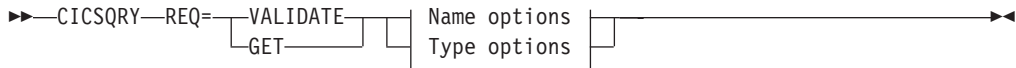
- CICSQRY** Get information about a CICS subsystem, see “CICSQRY: Name Lookup” on page 48.
- CICSRCMD** Route a command to an Agent that is controlling a CICS subsystem, see “CICSRCMD: Request a CICS Function” on page 52.

CICSQRY: Name Lookup

Use this routine to retrieve CICS subsystem information.

Note that CICSQRY does not recognize subsystems that are in FALLBACK or MOVED status.

Format



Name options:



Type options:



Keyword and Parameter Definitions

REQ=

The request type. The request types are:

VALIDATE

A search is made for the name (NAME=) and type (TYPE=) specified so that the name can be validated.

GET

CICS Automation searches for a specific CICS subsystem to retrieve the subsystem characteristics.

GET and VALIDATE are treated as synonyms. They both check the resource and set the Task global variables.

NAME=

Used with VALIDATE to provide a specific group, domain, or jobname. Used with GET to provide a specific subsystem value. Valid values for the NAME= variable are:

subsystem

The name by which a CICS subsystem is known to SA z/OS.

resource

The resource name in the *name/APL/system* format; thus, for example, APG is not accepted as the resource type.

jobname

The job name that a CICS subsystem is known to SA z/OS by.

TYPE=

Used to provide a specific type. The types are:

CICS (default)

Search for a specific CICS subsystem name, as it is known to SA z/OS.

ANY

Search for a CICS name first, then a domain, then a group name. If the name is longer than 5 characters the search for a domain is bypassed.

DOMAIN

The NetView domain ID.

GROUP

If you specify GROUP, CICSQRY returns the name of the group to which the subsystem belongs in the EVELOOKUP_GROUP variable.

JOBNAME

Used with GET to provide a specific job name. Works only when NAME=*jobname*.

Comments and Usage Notes

1. The return codes are:

Table 3. CICSQRY Return Codes

RC	Meaning
0	Good.
4	An internal error occurred.
8	A timeout occurred on a request forwarded to a remote system.
12	An internal error occurred.
20	A subsystem, group, or domain was not found for the search criteria specified.
24	The parameters for this request are invalid.
28	An internal error occurred.
32	Unsupported function.
36	Resource name is ambiguous (more than one resource of the same name exists within the sysplex but none are defined on the local system).
40	System name where the CICS resource resides is not unique within the enterprise.
44	The CICS resource is not unique within the enterprise and its resource tree contains more than one MOVE group (MOVE groups within MOVE groups are not supported by CICSQRY).

2. The following are set in the caller's variable pool:

EVELOOKUP_NAME

Unless TYPE=JOBNAME, set to the value of the NAME= parameter. If TYPE=JOBNAME, set EVELOOKUP_NAME to the subsystem name. Otherwise, set to null.

EVELOOKUP_TYPE

Set to the value of the TYPE= parameter, unless TYPE=ANY or JOBNAME, in which case it is set to CICS or DOMAIN or GROUP as appropriate.

EVELOOKUP_JOBNAME

The jobname associated with the subsystem.

EVELOOKUP_DOMAIN

The NetView domain on which SA z/OS, managing this subsystem, is running.

CICSQRY: Name lookup

EVELOOKUP_AUTOOPS

The NetView automated operator that handles automation for this subsystem.

EVELOOKUP_USERVAR

The VTAM USERVAR (or generic application ID) associated with this subsystem. This is set to '*****' if a VTAM USERVAR is not defined.

EVELOOKUP_APPLID

The specific VTAM application ID associated with this subsystem.

EVELOOKUP_RESHOME

The location of the resource in the following format:

sysplex.domain.system\VxRyMz

EVELOOKUP_RESLIST

The resource name in the following format

name/type/system

EVELOOKUP_AGENTDATA

Information about the agent responsible for the subsystem in the following format

agent_name sysplex_name system domain agent_version [netview_version]

EVELOOKUP_GROUP

The name of the group(s) to which the resource belongs.

Restrictions

The CICSQRY command can only be called in a REXX exec.

Examples of Usage

Example 1:

```
CICSQRY REQ=GET NAME=CICSAPL1
```

For RC=0 the following Task Globals are set:

Variable	Value
<i>evelookup_reslist</i>	CICSAPL1/APL/SYS1
<i>evelookup_reshome</i>	TESTPLEX.IPSFM.SYS1 \ V3R1M0
<i>evelookup_AgentData</i>	IPSFM TESTPLEX SYS1 IPSFM V3R1M0 V5.1
<i>evelookup_Name</i>	CICSAPL1
<i>evelookup_Domain</i>	IPSFM
<i>evelookup_autoops</i>	AUTWRK01
<i>evelookup_jobname</i>	EYUMAS1M
<i>evelookup_applid</i>	IPSAMC1M
<i>evelookup_uservar</i>	*****
<i>evelookup_majnode</i>	KEY1CICS
<i>evelookup_type</i>	CICS

Example 2:

```
CICSQRY REQ=GET NAME=CICSAPL1 TYPE=GROUP
```

For RC=0 the following Task Globals are set:

Variable	Value
<i>evellookup_group</i>	CICS/APG/SYS1
<i>evellookup_type</i>	GROUP

CICSRCMD: Request a CICS Function

This common routine is used to perform the requested function (CMD=) on the domain where the named CICS subsystem resides, whether local or remote. The calling program does not have to be aware of where the CICS subsystem resides. It is particularly useful with single point of control as CICSRCMD first determines the domain in which the subsystem resides before building and issuing the request. It then either calls the requested function if the subsystem is on the local domain, or it forwards the command to the remote domain, thus allowing cross-domain communications.

Format



Keyword and Parameter Definitions

NAME=

The name by which the target CICS subsystem is known to SA z/OS.

RESP=

Send back a response (YES) or just send an acknowledgment (ACK).

OPER=

The operator, on the target domain, that will execute this command. If this is omitted, the assign-by-jobname automation operator is used.

CMD=

The requested function to be performed. This may be delimited by single quotes, double quotes, or slashes.

Comments and Usage Notes

This command will work sysplex-wide. It is enterprise-wide when invoked on a focal point agent with a fully qualified resource name, that is, subsystem/APL/system.

Table 4. CICSRCMD Return Codes

RC	Meaning
0	Good.
4	Subsystem name was not supplied.
8	Function to be performed was not supplied.
12	Incorrect keyword supplied.
20	Subsystem was not found on any domain.

Part 3. Using CICS Automation

This part describes the tasks of the operator who manages CICS subsystems through CICS Automation. It contains the following chapters:

- Chapter 6, “Working with CICS Resources,” on page 55
- Chapter 7, “The Status Display Facility,” on page 65

Chapter 6. Working with CICS Resources

This chapter explains how to use the CICS Automation panels and how to work with subsystems. It assumes that you are familiar with the SA z/OS operator interface. This chapter describes the characteristics of CICS Automation. To thoroughly understand your role as the CICS Automation operator, some hands-on experience with SA z/OS is useful.

Using CICS Automation Panels

This section explains how to work with the CICS Automation main panel. To start a CICS Automation operator session and display the CICS Automation Main Menu, enter CICS on a NetView command line.

Using the Main Menu

The main menu lists all tasks available with the operator interface.

```
EVEK0000          SA z/OS - Command Dialogs
Domain ID  = IPSFM  ----- CICS -----   Date = 06/20/07
Operator ID = NETOP1      System = KEY1      Time = 11:18:27

Resource => _____ Format: name/type/system
System   => _____ System name, domain ID or sysplex name

Action => 1. Inquire          Display CICS Information
          2. Start           Start a CICS subsystem
          3. Shutdown        Shutdown a CICS subsystem
          4. Triggers         Display trigger conditions
          5. Service Periods  Perform scheduling functions
          6. Master Terminal  Perform master terminal commands
          7. Monitoring       Perform monitoring functions
          8. Broadcast        Send messages to users

          99. Local functions  Provide access to user defined local
                               functions

Command ==>
PF1=Help      PF2=End      PF3=Return      PF6=Ro11
               PF12=Retrieve
```

Figure 3. CICS Automation Main Menu

The following describes the options you can select on the main menu:

1. Inquire

This option invokes the INGCICS REQ=INFO command which will execute a preset sequence of commands and display the results.

The preset sequence of commands must all be CEMT commands and they are defined in the subsystem's CICSINFO MESSAGES/USER DATA policy item.

2. Start

This option initiates the startup process of a resource. By choosing this option you issue the INGREQ command. See "Startup" on page 56.

Using CICS Automation Panels

3. Shutdown

This option initiates the shutdown process of a resource. By choosing this option you issue the INGREQ command. See “Starting and Stopping Resources.”

4. Triggers

This option displays the triggers associated with a resource. By choosing this option you issue the DISPTRG command. See *IBM Tivoli System Automation for z/OS Operator's Commands*.

5. Service Periods

Use this option if you want to display or override the schedule associated with a resource. By choosing this option you call the INGSCHED command. See *IBM Tivoli System Automation for z/OS Operator's Commands*.

6. Master Terminal

This option invokes the INGCICS REQ=CMD command which allows you to enter a command to be executed on the CICS subsystem. The output of the commands will be displayed.

7. Monitoring

Use this option to work with link monitoring and health checking. See “Monitoring Your CICS Subsystems” on page 59.

8. Broadcast

This option invokes the INGCICS REQ=BROADCAST command which allows you to specify the parameters for the CMSG transaction. It will then execute the transaction and return the results to the display.

99. Local Functions

CICS Automation allows your system programmer to add functions to this operator interface. If functions have been added at your installation, you would select this option to view a menu of them.

Note: The options 7 and 99 are only valid for the local sysplex. You cannot access a remote sysplex with any of these functions.

Starting and Stopping Resources

CICS Automation uses the INGREQ command of SA z/OS for starting and stopping resources. For information on INGREQ, see *IBM Tivoli System Automation for z/OS Operator's Commands*. The following describes things to consider when starting or stopping a CICS resource.

Startup

If you select option 2 Startup on the main menu, the INGREQ command dialog is invoked.

The CICS-specific features apply to the Type, Override, and Appl Parm fields:

Type In this field, you can specify the start type. The possible values depend on the CICS version as follows:

Start Type	Explanation	Valid for
AUTO	Uses the restart data set to determine the startup type.	All releases
COLD	Initiates a cold start.	All releases

Start Type	Explanation	Valid for
INITIAL	Initiates a cold start with additional processing for CICS TS resources.	CICS TS V1R1 and above
NORM	This is the same as AUTO.	
STANDBY	Initiates a start for an XRF backup.	All releases

The SA z/OS-supported start types may not be meaningful for some releases of CICS. In addition the NORM start type is the default start type that is used by SA z/OS. This can be useful when you want to COLD start CICS normally, but AUTO start it after an abend.

If CICS is set up to prompt with message DFHPA1104 during startup, to indicate that it is ready to read parameters from the console, then System Automation will reply with START=AUTO whenever it comes to the conclusion that a start type of NORM or AUTO is to be performed.

If you want to see the startup types that have actually been defined for the subsystem to be started, enter a question mark in the Type field and press ENTER. You will see a panel like the following:

```

AOFKSEL3          SA z/OS - Command Dialogs          Line 1 of 4
Domain ID = IPSFM  ----- INGREQ -----          Date = 05/03/00
Operator ID = SCHR                                     Time = 12:34:12

The following start types are defined for CICS1H/APL/KEY1
Select one item to be processed, then press ENTER.

      Sel  Start types
      ---  -----
      -    AUTO
      -    COLD
      -    INITIAL
      -    NORM

Command ==>>
PF1=Help    PF2=End    PF3=Return
PF6=Roll

                                           PF12=Retrieve
    
```

Enter s in the **Sel** column to select the desired type.

Note: When you select a startup type that is valid for CICS, but has not been defined in the STARTUP policy item of the target resource, INGREQ issues the command defined for the NORM startup type in the STARTUP item. If that entry does not exist either, the command MVS START *jobname* is issued.

Override

The following values are CICS specific:

Value	Condition overridden
INIT	This override allows you to select a start type other than INITIAL although an INITIAL start has been requested for CICS.

Starting and Stopping Resources

Value	Condition overridden
UOW	This override allows you to select the INITIAL or COLD startup type, although <ol style="list-style-type: none"> 1. The field Keypoint req of the CICS CONTROL policy item is set to YES in the policy database for this application, AND 2. Indoubt units of work were detected during the last shutdown of CICS, or no warm keypoint was taken during the last execution.

Appl Parm

You can enter overrides to the system initialization table (SIT) in the following format:

KEYWORD=value

For details see the CICS documentation.

Note: To use this function, your CICS must be set up to allow SIT overrides input from the console.

You can specify more than one parameter in this field. The entries must be separated by a blank or a comma.

If you have not changed the default value of YES for the **Verify** field, CICS Automation will display a verification panel (see Figure 4) after you have pressed ENTER. This panel displays the target resource and in addition all the resources which SA z/OS will try to start because the startability of the selected resource directly or indirectly depends on them.

```

AOFKVFY1          SA z/OS - Command Dialogs          Line 1 of 3
Domain ID = IPSFM ----- INGREQ -----          Date = 05/03/00
Operator ID = SCHR                                     Time = 12:34:11

Verify list of affected resources for request START

CMD: S show overrides  T show trigger details  V show votes
Cmd Name      Type System  TRG SVP  W  Action Type  Observed Stat
-----
CICSK1H      APL  KEY2                Y      AUTO  UNAVAILABLE
JES2         APL  KEY2                AUTO  UNAVAILABLE
VTAM         APL  KEY2                AUTO  UNAVAILABLE

Command ==>
PF1=Help  PF2=End  PF3=Return          PF6=Ro11
PF10=GO   PF11=CANCEL  PF12=Retrieve
  
```

Figure 4. Verification Panel for INGREQ

For more information on the verification panel of INGREQ, see *IBM Tivoli System Automation for z/OS Operator's Commands*.

Shutdown

When you select option 3, Shutdown, from the main menu panel, the INGREQ panel is invoked.

Monitoring Your CICS Subsystems

CICS Automation provides two functions to monitor CICS subsystems:

- Link monitoring
- Health checking

Link Monitoring

SA z/OS provides an enhanced monitoring infrastructure that makes use of existing base components, especially monitor resources, with the following benefits:

- Event-based monitoring rather than timer-based monitoring reduces the system load
- Monitor resources make it possible, for example, to have direct relationships to the CICS APLs
- Using monitor resources further makes it possible to have customizable recovery actions from the MESSAGES/USER DATA or HEALTHSTATE policy item, or both

It should also be noted that the status is no longer displayed in CICS panels (transaction COLO) and that the status of link monitoring is displayed in SDF as MTRs rather than on the CICS product automation panel.

Enhanced Monitoring Infrastructure

A monitor resource is expected to be monitoring a real-world object, such as a file system, the CPU or a CICS link. Previously the monitor resource was not directly related to such an object. INGMON is used to set the health status from any information that is available to it, and this is the basis for the enhanced monitoring infrastructure.

The MONITOR INFO policy item for a monitor resource contains a **Monitored Object** field that describes which object this monitor resource is being used for. This field is not interpreted by SA z/OS and can have any semantic the monitoring process that might be needed.

For CICS monitoring the **Monitored Object** field begins with a customizable prefix (for example, CPSM) followed by the CICS name, the type (for example, a connection) and the name (for example, CPSM.CICSTOR1.CONNECT.CT12). For OMEGAMON the field might contain the session name and the exception type (for example, SESS1/CPUB). Each monitor resource must have exactly one object assigned to it, but an object can be referenced by several monitor resources (for example, multiple IMS monitors might specify OLDS as an object).

Event-Based Monitoring

In CICS there are real-world objects such as ISC or IRC links that should be monitored. Their status should be reflected in the health status of monitor resources. This can be done with an external monitor like CICSplex SM. Using CICSplex SM also allows SA z/OS to monitor approximately 40 additional objects, such as CICS journals or DB2 threads.

SA z/OS provides CICS-dependent code that subscribes to and listens for events produced by CICSplex SM. To be able to subscribe for the objects of interest, the

Monitoring Your CICS Subsystems

configuration is read and the objects defined in the MTRs are interpreted. Whenever an event is received from CICSplex SM, INGMON is called with the object in trouble and the status as reported by the external monitor. INGMON maps this status to a valid health status, updates the affected MTRs, and triggers recovery action if it has been defined in the policy. Initial monitoring also triggers INGMON to set the initial status.

Because the listening task is a never-ending job, SA z/OS uses an independent virtual OST (VOST) that is attached to the auto task. The auto task itself attaches the task and is immediately free for other work.

Component Overview

Figure 5 gives an overview of the components that are involved in the enhanced monitoring infrastructure.

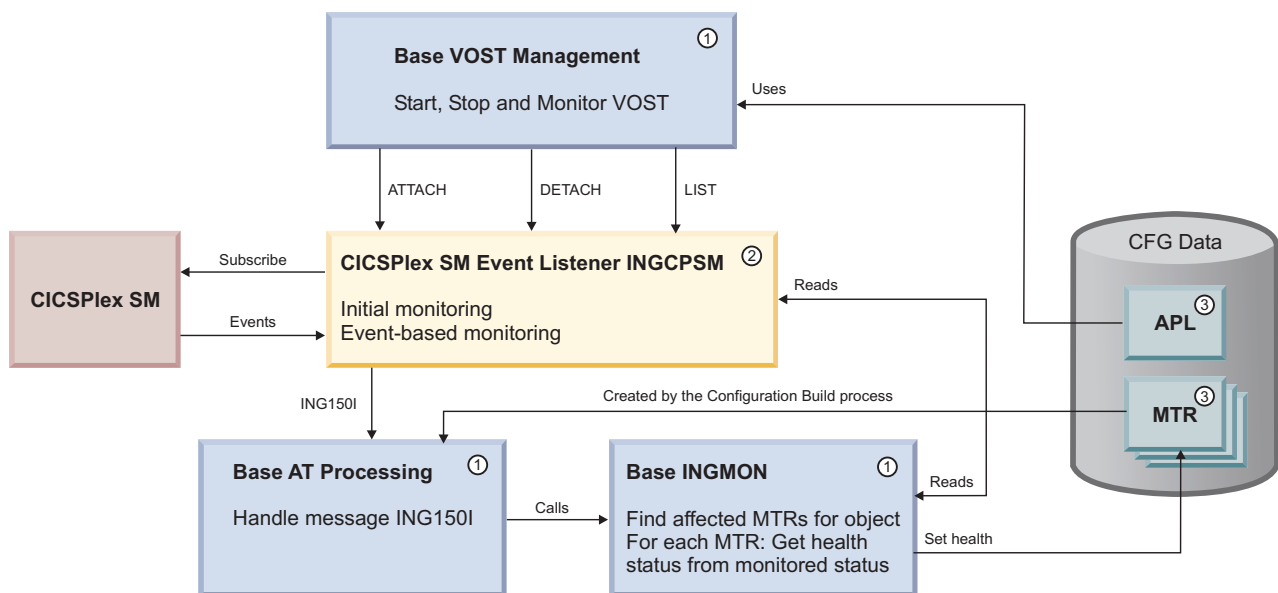


Figure 5. Event-based Monitoring Infrastructure

The different components are numbered as follows:

1. Base elements
2. CICS-dependent
3. Configuration data.

Message ING150I is introduced as generic interface to inject monitoring events into SA z/OS.

INGMON is triggered from the NetView automation table for ING150I. INGMON automatically finds the affected monitor resources by scanning the configuration. It finally sets the health status of the affected monitor resources.

Whenever an event is received from CICSplex SM, it emits an ING150I message.

Because the event listener is a long running CLIST it is executed in a VOST rather than blocking a work operator. VOST management routines are supplied by the base system.

The event listener scans the configuration on startup and performs a LISTEN for events. It periodically checks whether the configuration has changed (that is, monitor resources have been added, deleted, changed, etc) or monitor resources are waiting for initial monitoring (that is, STATUS=ACTIVE and HEALTH=UNKNOWN). For initial monitoring the CICSplex SM data tables are read and an ING150I message is produced.

Setting Up the Monitor Resources

To set up the monitor resources in the customization dialogs, perform the following steps:

1. Create one MTR for each CPSM object that you want to monitor (for example, each connection).
2. Enter the monitored object according to your naming conventions (for example, CPSM.CICSTOR1.CONNECT.CT12).
3. Define code processing for the message ID ING150I in the monitor resource's MESSAGE/USER DATA policy item to map the CPSM severities to valid health states. Use the first word of the Value Returned field to define a recovery command to be executed. For example:

Code 1	Code 2	Code 3	Value Returned
VHS			NOP CRITICAL
*			NOP NORMAL

The first word (NOP) in Value Returned specifies the selection of recovery command to be executed.

INGMON Routine

INGMON is triggered by message ING150I. The monitored object is passed together with the monitored status as the CODE1 parameter.

INGMON checks the configuration and finds the monitor resource with the given monitored object.

Job name checking is not performed because a monitored job name is not entered for the monitor resource.

Code matching is done to match the monitored status (CODE1) and to find a selection of commands to be performed and a valid health status.

If commands are found for the selection returned by CDEMATCH they are executed.

Finally the health status returned by CDEMATCH is applied to the monitor resource.

The CICSplex SM Listener

INGCPSM is the CICSplex SM listener. It connects to the local CMAS and performs initial monitoring. It reads the EVENT data table and extracts all entries with the specified ACTION. For each event a ING150I message is built that contains the monitored object and the monitored status. These messages are handled by INGMON from the AT.

Next INGCPSM subscribes to the EACTNTFN data table and enters the "receive loop. Within this loop it checks for uninitialized monitor resources (that is, monitor resources that became active after INGCPSM was started) and provides initial

Monitoring Your CICS Subsystems

monitoring for them by checking the EVENT data table as mentioned above. It then receives all events available by the current subscription. ING150I messages are produced for each event.

INGCPSM is a never-ending task. It executes the receive loop until the task is stopped. In order to prevent a work operator being blocked INGCPSM runs in a virtual OST (VOST).

Health Checking

Through health checking, you execute programs that check the health of an application running under CICS. CICS Automation initiates a health check program by sending a request across the program-to-program interface to CICS. CICS invokes the program, and sends back an Acknowledgement (ACK), indicating that the program executed as expected, or it sends a Negative Acknowledgement (NACK), indicating that the program did not execute as expected. A NACK includes data describing the error.

Because health checking is application specific, the actual health checking programs must be written by programmers in your installation. CICS Automation does, however, provide some samples for system programmers in the CICS Automation source library.

Health check routines are executed automatically at timed intervals. When CICS Automation receives an abnormal response from CICS (NACK) or when a timeout occurs, CICS Automation sends out notification messages. CICS Automation panels can also be used to manually work with health checking.

Select option 2, Health checking, from the CICS Automation Monitoring panel to display the following:

```

EVEKM200          SA/CICS - Health Checking          Page: 1 of 1
                                                         Date: 01/09/02
Resource . . . . . CICS2      (? for list)          Time: 10:20

Select a command:  1. Start          3. Suspend          5. Detail
                  2. Stop           4. Resume           6. Immed check

CMD  Program  Description          Status   Date   Time   Response
-   HCPAY1   CHECK PAYROLL DATA BASE  ACTIVE   01/09/00 10:32:00 ABNORMAL
-   HCEDI5   ACCESS TO EDITOR          INACTIV  01/06/00 09:00:00 NORMAL
-   HCFULL   95% FULL CONDITION       ACTIVE   01/09/00 11:30:05 NORMAL
-   HCACTV   95% ACTIVE CONDITION     ACTIVE   01/09/00 09:09:59 NORMAL
-   HCAREC   ACCOUNTS RECEIVABLE     ACTIVE   01/09/00 11:33:00 ABNORMAL
-   HCTERMA  95% TERMINALS ACTIVE    INACTIV  01/06/00 09:00:00 NORMAL
-   HCOU1    ACCESS TO OUTMAIL FILE   ACTIVE   01/09/00 11:33:10 ABNORMAL
-   HCIN1    ACCESS TO INMAIL FILE    INACTIV  01/06/00 09:10:00 ABNORMAL
-   HCPAY2   PAYROLL SUBMIT           ACTIVE   01/09/00 08:32:55 NORMAL
-   HCTERM2  REMOTE TERMINAL PROGRAM  INACTIV  01/03/00 08:00:00 NORMAL

Command====>
F1=Help      F2=End      F3=Return   F4=CICS menu  F5=Refresh   F6=Ro11
    
```

Figure 6. Health Checking Panel

The Health Checking panel lists the health check routines for a particular subsystem. The list includes:

- The program name

- The description
- The status (ACTIVE, INACTIV, or SUSPEND, which shows whether automatic execution is active)
- A time stamp of the last time the routine ran
- The response (NORMAL or ABNORMAL).

From this panel, you can:

- 1. Start** Initiate automation of a routine
- 2. Stop** Stop automation of a routine
- 3. Suspend** Temporarily stop automation execution of a routine
- 4. Resume** Continue with a process that was temporarily stopped
- 5. Detail** Expand upon the given information to show the details of a health check routine
- 6. Immed check** Immediately submit a status check regardless of status and scheduled time interval

Broadcasting Messages

INGCICS REQ=BROADCAST allows you to send messages to any user of the CICS subsystem. Figure 7 shows a sample Broadcast Messages panel:

```

EVEKYCMD                    SA z/OS - Command Dialogs            Line
Domain ID   = IPSFM        ----- INGCICS -----            Date = 08/30/02
Operator ID = NETOP1                                            Time = 18:45:43

Resource        => CICSK1H/APL/KEY1                            Format: name/type/system
System          =>                                               System name, domain ID or sysplex name
Request         => BROADCAST                                    CMD, BROADCAST or INFO
CICS Transaction => MSG R=&ROUTE,'&MESSAGE',HEADING=YES,S
CICS Route      => ALL
CICS Message   =>
                                                                 =>

AOF145I PARAMETER MISSING
Command ==>
         PF1=Help    PF2=End            PF3=Return    PF4=DISPINFO            PF6=Ro11
                                                                 PF9=Refresh            PF12=Retrieve
    
```

Figure 7. Broadcast Messages Panel

The &ROUTE and &MESSAGE items are placeholders for the information in the Route and Message lines. The CICS transaction field is automatically filled with the text above. You can make changes to it before pressing Enter to execute the transaction. Specify the routing information in the route field - routing information format is the same as specified for the CMSG transaction. Specify the message in the two message fields provided.

Broadcasting Messages

Chapter 7. The Status Display Facility

The Status Display Facility uses color to represent the various subsystem resource statuses such as error, warning, action, or informational states. Typically, a subsystem shown in green on a Status Display Facility status panel indicates that it is up, whereas red indicates a stopped or problem state.

The Status Display Facility status display panels can be tailored to present the status of system components in a hierarchical manner. The hierarchical display of status information is implemented using tree structures. A tree structure always starts with the system name as the root component. The “leaves” of the tree are the monitored resources.

Color can be propagated up or down the leaves of the tree structure based on the order of dependencies. The effect of propagation is to consolidate, at the root component, the status of all the monitored resources in that system. In this way, the color of the root component reflects the most important or critical status in a computer operations center. If all the monitored resources are green, the root component (the system) will be green.

CICS Automation provides additional Status Display Facility panels that monitor events that occur in the following areas for all CICS regions defined to CICS Automation:

Health

Messages associated with health checking are routed to the Status Display Facility health checking panels.

CICS Storage

Storage violation messages are routed to the Status Display Facility storage panels.

CICS Transaction

Messages associated with transactions are routed to the Status Display Facility transactions panels.

| A special set of SDF sample panels is provided for CICS Automation. See *IBM*
| *Tivoli System Automation for z/OS Customizing and Programming* for more details.

Appendix. CICS Automation and the Program-to-Program Interface

Warning!

CICS Automation uses the program-to-program interface for health checking. If you are considering using the CICS Automation program-to-program interface code for your own purposes, remember that this interface is release-sensitive. The significance of this is that you may need to recompile or make changes to your code to be compatible with new releases of CICS Automation or NetView.

This appendix provides details about the NetView program-to-program interface, as follows:

- “Program-to-Program Interface Components in NetView and CICS”
- “NetView Requests Using the Program-to-Program Interface” on page 68
- “CICS Requests Using the Program-to-Program Interface” on page 72
- “Programming Interface” on page 74
- “Customizing CICS Definitions” on page 90
- “Definition Members” on page 90
- “Security Checking Using CICS” on page 93

Program-to-Program Interface Components in NetView and CICS

Figure 8 on page 68 illustrates CICS Automation program-to-program interface components in NetView and in CICS.

NetView Requests Using the Program-to-Program Interface

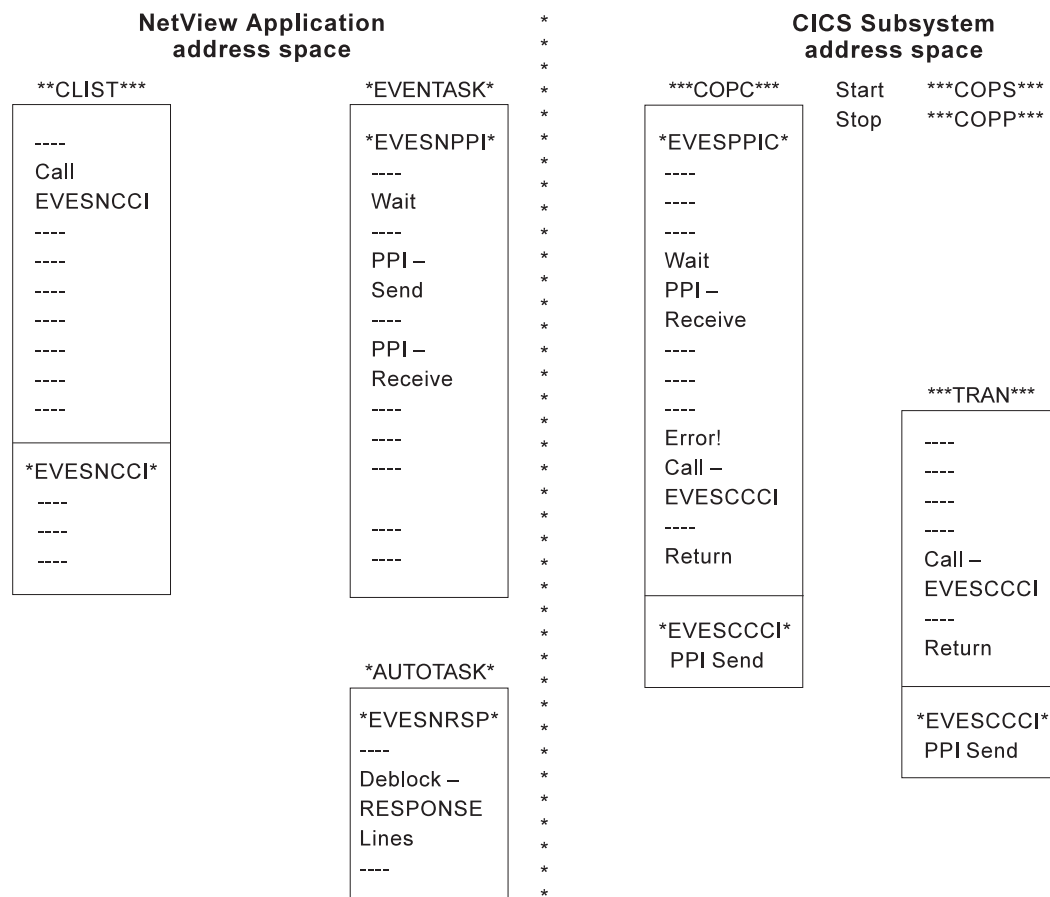


Figure 8. Program-to-Program Interface Components in NetView and CICS. Two of the programs shown in this figure have not been previously mentioned: EVESNPPI and EVESPPIC. EVESNPPI is the NetView subtask program. EVESPPIC is the CICS long-running receiver program. These are internal programs and are not described in this manual.

NetView Requests Using the Program-to-Program Interface

The following requests from NetView are described in this section:

- CONVERSE
- SEND
- CANCEL

EVESNCCI is used for these requests. This section only provides an overview of how EVESNCCI works. The programming details, such as command syntax, return codes, and segment support, are provided in "EVESNCCI: NetView to CICS Communication Interface" on page 76.

CONVERSE from NetView

A CONVERSE request from a NetView command list (or command processor) starts a CICS transaction on the same host. The CICS transaction is expected to return a response to a specified NetView task in a named NetView domain. The response can have the form of a RESPONSE, an ACK, or a NACK.

The EVESX001 (CICSSEC) and EVESX022 (CICSQRY) command processors are called to verify the authorization of the caller and to obtain the VTAM application identifier, which is used as the program-to-program interface receiver

NetView Requests Using the Program-to-Program Interface

identification. EVESNCCI returns a message and a return code indicating whether the command was successful or not. If the command is accepted, it is forwarded to the EVENTASK optional task.

The EVENTASK optional task uses the program-to-program interface to notify CICS that the transaction is to be started.

The started transaction retrieves the input data. The CICS transaction returns a response to NetView. The response is sent back to the EVENTASK optional task using the program-to-program interface. The processing of the response sent to NetView differs for the various response types, as described below:

A for ACK

The following message is sent to the requestor:

```
EVE128I POSITIVE ACKNOWLEDGEMENT
```

N for NACK

The following message is sent to the requestor:

```
EVE129E response-data
```

where *response-data* is the data returned by the CICS program.

R for RESPONSE

When this request type is used, the NetView program-to-program interface initialization member is interrogated to locate the function requested so that the command list can be identified. The request is then scheduled under the autotask associated with the requested function, after which the data is sent to the requestor.

The following figure illustrates the flow of events initiated by an EVESNCCI CONVERSE request:

NetView Requests Using the Program-to-Program Interface

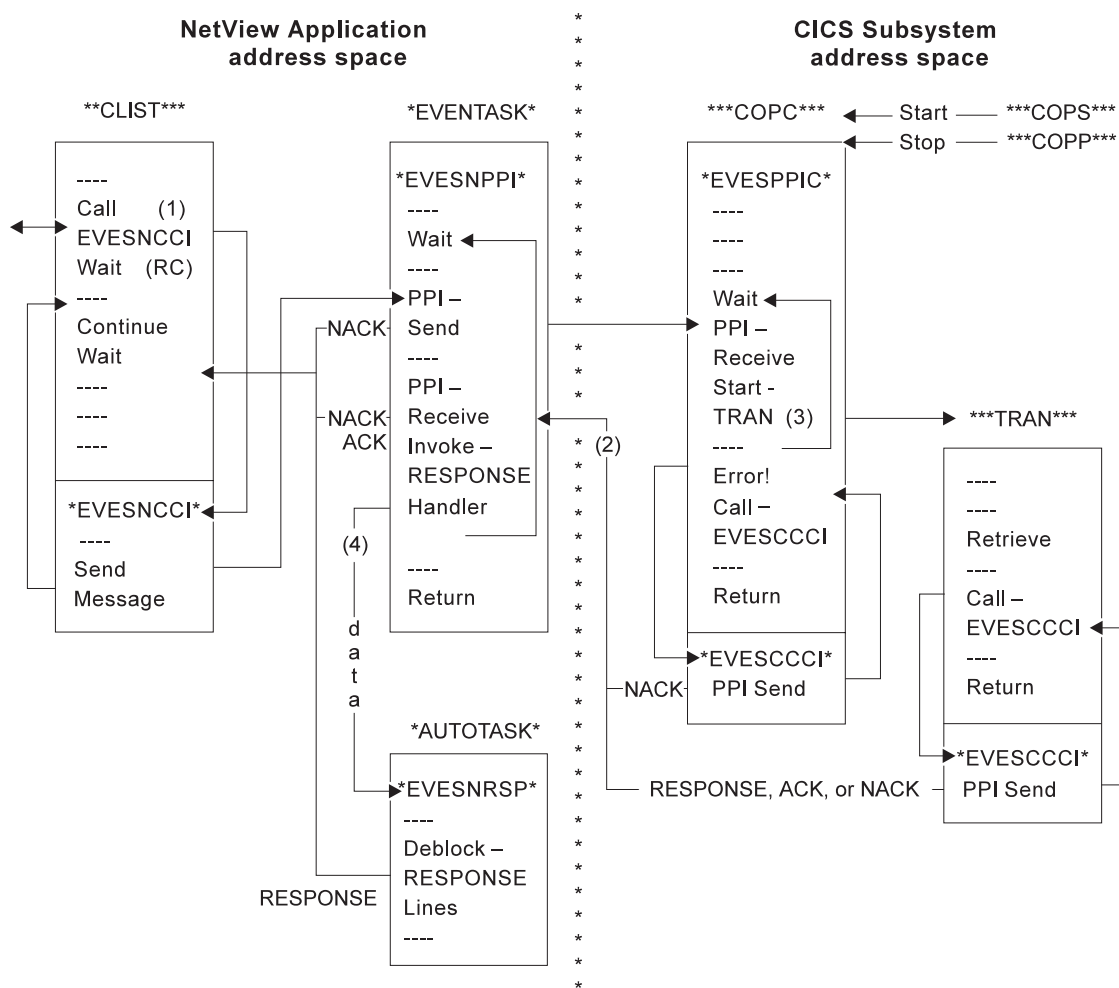


Figure 9. An EVESNCCI CONVERSE Request

Notes:

1. Parameters passed with EVESNCCI indicate the request type (in this case C for CONVERSE), the target CICS, the name of the function to be invoked, data (if required), and the requesting operator ID and domain ID.
2. The responses (RESPONSE, ACK, or NACK) are returned from CICS to the requestor (operator ID and domain ID).
3. The function name is used to locate the transaction name in the CICS initialization member. If the transaction name cannot be found, a NACK response is returned.
4. The name of the autotask and the name of the command processor or command list in NetView are obtained using the function name in the NetView initialization member. If the name or names cannot be found, or if the scheduling of the autotask fails, a NACK response is returned.

SEND from NetView

A SEND request from a NetView command list starts a CICS transaction on the same host. No response is returned. The EVESX001 (CICSSEC) and EVESX022 (CICSQRY) command processors verify the authorization of the caller and obtain the VTAM application identifier, which is used as the program-to-program interface receiver identification. EVESNCCI returns a message and return code

NetView Requests Using the Program-to-Program Interface

indicating whether the command was successful or not. If the command is accepted, it is forwarded to the EVENTASK optional task. This task uses the program-to-program interface to notify CICS to start the transaction.

Errors found by EVESNCCI are returned to the caller. Other errors detected during the processing of a SEND request are not returned. These errors are only logged.

Refer to the following figure:

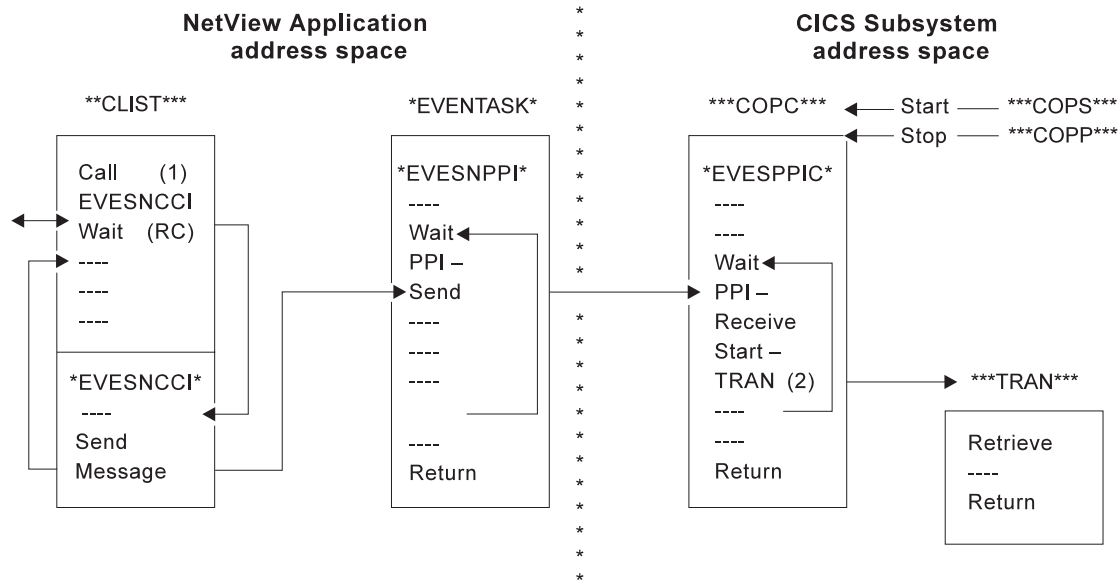


Figure 10. An EVESNCCI SEND Request

Notes:

1. Parameters passed with EVESNCCI indicate the request type (in this case S for SEND), the target CICS, the name of the function to be invoked, and data (if required).
2. The function name is used to locate the transaction name in the CICS initialization member.

CANCEL from NetView

The maximum amount of text that can be specified on the EVESNCCI command is limited to 240 bytes, including the command name EVESNCCI. The maximum amount of data that can be specified varies with the particular EVESNCCI command, but it is always less than 240 bytes. The CICS Automation program-to-program interface implementation provides segment support which allows up to 32656 bytes to be sent from NetView to another program-to-program interface receiver. Segment support is described in "EVESNCCI: NetView to CICS Communication Interface" on page 76 (refer to the SEGMENT= keyword and the usage notes).

The EVESNCCI CANCEL request is used when the segment assembly process must be terminated. This request type causes all saved segments for the specified segment identifier to be freed. If the command is accepted, it is always successful, whether saved segments with the specified segment identifier exist or not.

NetView Requests Using the Program-to-Program Interface

Users of the segment function are requested to use CANCEL when applicable. This prevents the NetView application system from being filled with unused data.

Errors found by EVESNCCI are returned to the caller. Each CANCEL is logged.

CICS Requests Using the Program-to-Program Interface

The following requests from CICS are described in this section:

- CONVERSE
- SEND

EVESCCCI is used for these requests. This section only provides an overview of EVESCCCI. The programming details, such as command syntax and return codes, are provided in “EVESCCCI: CICS to NetView Communication Interface” on page 82.

CONVERSE from CICS

A CONVERSE request from a CICS transaction starts a command list or a command processor in the NetView application system on the same host. The command list or command processor is expected to return a response to the CICS transaction. The response can have the form of a RESPONSE, an ACK, or a NACK.

If the request is accepted, EVESCCCI sends the request to the EVENTASK optional task. The EVENTASK optional task translates the function specification to a command list or command processor name using the EVENTASK initialization member. The autotask under which the command list or processor is to be executed is also obtained from the EVENTASK initialization member.

Errors found by EVESCCCI are returned to the caller. When other errors are detected during the processing of a CONVERSE request, the CICS Automation uses the program-to-program interface to return a NACK response to the CICS transaction. The NACK response data indicates the type of error that occurred. The NACK response is logged via message EVE180E.

Refer to Figure 11 on page 73.

CICS Requests Using the Program-to-Program Interface

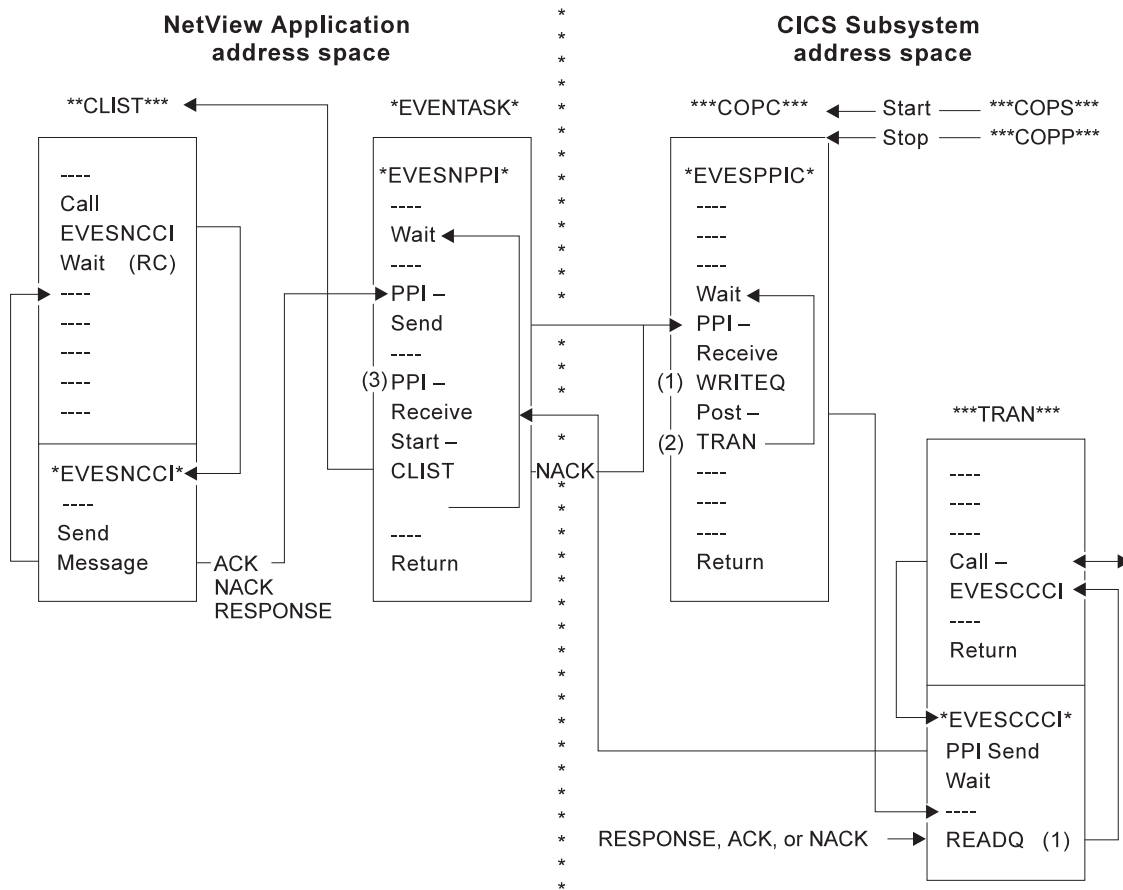


Figure 11. An EVESCCCI CONVERSE Request

Notes:

1. The received data is saved and obtained from temporary storage.
2. The Post-tran is actually a CANCEL of an interval control request.
3. The EVENTASK optional task translates the function specification to a command list or command processor name using the EVENTASK initialization member. The autotask under which the command list or processor is to be executed is also obtained from the EVENTASK initialization member. If the name cannot be obtained or if the scheduling of the command list fails, a NACK response is returned.

SEND from CICS

A SEND request from a CICS transaction starts a command list or a command processor in NetView. No response is returned by the command list or command processor to the CICS transaction.

If the request is accepted, EVESCCCI sends the request to the EVENTASK optional task. The EVENTASK optional task translates the function specification to a command list or command processor name using the EVENTASK initialization member. The autotask under which the command list or processor is to be executed is also obtained from the EVENTASK initialization member. Then, the NetView command processor or command list returns to NetView.

CICS Requests Using the Program-to-Program Interface

Errors found by EVESCCCI are returned to the caller via a return code and an error code. Other errors detected during the processing of a SEND request are not returned to the requestor. The errors are only logged.

Refer to the following figure:

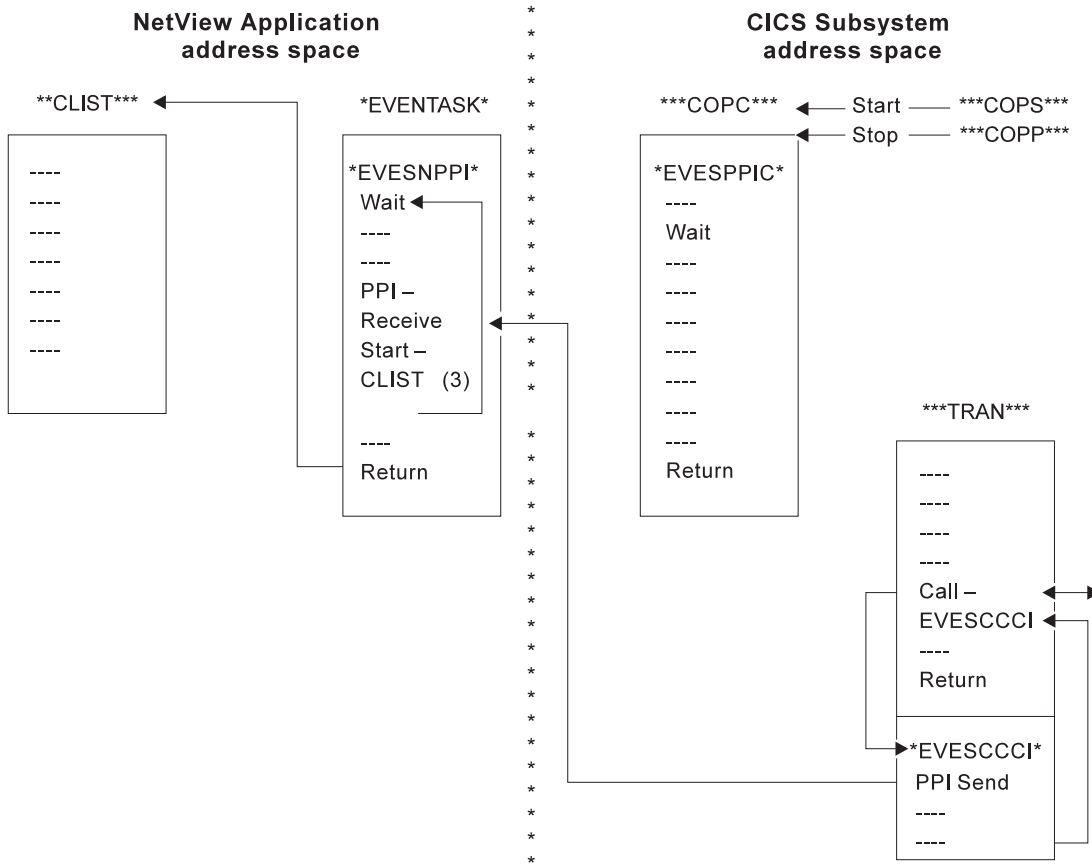


Figure 12. An EVESCCCI SEND Request

Programming Interface

The CICS Automation CICS to NetView program-to-program interface members are:

“EVESNCCI: NetView to CICS Communication Interface” on page 76

Allows you to:

- Initiate, from NetView, the execution of a CICS transaction.
- Send a response to a CICS transaction.

“EVESNRSP: Common Response Handler from CICS” on page 81

The maximum length of a RESPONSE response line is 216 characters. EVESNRSP unblocks the response data in NetView and turns it into a multi-line WTO by calling the EVESX002 (CICSBMMSG) command processor for each response line.

“EVESCCCI: CICS to NetView Communication Interface” on page 82.

Use this subroutine to request, from a CICS transaction, the initiation of a NetView command, a command list, or command processor in NetView.

“EVEMPINT: EVESCCCI Parameter List Copy Book” on page 85.

The EVEMPINT copy book describes the CICS Automation program-to-program interface parameter list on the CICS side.

EVESNCCI: NetView to CICS Communication Interface

Use this subroutine to:

- Initiate, from NetView, the execution of a CICS transaction.
- Send a response to a CICS transaction.

Syntax

```
EVESNCCI TYPE=C|S|R|A|N|X  
         ,NAME=subsnm  
         ,FUNC=function  
         [,DATA=data]  
         [,REQID=reqid]  
         [,SEGMENT=segment]  
         [,ID=id]
```

Keyword and Parameter Definitions

TYPE=

Specifies the type of command. Valid command types are:

- C** For CONVERSE: Starts a process at the other end. A response is expected.
- S** For SEND: Starts a process at the other end. No response is expected.
- R** For RESPONSE: This is used to send data to the CICS transaction that issued the CONVERSE request.
- A** For ACK: Signals the successful completion of a CONVERSE request. No data is provided.
- N** For NACK: Signals the unsuccessful completion of a CONVERSE request. Data can optionally be provided (maximum of 100 bytes).
- X** For CANCEL: Use this request type to cancel the segment assembly process.

NAME=

Specifies the CICS subsystem name as it is known to CICS Automation.

FUNC=

Specifies the symbolic name of a process to be initiated or to be responded to, such as a CICS transaction or a NetView command list. The relation between a function name and a process name is contained in the EVENTASK and EVESPINM initialization members.

DATA=

Specifies the request or response data. Not accepted for ACK responses. The maximum data length on a NACK response is 100 bytes. If omitted, no data is passed.

Three data delimiter pairs are supported: single quotes, double quotes, or parentheses. These delimiters must be used when the data contains blanks or commas, or starts with one of the data delimiters itself. Data delimiters are stripped off before passing the data on.

REQID=

Specifies the request identification. This field relates the response to the CICS transaction requesting the response. The value is provided on the CICS

EVESNCCI: NetView to CICS Communication Interface

CONVERSE request and should simply be copied. The request identification may not contain commas or blanks. It is required and only accepted for responses.

SEGMENT=

Use this keyword when the EVESNCCI command must be longer than 240 bytes. The SEGMENT= parameters are:

- F** First segment.
- M** Neither the first and nor the last segment.
- L** Last segment.

When the SEGMENT= keyword is used, the ID= keyword is used to identify the segment chain within a NetView task.

ID=

Identifies the segment chain within a NetView task. This data field is 16 bytes. The segment chain identification may not start with DSI or EVE and may not contain commas or blanks.

Comments and Usage Notes

1. The old parameters **DOMAIN=** and **OPID** are ignored.
2. The requested function must be defined in the CICS Automation program-to-program interface initialization members, as shown for the CEMT function:

In NetView (EVENTASK)

```
SERVER=RESPONSE,CEMT,AUTCPPI,EVESNRSP
```

In CICS (EVEMPINM)

```
EVEMPINM TYPE=ENTRY,      DEFINE A FUNCTION  
          FUNCTION=CEMT,   FUNCTION NAME  
          TRANSID=COMT     TRANSACTION NAME
```

3. The maximum amount of text that can be specified on the EVESNCCI command is limited to 240 bytes, including the command name EVESNCCI.
4. The maximum amount of text that can be specified with a segment chain is 32656 bytes.
5. EVESNCCI saves the segments until the final segment is received. After receiving the final segment, EVESNCCI assembles the segments into one block which is sent to the specified receiver using the EVENTASK optional task.

Note: It is assumed that the user of the segment function provides the segments in the correct order and that the segment identification is unique within the NetView task.

6. If an internal error is found during processing of a segment request, such as a GETMAIN failure, all existing segments are deleted.
7. All CANCEL commands (initiated when TYPE=X is used) are logged together with their results. The CANCEL command is always successful. Users of the segment function are requested to use TYPE=X when applicable to prevent NetView from being filled with unused data.

EVESNCCI: NetView to CICS Communication Interface

8. The following matrix shows the required (R), optional (O), and invalid (¬) keywords for the various command types.

TYPE=	NAME	FUNC	DATA	OPID	DOMA	REQUI	SEGM	ID
C	R	R	O	O	O	¬	O _a	O _d
S	R	R	O	¬	¬	¬	O _a	O _d
R	R	R	O	¬	¬	R	O _a	O _d
N	R	R	O _b	¬	¬	R	¬	¬
A	R	R	¬	¬	¬	R	¬	¬
X	¬	¬	¬	¬	¬	¬	¬	R
none	¬	¬	O	¬	¬	¬	R _c	R _d

Notes:

- SEGMENT=F only can be used with these TYPEs.
 - The length of the data must be less than 101 bytes.
 - SEGMENT=M or SEGMENT=L can only be used when no TYPE is specified.
 - When a SEGMENT is specified, an ID must also be given.
9. EVESNCCI returns the following return codes and error messages to the caller. Some of the messages are written to the NetView log. Message EVE122E is also sent to the authorized receiver.

RC=0 EVE120I COMMAND ACCEPTED FOR *subsys*, APPLID = *applid*

The command has been forwarded to the EVENTASK optional task, where *subsys* is the symbolic name by which this CICS subsystem is known to CICS Automation, and *applid* is the generic VTAM application identifier of the target CICS subsystem.

RC=4 EVE121E ERROR ON DSI_{xxx} REQUEST IN EVESNCCI, RC=*ccc*

The command has not been forwarded to the EVENTASK optional task, where *xxx* identifies the DSI request (such as GET, FRE, FIND, PUSH, or POP), and *ccc* is the return code returned by the DSI_{xxx} function.

RC=8 EVE122E EVENTASK TASK NOT ACTIVE

The command has not been forwarded to the EVENTASK optional task.

RC=12 EVE123E INPUT ERROR AT DISPLACEMENT *ddd*, CODE=*ccc*

The command has not been forwarded to the EVENTASK optional task, where *ddd* contains the location in the command string where the error was detected, and *ccc* is one of the following:

- 004** Unrecognized keyword.
- 008** Syntax error.
- 012** Operand error.
- 016** Duplicate keyword.
- 020** Conflicting keyword.
- 024** Required keyword(s) omitted.
- 028** Incorrect data length. The data length on an ACK

EVESNCCI: NetView to CICS Communication Interface

response was not zero, or the data length on a NACK response was larger than 100 bytes.

RC=16

EVE124E SEGMENT ERROR, CODE = *ccc*

The command has not been forwarded to the EVENTASK optional task. All existing segments that have the current ID are deleted, except when the segment-chain was corrupted, where *ccc* is one of the following:

004 SEGMENT SEQUENCE ERROR. A middle or last segment has been offered while no first segment with an identical ID was available, or a first segment has been offered while another first segment with the same ID already exists.

008 TOO MUCH DATA. In a series of segments with identical ID the total amount of data exceeds 32656 bytes.

012 SEGMENT-CHAIN CORRUPTED. Storage used for saving segment data has been overwritten.

RC=20

EVE125E NO STORAGE AVAILABLE ON DSI_{xxx} REQUEST IN EVESNCCI

The command has not been forwarded to the EVENTASK optional task.

RC=24

ERROR ON EVESX_{mmm} CALL IN EVESNCCI, RC = *ccc*

The command has not been forwarded to the EVENTASK optional task. EVESNCCI calls the EVESX001 (CICSSEC) and EVESX022 (CICSQRY) command processors. A non-zero return code from either of these command processors results in this return code and error message. This error implies (normally) that either the caller is not authorized for the function on the specified subsystem, or the specified subsystem is not defined.

10. When other errors are detected during the processing of a CONVERSE request, CICS Automation program-to-program interface returns a NACK response to the requestor. The NACK response data indicates the type of error that occurred. The errors are also logged.

The following NACK responses can be expected:

EVE129E text

EVE122E *task* TASK NOT ACTIVE

EVE136E ERROR ON PPI REQUEST *rrr*, RC = *ccc*

EVE137E NETVIEW SUBSYSTEM NOT AVAILABLE

EVE141E INCORRECT MQS BUFFER RECEIVED IN *progrname*

EVE142E FUNCTION *funcname* NOT FOUND IN *membrname*

EVE171E *progrname* : ERROR IN *progrname(tran)*, CODE = *cccc*

EVE175E *progrname* : FUNCTION *funcname* NOT FOUND IN EVESPINM

EVE181E *progrname* : ERROR ON TRANSACTION START *tran* FOR FUNCTION *funcname*

Examples of Usage

Example 1

```
"EVESNCCI TYPE=C,"||,  
  "NAME=CICS1,"||,  
  "FUNC=CEMT,"||,  
  "DATA='I PR(E*)'"
```

This command starts a CEMT transaction in the CICS *subsystem* known to CICS Automation as CICS1.

Example 2

```
"EVESNCCI TYPE=R,"||,  
  "NAME=CICS2,"||,  
  "FUNC=LMT,"||,  
  "DATA=(1st segment of LMT response),"||,  
  "REQID=1234567890123456,"||,  
  "SEGMENT=F,"||,  
  "ID=QAZWSXEDCRFVTGBY"  
  
"EVESNCCI SEGMENT=M,"||,  
  "DATA=(2nd segment of LMT response),"||,  
  "ID=QAZWSXEDCRFVTGBY"  
  
"EVESNCCI SEGMENT=L,"||,  
  "DATA=(3rd segment of LMT response),"||,  
  "ID=QAZWSXEDCRFVTGBY"
```

This is a link monitor response, which is in 3 segments. Error logic is not included.

EVESNRSP: Common Response Handler from CICS

The maximum length of a RESPONSE response line is 216 characters. EVESNRSP deblocks the response data in NetView and turns it into a multi-line WTO by calling the EVESX002 (CICSBMSG) command processor for each response line.

Comments and Usage Notes

1. The request data is scanned for the presence of NL (X'15') characters within the maximum response line length. If an NL character is found, it delimits the current response line. If no NL character is found, a maximum length response line is assumed, or the end of the response data delimits the current response line.

2. The following EVESX002 (CICSBMSG) commands are issued by EVESNRSP:

```
EVESX002 START,"domainid,opid
EVESX002 DATA,C,EVE790I PPI RESPONSE FROM applid FOR FUNCTION function maxln
totln
EVESX002 DATA,D,response line text
EVESX002 DATA,E,EVE792I END"
```

This results in the following multi-line WTO to be sent to *opid* on *domainid*:

```
EVE790I PPI RESPONSE FROM applid FOR FUNCTION function maxln totln
response line text
:
response line text
EVE792I END
```

where *maxln* and *totln* contain the maximum response line length and the total length of the RESPONSE response data sent to NetView.

3. The maximum value of the total response data length is 32656 bytes. If this number of bytes is sent on a CEMT response, the CEMT response may be truncated.
4. Errors found during EVESNRSP processing are logged. The following error messages may be displayed:

```
EVE121E ERROR ON DSIxxx REQUEST IN EVESNRSP, RC = ccc
EVE125E NO STORAGE AVAILABLE ON DSIxxx REQUEST IN EVESNRSP
EVE127E ERROR ON EVESX002 CALL IN EVESNRSP, RC = ccc
EVE141E INCORRECT MQS BUFFER RECEIVED IN EVESNRSP
```

EVESCCCI: CICS to NetView Communication Interface

Use this subroutine to request, from a CICS transaction, the initiation of a NetView command, a command list, or command processor in NetView. There are three types of EVESCCCI requests:

1. A CONVERSE request, which expects a response from NetView.
2. A SEND request, which does not expect a response from NetView.
3. RESPONSE.

The request is initiated by setting up a parameter list and linking to the EVESCCCI routine, as shown in the following assembler example:

```
EXEC CICS LINK PROGRAM(EVESCCCI) COMMAREA(area) LENGTH('60')
```

The parameter list is located in *area*. The following is an example of a parameter list built for a CONVERSE request (see Table 7 on page 84 for a SEND request parameter list example):

Table 5. EVESCCCI CONVERSE Request Parameter List

Name	Disp.	Length	Contents and description
OUTIDENT	000	08	"POUT0001"
OUTREQID	008	16	Not used (set by EVESCCCI)
OUTFNAME	024	08	<i>function</i>
OUTRTYPE	032	01	"C"
	033	01	Reserved (binary zeros)
OUTWAITC	034	02	CONVERSE wait time in seconds
OUTDATAL	036	04	Length of request data area (binary)
OUTDATAA	040	04	Address of request data area

Note: Sample copy books for this parameter list are included in the CICS Automation sample library. Refer to "EVEMPINT: EVESCCCI Parameter List Copy Book" on page 85 for field descriptions and important usage information.

If the request data address is located below the line, the response area (if any) is allocated below the line as well. If no data is passed on the CONVERSE request, the length field (OUTDATAL) must be zero. The address field is still used to determine the location of the response area in this case.

If the request is accepted, EVESCCCI sends the request to the NetView EVENTASK optional task, which translates the function into a command list or command processor.

EVESCCCI waits *nn* seconds (see OUTWAITC) for the response to arrive. If the OUTWAITC value is zero, the default wait time is 30 seconds. Valid specifications in the OUTWAITC field range from 1 up to and including 999 (seconds).

If the response does not arrive within the expected interval, the timeout return code is passed to the caller.

EVESCCCI: CICS to NetView Communication Interface

EVESCCCI returns the following fields to the caller of the CONVERSE request.

Table 6. EVESCCCI Fields Returned to Caller from CONVERSE Request

Name	Disp.	Length	Contents and description
OUTRESPA	044	4	Address of response area
OUTRESPL	048	4	Length of response area
OUTRTRNC	052	4	Binary return code: 0 Successful request 4 Timeout on CONVERSE 8 Program-to-program interface not available (see OUTABNDC for error code) 12 Incorrect parameter list 16 Internal processing error (see OUTABNDC for error code)
OUTABNDC	056	4	Error code (character) <ul style="list-style-type: none"> • OUTRTRNC = 8 <ul style="list-style-type: none"> C003 Program-to-program interface not active C015 CICS LOAD/LINK failure C017 No CICS storage available C2XX Program-to-program interface request error • OUTRTRNC = 16 <ul style="list-style-type: none"> A... CICS abend codes C012 CICS READQ failure C016 CICS POST failure C018 CICS FREEMAIN failure C960 Incorrect TS item length C961 RQE chain corrupted

Note: Refer to “EVEMPINT: EVESCCCI Parameter List Copy Book” on page 85 for field descriptions and important usage information.

The response area contains the response (if any) on the CONVERSE request. It may contain a RESPONSE, an ACK, or a NACK. The area has the format as described by the EVEMPINT copy book. **It is the responsibility of the caller of EVESCCCI to free the response area via EXEC CICS FREEMAIN.**

The response area format is similar to the transaction input data passed on a CONVERSE or SEND request from NetView:

Name	Disp.	Length	Contents and description
INTIDENT	000	08	“PINT0001”
INTREQID	008	16	<i>domainid opid</i>
INTFNAME	024	08	<i>function</i>
INTRTYPE	032	01	Response type: “R”, “A”, or “N”
	033	03	Reserved (binary zeros)
INTDATAL	036	04	Length of <i>data</i> (binary). Zero for ACK response
INTSDATA	040	<i>nn</i>	<i>data</i>

Errors found by EVESCCCI are returned to the caller via a return code and an error code. When other errors are detected during the processing of a CONVERSE request, CICS Automation returns a NACK response to the CICS transaction. The

EVESCCCI: CICS to NetView Communication Interface

NACK response data indicates the type of error that occurred. The NACK response is logged via message EVE180E. The following NACK responses can be expected:

```
EVE180E text
      EVE121E ERROR ON DISxxx REQUEST IN progname, RC = ccc
      EVE122E tttttttt TASK NOT ACTIVE
      EVE125E NO STORAGE AVAILABLE ON DSIxxx REQUEST IN progname
      EVE142E FUNCTION funcname NOT FOUND IN memname
```

The following is an example of a parameter list built by a SEND request:

Table 7. EVESCCCI SEND Request Parameter List

Name	Disp.	Length	Contents and description
OUTIDENT	000	08	"POUT0001"
OUTREQID	008	16	Not used (set by EVESCCCI)
OUTFNAME	024	08	<i>function</i>
OUTRTYPE	032	01	"S"
	033	01	Reserved (binary zeros)
	034	02	Not used for SEND (binary zeros)
OUTDATAL	036	04	Length of request data area (binary)
OUTDATAA	040	04	Address of request data area

Note: Refer to "EVEMPINT: EVESCCCI Parameter List Copy Book" on page 85 for field descriptions and important usage information.

If no data is passed on the SEND request, the length field (OUTDATAL) must be 0 (zero).

On a SEND request, errors found by EVESCCCI are returned to the caller via a return code and an error code. Other errors detected during the processing of a SEND request are not returned to the requestor. The errors are only logged.

EVEMPINT: EVESCCCI Parameter List Copy Book

The EVEMPINT copy book describes the CICS Automation program-to-program interface parameter list on the CICS side. It consists of two parts: one part describes the input data for CICS transactions, and the other part describes the format of output requests from CICS transactions.

The following data area is passed to a started CICS transaction as result of a NetView CONVERSE or SEND request. The same data area is passed in the response area as result of a NetView RESPONSE, ACK, or NACK response. See "EVESNCCI: NetView to CICS Communication Interface" on page 76.

Note: The started transaction must obtain the input data using an EXEC CICS RETRIEVE command.

Name	Disp.	Length	Contents and description
INTIDENT	000	08	Provides an eye-catcher and a block format level.
INTREQID	008	16	This field contains the request identifier which is used to relate a response to a specific request. For a CONVERSE request from NetView, it contains the <i>domainid</i> <i>opid</i> concatenation specified on the EVESNCCI command. For responses from NetView, it contains the request identifier allocated by the EVESCCCI routine on a CICS CONVERSE request. This field is not used for a SEND request.
INTFNAME	024	08	Contains the function name specified with the EVESNCCI FUNCTION= keyword.
INTRTYPE	032	01	The response type: C for CONVERSE, S for SEND, R for RESPONSE, A for ACK, and N for NACK.
	033	03	Reserved (binary zeros)
INTDATAL	036	04	Contains the length of the request or response data. This field may have a 0 (zero) value. For an ACK response, this field is 0 (zero).
INTSDATA	040	<i>nn</i>	The request or response data, if any. The length of the response data is contained in the INTDATAL field.

The following fields are set by the caller of the EVESCCCI routine.

Name	Disp.	Length	Contents and description
OUTIDENT	000	08	Provides an eye-catcher and a block format level. Must be set to POUT0001.
OUTREQID	008	16	This field is used as a request identifier to relate a response to a specific CONVERSE request. For CICS CONVERSE requests this field is set by the EVESCCCI routine. For CICS responses, the field must be set by the caller (copied from INTREQID). This field is not used for SEND requests.
OUTFNAME	024	08	Specifies the function name. The EVENTASK initialization member must contain a SERVER=REQUEST entry (for CONVERSE or SEND) or a SERVER=RESPONSE entry (for RESPONSE) specification. For responses, this field is normally copied from the transaction input data (INTFNAME field) Note: If the name is less than eight characters, pad it with blanks.

EVEMPINT: EVESCCCI Parameter List Copy Book

Name	Disp.	Length	Contents and description
OUTRTYPE	032	01	Specifies the type of command as REQUEST, SEND, RESPONSE, ACK, or NACK. REQUEST (also referred to as CONVERSE) and SEND are requests. RESPONSE, ACK, and NACK are responses. REQUEST starts a command processor or a command list in NetView. A response is expected. SEND also starts a command processor or a command list in NetView, but no response is expected. RESPONSE is used to send data to a NetView task. ACK signals the successful completion of a CONVERSE request. No data is provided. NACK signals the unsuccessful completion of a CONVERSE request. Data may optionally be provided (maximum of 100 bytes). It is recommended for health checking.
	033	01	Reserved (binary zeros)
OUTWAITC	034	02	Specifies the number of seconds (binary value) to wait for a response on a CONVERSE request. If the response does not arrive within the specified time interval, the timeout return code is passed to the caller. If binary zero is specified, the default wait interval (30 seconds) is used. The maximum binary value that can be specified is 999. The field must contain binary zeros for all requests other than CONVERSE.
OUTDATAL	036	04	This field must be set to the length of the CONVERSE or SEND request data area or to the length of the RESPONSE or NACK response area. The maximum length of a CONVERSE or SEND request area or a RESPONSE response area is 32656 bytes. The maximum length of a NACK response area is 100 bytes.
OUTDATAA	040	04	Specifies the address of the CONVERSE or SEND request area or the address of the RESPONSE or NACK response area. If the request data address is located below the line, the response area (if any) is allocated below the line as well. If no data is passed on the CONVERSE request, the length field (OUTDATAL) must be zero. The address field is still used to determine the location of the response area in this case.

The following fields are set by EVESCCCI upon return to the caller:

Name	Disp.	Length	Contents and description
OUTRESPA	044	04	Address of response area allocated by EVESCCCI. It may contain a RESPONSE, ACK, or NACK response. This field is only returned on a successful CONVERSE request (OUTRTRNC=0). It is the responsibility of the caller of EVESCCCI to free the response area using EXEC CICS FREEMAIN.
OUTRESPL	048	04	Contains the length of the response area addressed by the OUTRESPA field.
OUTRTRNC	052	04	Contains the return code which will have one of the following binary values: 000 Successful request. 004 Timeout on CONVERSE. 008 Program-to-program interface not available (see OUTABNDC for error codes). A transaction dump is provided depending on the error code. 012 Incorrect parameter list. No transaction dump is provided. 016 Internal processing error (see OUTABNDC for error codes). A transaction dump is provided.

EVEMPINT: EVESCCCI Parameter List Copy Book

Name	Disp.	Length	Contents and description
OUTABNDC	056	04	<p>This field contains an error code for return codes 008 and 016. For return code 008 the field will have one of the following character values:</p> <p>C003 The CICS component of the program-to-program interface is not active. Use the COPS transaction to start it. No transaction dump is provided. An error message is logged.</p> <p>C015 A CICS LOAD of or LINK to a required module was not successful. Ensure that EVESPERR and EVESPMMSG have been properly installed and are enabled. A transaction dump is provided. An error message is logged.</p> <p>C017 No CICS storage is available. No transaction dump is provided.</p> <p>C2xx A program-to-program interface request error occurred, where <i>xx</i> contains the program-to-program interface request return code. For error codes C220, C222, C223, C225, C231, C233, C236, C240, and C290, a transaction dump is provided and an error message is logged.</p> <p>For return code 016 the field will have one of the following character values:</p> <p>A*** A CICS abend has occurred. A transaction dump is provided. An error message is logged for most A*** errors.</p> <p>C012 An unexpected error has occurred on a READQ command. A transaction dump is provided. An error message is logged.</p> <p>C016 An unexpected error has occurred on a POST command. A transaction dump is provided. An error message is logged.</p> <p>C018 An unexpected error has occurred on a FREEMAIN command. A transaction dump is provided.</p> <p>C961 Internal error. Incorrect TS item length. A transaction dump is provided. An error message is logged.</p> <p>C962 Internal error. The RQE chain is corrupted. A transaction dump is provided. An error message is logged.</p>

When a RESPONSE response (R) is sent, *function* is used to locate the name of a command processor or command list in the EVENTASK initialization member. This command is scheduled under a task (also defined in the initialization member) with the following command text:

Name	Disp.	Length	Contents and description
	000	8	Command processor or command list name
	008	8	" " (8 blanks)
RHDRCVID	016	8	Program-to-program interface receiver identification
RHDSNDID	024	8	Generic applid (Program-to-program interface sender identification)
RHDPRCNM	032	8	Program-to-program interface sender's JOB- or STC-name
RHDDOMID	040	8	<i>domainid</i>
RHDTSKID	048	8	<i>opid</i>
RHDFNAME	056	8	<i>function</i>
RHDRTYPE	064	1	"R"
	065	7	"*****" (7 asterisks)
RHDSDATA	072	n	Response data (n = OUTDATAL)

A common response processor, EVESNRSP, is available that turns the response data into a multi-line WTO (EVE79xI), which is sent to *opid* in *domainid*.

Examples of Usage

Example 1

This assembler example shows the processing of a CICS CONVERSE request.

```

*****
*          ISSUE A CONVERSE REQUEST          *
*****
*
*      XC  CISPOUTP,CISPOUTP  ZERO PARAMETER LIST
*      LA  R6,CISPOUTP      ADDRESS PARAMETER LIST
*      USING OUTDSECT,R6    ESTABLISH ADDRESSABILITY
*      SPACE 1
*      MVC OUTIDENT,=C'POUT0001' SET EYE CATCHER/BLOCK LEVEL
*      MVC OUTFNAME,=CL8'TESTCLST' SET FUNCTION NAME
*      MVI OUTRTYPE,OUTRTYPC SET CONVERSE REQUEST TYPE
*      LA  R1,L'CONVTEXT    LENGTH OF VARIABLE DATA
*      ST  R1,OUTDATAL      SET LENGTH OF VARIABLE DATA
*      LA  R1,CONVTEXT      ADDRESS OF VARIABLE DATA
*      ST  R1,OUTDATAA      SET ADDRESS OF CONVERSE DATA
*      MVC CISHWORD,=Y(OUTHL) LENGTH OF PARAMETER LIST
*
*      EXEC CICS LINK,      REQUEST PROGRAM LINK, TO          *
*      PROGRAM('EVESCCCI'), CPDS COMMUNICATION INTERFACE    *
*      COMMAREA(CISPOUTP), PARAMETER LIST                    *
*      LENGTH(CISHWORD)  PARAMETER LIST LENGTH
*
*      L   R15,OUTRTRNC    PICK UP THE RETURN CODE
*      Process the return code and error code please!
*
*      L   R4,OUTRESPA     ADDRESS RESPONSE AREA
*      USING INTDSECT,R4  ESTABLISH ADDRESSABILITY
*
*      LA  R14,INTSDATA   ADDRESS RESPONSE DATA
*      L   R15,INTDATAL   LENGTH RESPONSE DATA
*      Do some meaningful processing please!
*
*      CONVTEXT DC  C'CONVERSE TEXT FROM CICS'
*
*      DFHEISTG ,
*
*      CISPOUTP DS  CL(OUTHL)  RESPONSE PARAMETER LIST
*      CISHWORD DS  H          PARAMETER BLOCK LENGTH
*
*      DFHEIEND ,
*
*      EVEMPINT ,          DEFINE INTERFACE DSECT

```

Example 2

This assembler example shows the processing of a NetView CONVERSE request in CICS.

```

*****
*      RETRIEVE TRANSACTION INPUT DATA      *
*****
*
*      EXEC  CICS RETRIEVE,      GET INPUT DATA      *
*            SET(R4),           RETURN ADDRESS HERE   *
*            LENGTH(CISHWORD)   RETURN LENGTH HERE
*
*      USING INTDSECT,R4      ESTABLISH ADDRESSABILITY
*
*      LA   R14,INTSDATA      ADDRESS REQUEST DATA
*      L    R15,INTDATAL      LENGTH REQUEST DATA
*      Do some meaningful processing please!
*
*****
*      RETURN A 'NORMAL' RESPONSE          *
*****
*
*      XC   CISPOUTP,CISPOUTP  ZERO PARAMETER LIST
*      LA   R6,CISPOUTP        ADDRESS PARAMETER LIST
*      USING OUTDSECT,R6      ESTABLISH ADDRESSABILITY
*
*      MVC  OUTIDENT,=C'POUT0001' SET EYE CATCHER/BLOCK LEVEL
*      MVC  OUTREQID,INTREQID   SET REQUEST IDENTIFIER
*      MVC  OUTFNAME,INTFNAME   SET FUNCTION NAME
*      MVI  OUTRTYPE,OUTRTYPR  SET RESPONSE RESPONSE TYPE
*      LA   R1,L'RESPTEXT      LENGTH OF VARIABLE DATA
*      ST   R1,OUTDATAL         SET LENGTH OF VARIABLE DATA
*      LA   R1,RESPTEXT        ADDRESS OF VARIABLE DATA
*      ST   R1,OUTDATAA        SET ADDRESS OF RESPONSE DATA
*      MVC  CISHWORD,=Y(OUTHL) LENGTH OF PARAMETER LIST
*
*      EXEC  CICS LINK,         REQUEST PROGRAM LINK, TO      *
*            PROGRAM('EVESCCCI'), CPDS COMMUNICATION INTERFACE *
*            COMMAREA(CISPOUTP), PARAMETER LIST              *
*            LENGTH(CISHWORD)  PARAMETER LIST LENGTH
*
*      L    R15,OUTRTRNC        PICK UP THE RETURN CODE
*      Process the return code and error code please!
*
RESPTEXT DC  C'RESPONSE TEXT FROM CICS'
*
*      DFHEISTG ,
*
*      CISPOUTP DS  CL(OUTHL)    RESPONSE PARAMETER LIST
*      CISHWORD DS  H           LENGTH OF PARAMETER BLOCK
*
*      DFHEIEND ,
*
*      EVEMPINT ,              DEFINE INTERFACE DSECT

```

Customizing CICS Definitions

Perform the following additional customization to use the NetView program-to-program interface.

These customization steps are only required if you want to use health checking or link monitoring as implemented in previous releases.

Step 1: Modifications to Program-to-Program Interface Initialization

Step 1a (Optional): Member EVENTASK

This member defines the name of the PPI receiver in NetView for requests from CICS, the default buffer queue limit, and the names and programs of the requests to be executed in NetView. The defaults as specified in this member will work for most installations. If you want to change any of the parameters, see “EVENTASK: NetView PPI Initialization Member” on page 92 for details.

Step 1b (Optional): Member EVESPINM

A sample member is delivered in SINGSRC (see “EVESPINM: CICS PPI Initialization Member” on page 91 for details). It only needs to be modified if:

- The RECEIVERID is changed. This value must be the same as the value defined in “Step 1: Modifications to Program-to-Program Interface Initialization.”
- You are using the program-to-program interface for your own transactions.

In these cases, do the following:

1. Edit EVESPINM. USERID=YES means that CICS security checking is required, USERID=NO means that CICS security checking is not required.
2. Assemble the program-to-program interface initialization member.
3. Place the assembled member in one of the libraries in the CICS DFHRPL chain.

Step 2: Define a NetView PPI Receiver Task

A PPI receiver is required to allow CICS subsystems to communicate with NetView. The CICS Automation Health Checking function uses this interface. This is a required step and must be done for correct automation of a CICS subsystem.

Step 3: Define a CICS PPI Receiver Task

A PPI receiver is required to allow NetView to communicate with CICS subsystems. This is an optional step that is required only for health checking. It enables operator control of the PPI receivers in each CICS subsystem.

Note: A CICS PPI subsystem may be defined for any CICS subsystem for which the operator wishes to start or stop the PPI receivers in the CICS address space.

Definition Members

The definition members are:

“EVESPINM: CICS PPI Initialization Member” on page 91.

This member is used on the CICS side of the program-to-program interface to:

1. Set the program-to-program interface buffer queue limit. This is the number of outstanding buffers that can be stored in the receiver buffer queue.
2. Define the program-to-program interface receiver identifier for the EVESNPPI NetView subtask program.
3. Provide the identifier of the console on which the long-running COPC transaction is started.
4. Define the relationships between function names and CICS transaction names.

“EVENTASK: NetView PPI Initialization Member” on page 92

This member is used on the NetView side of the program-to-program interface to:

1. Set the program-to-program interface buffer queue limit. This is the number of outstanding buffers that can be stored in the receiver buffer queue.
2. Define request programs and autotasks to be used for specific functions.
3. Define response programs and autotasks to be used for specific functions.
4. Define the program-to-program interface receiver identifier for the EVESNPPI NetView subtask program.

EVESPINM: CICS PPI Initialization Member

This member is used on the CICS side of the program-to-program interface to:

1. Set the program-to-program interface buffer queue limit. This is the number of outstanding buffers that can be stored in the receiver buffer queue.
2. Define the program-to-program interface receiver identifier for the EVESNPPI NetView subtask program.
3. Provide the identifier of the console on which the long-running COPC transaction is started.
4. Define the relationships between function names and CICS transaction names.

Note: There is a corresponding initialization member on the NetView side. See “EVENTASK: NetView PPI Initialization Member” on page 92.

A sample for this member is delivered in the SINGSAMP data set.

Keyword and Parameter Definitions

TYPE=

Indicates the type of entry this is. Valid types are:

- | | |
|----------------|--|
| INITIAL | The first EVEMPINM type specified. Only one INITIAL entry can be specified. |
| ENTRY | This type associates a function with a CICS transaction. |
| FINAL | Indicates that this is the final entry. Only one FINAL entry can be specified. |

BUFFQL=

Specifies the buffer queue limit for the CICS receiver side of the program-to-program interface to NetView. A minimum value of 1 and a maximum value of 15 can be specified. If this keyword is omitted, a default value of 3 is assumed. This keyword is only valid with TYPE=INITIAL.

Definition Members

RECEIVERID=

Specifies the identifier of the NetView receiver. If this keyword is omitted, NETVCPPI is assumed. This keyword is only valid with TYPE=INITIAL.

USERID=[YES | NO]

Specifies that the transaction will be invoked with the NetView user ID that invoked the PPI process. The default is NO.

CONSOLE=

Specifies the 1- to 4-character terminal identifier of the console on which the long-running COPC transaction is started. If this specification is omitted, COPC is started without a terminal. This keyword is only valid with TYPE=INITIAL.

FUNCTION=

The name of the function to be executed. The function name can be from 1 to 8 characters and must not start with the characters EVE.

TRANSID=

The name of the CICS transaction associated with this function. This transaction will be executed when the function is requested.

Comments and Usage Notes

1. A function name may not start with EVE.
2. EVESPINM must be link-edited into one of the CICS DFHRPL libraries.
3. At least one valid TYPE=ENTRY must be specified.
4. There must be a TYPE=ENTRY definition for each function that uses the CICS Automation program-to-program interface. The corresponding NetView side initialization member entry looks like this:
SERVER=RESPONSE,CEMT,AUTCPPI,EVESNRSP
where CEMT is the function.
5. If you are running CICS Automation in more than one NetView domain on the same MVS system, then you need to provide unique RECEIVERIDs in this member. This must also be changed in the corresponding NetView program-to-program interface initialization member. See "EVENTASK: NetView PPI Initialization Member," which explains where the matching RECEIVERID is changed for that member.

EVENTASK: NetView PPI Initialization Member

This member is used on the NetView side of the program-to-program interface to:

1. Set the program-to-program interface buffer queue limit. This is the number of outstanding buffers that can be stored in the receiver buffer queue.
2. Define request programs and autotasks to be used for specific functions.
3. Define response programs and autotasks to be used for specific functions.
4. Define the program-to-program interface receiver identifier for the EVESNPPI NetView subtask program.

Note: There is a corresponding initialization member on the CICS side. See "EVESPINM: CICS PPI Initialization Member" on page 91.

A sample for this member is delivered in the SINGSAMP data set.

The following is an example of the information contained in the EVENTASK program-to-program interface initialization member:

Format

```

BUFFQL=20
SERVER=REQUEST,LMT,AUTCPPI,EVEEYPPS
SERVER=RESPONSE,CEMT,AUTCPPI,EVESNRSP
SERVER=RESPONSE,LMT,AUTCPPI,EVESNRSP
SERVER=REQUEST,NACK,AUTCPPI,EVESNACK
SERVER=RESPONSE,NACK,AUTCPPI,EVESNACK
RECEIVERID=NETVCPPI

```

Keyword and Parameter Definitions**BUFFQL**

This is a 2- or 3-digit numeric value. The minimum value is 10 and the maximum is 999. If this entry is omitted, a value of 15 is assumed.

SERVER=

These entries define:

1. Whether this function is a REQUEST or a RESPONSE. A REQUEST is used to identify a receiver program to be invoked if NetView gets a CONVERSE or SEND from CICS. A RESPONSE is used to identify a sender program to be invoked if CICS sends a RESPONSE. See “EVESCCCI: CICS to NetView Communication Interface” on page 82.
2. The function, such as LMT (link monitor) or CEMT.
3. The operator ID under which the program runs, for example, AUTCPPI.
4. The command list or command processor used for this function, such as EVEEYPPS (the receiver program for LMT functions from CICS) and EVESNRSP (the common response handler).

RECEIVERID=

The program-to-program interface receiver identifier for the NetView side program-to-program interface subtask program. If omitted, NETVCPPI is assumed.

Comments and Usage Notes

1. A function name may not start with EVE.
2. At least one valid SERVER must be specified.
3. There must be a SERVER entry for each function that uses the CICS Automation program-to-program interface. The corresponding CICS side initialization member entry looks like this:

```

EVEMPINM TYPE=ENTRY,      DEFINE A FUNCTION
      FUNCTION=LMT,        FUNCTION NAME
      TRANSID=COLR         TRANSACTION NAME

```

4. If you are running CICS Automation in more than one NetView domain on the same MVS system, then you need to provide unique RECEIVERIDs in this member. This must also be changed in the corresponding CICS program-to-program interface initialization member EVESPINM. See “EVESPINM: CICS PPI Initialization Member” on page 91.

Security Checking Using CICS

You can use CICS-supplied security to restrict which operators can access defined resources within a CICS environment.

The security check works by using the NetView operator ID that invoked the CICS Automation function. When the function to be performed is invoked in the NetView environment, the invoking operator ID is passed to the CICS system on which the action will be taken. The appropriate transaction or function is invoked, and the NetView operator ID is used in all CICS security checks.

Security Checking Using CICS

To use this security, you must:

- Define all NetView operators which will invoke CICS functions to RACF® (or your SAF-compliant security system). This will include:
 - Regular NetView operators
 - NetView autotasks which perform CICS-related actions. These autotasks include those autotasks specifically defined for CICS Automation use, and may include the autotasks which process shutdown functions or resynchronization functions.
- Define RACF surrogate authorization for CICS.
- Connect the NetView operators to the CICS resources which they will need to access, such as transactions, programs and files. This connection is done through your SAF security manager (such as RACF).
- Enable the security by modifying the EVESPINM member and specifying USERID=YES to enable extended support. For more information on EVESPINM, see “EVESPINM: CICS PPI Initialization Member” on page 91.
- Enable non-terminal transaction security in CICS by modifying the CICS SIT to specify XTRAN=YES and XUSER=YES. Additional CICS definitions may require similar modification, such as PLTPIUSER.

Note: In order to perform any of the basic functions of CICS Automation, such as displaying subsystem information, an operator must be authorized to use the ACF command. For this command, see *IBM Tivoli System Automation for z/OS Operator's Commands*.

Glossary of CICS and Other Terms

This glossary defines special CICS terms used in the library and words used with other than their everyday meaning. In some cases, a definition may not be the only one applicable to a term, but gives the particular sense in which it is used in the CICS Automation library.

abend. Abnormal end of task.

ACB. Access method control block (VTAM and VSAM).

ACK. Acknowledgement.

APAR. Authorized program analysis report.

application program. (1) A program written for or by a user that applies to the user's work. (2) In data communication, a program used to connect and communicate with stations in a network, enabling users to perform application-oriented activities.

ASCII. American National Standard Code for Information Interchange.

batch. An accumulation of data to be processed.

CEC. Central Electronic Complex.

CEMT. The CICS master terminal transaction.

central electronic complex (CEC). A conglomeration of several processors and other devices in one or more physical units. This usually means several processors running under the control of a single MVS/ESA™ operating system. For example, a 3090™ model 400® processor complex can run as a 4-processor CEC, or can be partitioned into the equivalent of two 3090 model 200s, each of which runs as a CEC with its own operating system.

CICS. Customer Information Control System.

command. In CICS, an instruction similar in format to a high-level programming language statement.v(Contrast with macro.) CICS commands invariably include the verb EXECUTE (or EXEC). They may be issued by an application program to make use of CICS facilities.

command-language statement. In CICS, synonym for command.

concurrent. Pertaining to the occurrence of two or more activities within a given interval of time.

data security. The protection of data against unauthorized disclosure, transfer, modifications, or destruction, whether accidental or intentional.

data set. The major unit of data storage and retrieval, consisting of a collection of data in one of several prescribed arrangements and described by control information to which the system has access.

end user. In CICS, anyone using CICS to do a job, usually by interacting with an application program (transaction) by means of a terminal.

exception. An abnormal condition such as an I/O error encountered in processing a data set or a file, or using any resource.

initial program load (IPL). The initialization procedure that causes an operating system to commence operation.

initialization. (1) Actions performed by CICS to construct the environment in the CICS region to enable CICS applications to be run. (2) A process started by SA z/OS and CICS Automation to construct the environment in which automation is to occur.

installation. (1) A particular computing system, in terms of the work it does and the people who manage it, operate it, apply it to problems, service it and use the work it produces. (2) The task of making a program ready to do useful work. This task includes generating a program, initializing it, and applying PTFs to it.

intercommunication facilities. A generic term covering intersystem communication (ISC) and multiregion operation (MRO).

interregion communication (IRC). The method by which CICS provides communication between a CICS region and another region in the same processor. Used for multiregion operation.

intersystem communication (ISC). Communication between separate systems by means of SNA networking facilities or by means of the application-to-application facilities of an SNA access method. ISC links CICS systems, and may be used for user application to user application communication, or for transparently executing CICS functions on a remote CICS system.

IPL. Initial Program Load.

IRC. Interregion communication.

ISC. Intersystem communication.

keyword. (1) A symbol that identifies a parameter. (2) A part of a command operand that consists of a specific character string. (3) An operand in a CEDDA definition. Key-sequenced data set—a VSAM database organization.

local. In data communication, pertaining to devices that are attached to a controlling unit by cables, rather than data links.

local device. A device, such as a terminal, whose control unit is directly attached to a computer's data channel. No data link or control unit is used. Contrast with remote device.

local system. In CICS intercommunication, the CICS system from whose point-of-view intercommunication is being discussed.

NACK. Negative Acknowledgement.

network. (1) An interconnected group of nodes. (2) The assembly of equipment through which connections are made between data stations.

network configuration. In SNA, the group of links, nodes, machine features, devices, and programs that make up a data processing system, a network, or a communication system.

online. (1) Pertaining to a user's ability to interact with a computer. (2) Pertaining to a user's access to a computer via a terminal.

operating system. Software that controls the execution of programs; an operating system may provide services such as resource allocation, scheduling, input/output control, and data management.

parameter. (ISO) A variable that is given a constant value for a specified application and that may denote the application.

processor. (ISO) In a computer, a functional unit that interprets and executes instructions.

program temporary fix (PTF). A temporary solution or by-pass of a problem diagnosed by IBM field engineering as the result of a defect in a current unaltered release of the program.

PTF. Program Temporary Fix.

PUT. Program update tape.

recovery routine. A routine that is entered when an error occurs during the performance of an associated operation. It isolates the error, assesses the extent of the error, and attempts to correct the error and resume operation.

remote. In data communication, pertaining to devices that are connected to a data processing system through a data link.

remote device. A device, such as a terminal, connected to a data processing system through a data link.

remote system. In CICS intercommunication, a system that the local CICS system accesses via intersystem communication or multiregion operation.

resource. Any facility of the computing system or operating system required by a job or task, and including main storage, input/output devices, the processing unit, data sets, and control or processing programs.

security. Prevention of access to or use of data or programs without authorization.

service. The carrying out of effective problem determination, diagnosis, and repair on a data processing system or software product.

SIT. System Initialization Table.

SNA. Systems Network Architecture.

software. (ISO) Programs, procedures, rules, and any associated documentation pertaining to the operation of a computer.

startup. The operation of starting up CICS by the system operator.

subsystem. (1) A secondary or subordinate system. (2) A resource defined to SA z/OS and CICS Automation.

system. In CICS, an assembly of hardware and software capable of providing the facilities of CICS for a particular installation.

system initialization table (SIT). A table containing user-specified data that will control a system initialization process.

systems network architecture (SNA). The description of the logical structure, formats, protocols, and operational sequences for transmitting information units through and controlling the configuration and operation of networks.

task. (1) (ISO) A basic unit of work to be accomplished by a computer. (2) Under CICS, the execution of a transaction for a particular user.

terminal. (1) A point in a system or communication network at which data can either enter or leave. (2) In CICS, a device, often equipped with a keyboard and some kind of display, capable of sending and receiving information over a communication channel.

terminal operator. The user of a terminal.

transaction. A transaction may be regarded as a unit of processing (consisting of one or more application programs) initiated by a single request, often from a

terminal. A transaction may require the initiation of one or more tasks for its execution.

update. To modify a file with current information.

VTAM. An acronym for the Virtual Telecommunications Access Method. This is one of the ways CICS communicates with terminals.

Index

Special characters

*CICS 9

A

ABCODESYSTEM 28
ABCODETRAN 21, 29
abend codes 21
accessibility xiii
ACK response 69
applications
 policy items
 CICS CONTROL 4, 10, 58
 MESSAGES/USER DATA 19
 MINOR RESOURCE FLAGS 19
 RESOURCE THRESHOLDS 20
 STARTUP 10
automation
 operators 9

B

broadcasting messages 63

C

CANCEL from NetView 71
CEMTMPI 25, 38
CICS add-on sample policy 9
CICS application samples 9
CICS Automation panels
 Broadcast Messages panel 63
 Health Checking 62
 main menu 55
CICS Message Processing 4
CICS messages, defining 12
CICS receiver program 68
CICSHLTH 39
CICSINFO 31
CICSOVRD 40
CICSplex SM address space (CMAS) 11
CICSplex SM REXX API
 installing 10
CICSPURG 42
CICSQRY 48
CICSRCMD 52
CICSRSYC 43
CICSSHUT 44
CMASSHUT 46
coexistence 17
commands
 CEMTMPI 38
 CICSHLTH 39
 CICSOVRD 40
 CICSPURG 42
 CICSQRY 48
 CICSRCMD 52
 CICSRSYC 43
 CICSSHUT 44

commands (*continued*)
 CMASSHUT 46
 EVEERDMP 45
CONVERSE
 from CICS 72
 from NetView 68
coordinating address space (CAS) 11
COPC 4

D

defining CICS messages 12
disability xiii
DISPTRG 56
dump 45

E

EVEERDMP 45
EVEMPINT—EVESCCCI parameter list
 copy book 85
EVENTASK 4, 90, 92
EVESCCCI - CICS to NetView
 communication interface 82
EVESNCCI—NetView to CICS
 Communication Interface 76
EVESNPPI 68
EVESNRSP - Common response handler
 from CICS 81
EVESPINM 90, 91
EVESPPIC 68

H

health checking 3, 21, 62
 programs 11
HEALTHCHK 32

I

INGREQ 25, 56
 Appl Parms field 58
 Override field 57
 Type field 56
INGSCHED 56

K

keyboard xiii

L

link monitoring 3, 22, 59
LISTSHUT 33
local functions 22
LookAt message retrieval tool xvi

M

main menu 55
message exit 12
message IDs, partial and
 performance 16
Message Processing
 CICS 4
message retrieval tool, LookAt xvi
messages
 broadcasting 63
MESSAGES/USER DATA keywords
 ABCODESYSTEM 28
 ABCODETRAN 21, 29
 ACORESTART 10
 HEALTHCHK 22, 32
 LISTSHUT 33
 RCVRSOS 35
 RCVRTRAN 36
migration 17
minor resources
 definitions for component
 recovery 19

N

NACK 3
NACK response 69
NetView subtask program 68

P

partial message IDs and performance 16
performance, partial message IDs
 and 16
policy data, refreshing 16
PPI
 See program-to-program interface
PPI (*see* program-to-program interface) 3
PPI receiver task 90
Processing
 CICS Messages 4
program-to-program interface 3
 CICS components 4
 CICS requests 72
 CONVERSE 72
 SEND 73
 communication components 4
 EVENTASK 4, 90, 92
 EVESPINM 91
 customization 90
 NetView requests 68
 CANCEL 71
 CONVERSE 68
 SEND 70

R

RCVRSOS 35
RCVRTRAN 36

- recovery 3
 - abend codes 21
 - minor resource flags 20
 - short-on-storage condition 19
 - thresholds 20
 - transactions 20
- refreshing policy data 16
- remote site recovery
 - for VSAM RLS 25
- RESPONSE response 69

S

- SA z/OS definitions for CICS 9
- SA z/OS operator commands
 - DISPTRG 56
 - INGREQ 25, 56
 - INGSCHED 56
- SDF (*see* status display facility) 65
- security checking 93
- segment support in NetView 77
- SEND
 - from CICS 73
 - from NetView 70
- shortcut keys xiii
- startup
 - types 56
- status display facility 65
- subsystems
 - monitoring 59
 - shutting down 59
 - starting 56
- system initialization table 94

T

- transactions
 - recovery 20
- transient data queue exit 12

V

- VSAM RLS 25

Readers' Comments — We'd Like to Hear from You

System Automation for z/OS
CICS Automation Programmer's Reference and Operator's Guide
Version 3 Release 2

Publication No. SC33-8267-04

We appreciate your comments about this publication. Please comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this book. The comments you send should pertain to only the information in this manual or product and the way in which the information is presented.

For technical questions and information about products and prices, please contact your IBM branch office, your IBM business partner, or your authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you. IBM or any other organizations will only use the personal information that you supply to contact you about the issues that you state on this form.

Comments:

Thank you for your support.

Submit your comments using one of these channels:

- Send your comments to the address on the reverse side of this form.
- Send a fax to the following number: FAX (Germany): 07031+16-3456
FAX (Other Countries): (+49)+7031-16-3456
- Send your comments via e-mail to: s390id@de.ibm.com

If you would like a response from IBM, please fill in the following information:

Name

Address

Company or Organization

Phone No.

E-mail address



Fold and Tape

Please do not staple

Fold and Tape



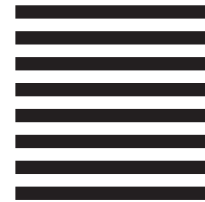
NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

IBM Deutschland Entwicklung GmbH
Department 3248
Schönaicher Strasse 220
D-71032 Böblingen
Federal Republic of Germany
72031-0000



Fold and Tape

Please do not staple

Fold and Tape



Program Number: 5698-SA3

Printed in USA

SC33-8267-04

