

System Automation for z/OS  
Version 4 .Release 1

*Product Automation  
Programmer's Reference and Operator's  
Guide*



**Note!**

Before using this information and the product it supports, read the information in [Appendix B, “Notices,”](#) on page 107.

**Edition Notes**

This edition applies to IBM® System Automation for z/OS (Program Number 5698-SA4) Version 4 Release 1, an IBM licensed program, and to all subsequent releases and modifications until otherwise indicated in new editions or technical newsletters.

This edition replaces SC34-2714-00.

© **Copyright International Business Machines Corporation 1990, 2017.**

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

---

# Contents

- Figures..... vii**
- Tables..... ix**
- Accessibility..... xi**
  - Using assistive technologies..... xi
  - Keyboard navigation of the user interface..... xi
- About this publication..... xiii**
  - Who Should Use This Publication..... xiii
  - What’s in This Publication..... xiii
  - Related Publications..... xiii
    - The System Automation for z/OS Library..... xiii
    - Related Product Information..... xiv
  - Summary of Changes for SC34-2714-01..... xiv
    - Changed Information..... xv
- Part 1. CICS Automation..... 1**
  - Chapter 1. CICS Automation Concepts..... 3
    - \*CICS Best Practices Policy..... 3
    - Customization Hints..... 3
    - Operation Hints..... 3
    - Using the CICS Automation Message Exit..... 4
      - Defining CICS Messages..... 4
      - Defining the Reload Command..... 8
      - Refreshing Policy Data..... 8
      - Partial Message IDs and Performance..... 8
  - Chapter 2. How to Set Up the Functions of CICS Automation..... 9
    - Automating Recovery for Application Components..... 9
      - How to Define Transaction Recovery..... 10
      - Monitoring of CICS - External Subsystem Connections..... 11
    - Event Monitoring..... 13
      - Setting Up an Event Monitor Resource..... 13
      - Relationships..... 15
    - Health Monitoring..... 15
    - Link Monitoring..... 15
      - CICS Link Monitor Resource..... 15
      - Code Definitions for Message ING150I..... 16
      - Command Definitions for Message ING150I..... 16
      - Relationships..... 16
  - Chapter 3. MESSAGES/USER DATA Entries for CICS Automation..... 17
    - CICS-Specific MESSAGES/USER DATA Keywords..... 17
      - ABCODETRAN: Transaction Abend Recovery..... 17
      - CICSINFO: Display Information..... 18
      - LISTSHUT: Transaction Purging During Shutdown..... 19
      - RCVRSOS: Short-On-Storage Handling..... 20

RCVRTRAN: Transaction Recovery.....	21
Chapter 4. CICS Automation Routines.....	23
CICSPURG: Purge Transactions.....	23
EVEERDMP: CICS Dump.....	24
EVEES100: Recovery Function for Short-On-Storage Conditions.....	24
EVEERTRN.....	24
Chapter 5. INGCICS Operator Command.....	27
INGCICS: Issue List of Defined Transactions and View the Output.....	27
Chapter 6. Monitor Command.....	29
INGRMADB Routine for the Monitoring of CICS - External Subsystem Connections.....	29
<b>Part 2. DB2 Automation.....</b>	<b>31</b>
Chapter 7. DB2 Automation for System Automation for z/OS.....	33
*DB2 Best Practices Policy.....	33
Customization Hints.....	33
Access Authority for Automation Work Operators.....	33
Critical Event Monitoring.....	34
INGDB2 Automation Utility.....	36
<b>Part 3. IMS Automation.....</b>	<b>37</b>
Chapter 8. IMS Automation Concepts.....	39
*IMS Best Practices Policy.....	39
Customization Hints.....	39
IMS Operation Hint.....	39
IMS Message Processing.....	40
Installing the IMS Message Exits.....	40
Using the IMS Automation Message Exit.....	42
Defining IMS Actions.....	44
Example.....	44
Chapter 9. How to Set Up the Special Functions of IMS Automation.....	45
Automating Recovery for Application Components.....	45
How to Define Transaction Recovery.....	45
Monitoring of Online Log Data Sets (OLDS).....	47
Monitoring of Recovery Control Data Sets (RECON).....	50
Monitoring of VTAM ACB.....	51
Monitoring of IMS - External Subsystem Connections.....	52
Chapter 10. MESSAGES/USER DATA Entries for IMS Automation.....	55
IMS-Specific MESSAGES/USER DATA Keywords.....	55
ABCODEPROG: Respond to IMS Application Abends.....	55
ABCODETRAN: Transaction Abend Recovery.....	56
DFS554A: Respond to Program Abend.....	57
IMSINFO: Display Information.....	57
TCO: Allocate data set with Commands for Time-Driven Procedures.....	58
TCOMEMBERS: Define TCO Members.....	59
Chapter 11. INGIMS Operator Command.....	61
INGIMS: Issue List of Defined Transactions and View the Output.....	61
Chapter 12. Monitor Commands.....	63
INGRMIDB Routine for the Monitoring of IMS - External Subsystem Connections.....	63

INGRMIOI Routine for OLDS Monitoring.....	63
INGRMIRE Routine for RECON Monitoring.....	65
INGRMIDC Routine for DC Monitoring.....	66
Chapter 13. Using IMS Automation.....	67
Using the Main Menu.....	67
Selecting and Viewing Subsystems.....	68
Selecting a Subsystem.....	68
Getting Detailed Status.....	69
Working with TCO Functions.....	70
Load a Specific Member.....	71
Start or Stop TCO Processing.....	71
Displaying IMS Dependent Region Information.....	71
Chapter 14. IMS Automation Routines.....	77
EVIECTOX.....	77
EVISTRCT.....	77
EVISTRMN.....	78
<b>Part 4. OMEGAMON Automation.....</b>	<b>79</b>
Chapter 15. *ITM Best Practices Policy.....	81
<b>Appendix A. CICS Automation and the Program-to-Program Interface.....</b>	<b>83</b>
The Program-to-Program Interface.....	83
NetView Components.....	83
CICS Components.....	83
Communication Components.....	84
Program-to-Program Interface Components in NetView and CICS.....	84
NetView Requests Using the Program-to-Program Interface.....	84
CONVERSE from NetView.....	85
SEND from NetView.....	86
CANCEL from NetView.....	87
CICS Requests Using the Program-to-Program Interface.....	87
CONVERSE from CICS.....	88
SEND from CICS.....	89
Programming Interface.....	89
EVESNCCI: NetView to CICS Communication Interface.....	90
EVESCCCI: CICS to NetView Communication Interface.....	94
EVEMPINT: EVESCCCI Parameter List Copy Book.....	97
Customizing CICS Definitions.....	102
Step 1: Modifications to Program-to-Program Interface Initialization.....	102
Step 2: Define a NetView PPI Receiver Task.....	103
Step 3: Define a CICS PPI Receiver Task.....	103
Definition Members.....	103
Security Checking Using CICS.....	106
<b>Appendix B. Notices.....</b>	<b>107</b>
Trademarks.....	108
Terms and conditions for product documentation.....	108
<b>Glossary of IMS, CICS, and Other Terms.....</b>	<b>111</b>
<b>Index.....</b>	<b>119</b>



---

# Figures

- 1. Example IMS Automation TCO Member Load Pop-up Panel..... 59
- 2. IMS Automation Main Menu.....67
- 3. Selection Panel for IMS Resources..... 69
- 4. Detailed Subsystem Information Panel for an IMS Control Region..... 70
- 5. IMS TCO Status Panel..... 71
- 6. IMS Dependent Regions Panel..... 72
- 7. Program-to-Program Interface Components in NetView and CICS..... 84
- 8. An EVESNCCI CONVERSE Request..... 86
- 9. An EVESNCCI SEND Request..... 87
- 10. An EVESCCCI CONVERSE Request..... 88
- 11. An EVESCCCI SEND Request..... 89





---

# Tables

1. System Automation for z/OS library.....	xiii
2. Keywords in CICS Message Definitions.....	4
3. Message IDs and Severity for EVEES100.....	24
4. Messages captured for INGDB2 utility.....	36
5. Keywords in IMS Message Definitions.....	42
6. EVESCCCI CONVERSE Request Parameter List.....	94
7. EVESCCCI Fields Returned to Caller from CONVERSE Request.....	95
8. EVESCCCI SEND Request Parameter List.....	96



# Accessibility

---

Accessibility features help users with physical disabilities, such as restricted mobility or limited vision, to use software products successfully. System Automation for z/OS supports several user interfaces. Product functionality and accessibility features vary according to the interface.

The major accessibility features in this product enable users in the following ways:

- Use assistive technologies such as screen reader software and digital speech synthesizer, to hear what is displayed on screen. Consult the product documentation of the assistive technology for details on using those technologies with this product and screen magnifier software
- Operate specific or equivalent features using only the keyboard
- Magnify what is displayed on screen.

The product documentation includes the following features to aid accessibility:

- All documentation is available to both HTML and convertible PDF formats to give the maximum opportunity for users to apply screen-reader software
- All images in the documentation are provided with alternative text so that users with vision impairments can understand the contents of the images.

## 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 1* 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.



## About this publication

---

This publication describes how to customize and operate product automation components (CICS®, DB2®, and IMS automation) with IBM System Automation for z/OS (SA z/OS) to provide a simple and consistent way to monitor and control all of the CICS, DB2, and IMS regions, both local and remote, within your organization. SA z/OS automates, simplifies, and standardizes console operations and the management of component, application, and production-related tasks.

## Who Should Use This Publication

---

This publication is intended for two kinds of users or user groups:

- System programmers, system designers, and application designers who will automate CICS, DB2, and IMS using SA z/OS.

Installing and customizing CICS, DB2, and IMS automation requires a programmer's understanding of NetView®, CICS, DB2, IMS, and SA z/OS, because most of the definitions take place 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, DB2, and IMS subsystems.

For operators, a working knowledge of CICS, DB2, and IMS will be assumed.

## What's in This Publication

---

This publication contains the following:

- Part 1, "CICS Automation," on page 1. This part describes the principal concepts of SA z/OS with regard to CICS Automation and describes the steps that are necessary to customize and set up CICS Automation.
- Part 2, "DB2 Automation," on page 31. This part gives an overview of the functions provided by DB2 Automation and how to set them up.
- Part 3, "IMS Automation," on page 37. This part describes the principal concepts of SA z/OS with regard to IMS Automation and describes the steps that are necessary to customize and set up IMS Automation. It also describes the tasks of the operator who manages IMS subsystems through IMS Automation.

## Related Publications

---

### The System Automation for z/OS Library

Table 1 on page xiii shows the information units in the System Automation for z/OS library. These manuals can be downloaded from [IBM Documentation](#).

<b>Title</b>	<b>Form Number</b>	<b>Description</b>
<i>Get Started Guide</i>	SC27-9532	This book is intended for SA z/OS beginners. It contains the information about early planning, configuring the product, making it secure, customizing your automation environment, and the basic operational tasks that you perform on a daily basis.

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

<b>Title</b>	<b>Form Number</b>	<b>Description</b>
<i>Planning and Installation</i>	SC34-2716	Describes SA z/OS new capabilities and how to plan, install, configure, and migrate SA z/OS.
<i>Customizing and Programming</i>	SC34-2715	Describes how to adapt the standard installation, add new applications to automation, write your own automation procedures, and add new messages for automated applications.
<i>Defining Automation Policy</i>	SC34-2717	Describes how to define and maintain the automation policy.
<i>User's Guide</i>	SC34-2718	Describes SA z/OS functions and how to use SA z/OS to monitor and control systems.
<i>Messages and Codes</i>	SC34-2719	Describes the problem determination information of SA z/OS, including messages, return codes, reason codes, and status codes.
<i>Operator's Commands</i>	SC34-2720	Describes the operator commands available with SA z/OS, including their purpose, format, and specifics of how to use them.
<i>Programmer's Reference</i>	SC34-2748	Describes the programming interfaces of SA z/OS and the definitions for the status display facility (SDF).
<i>End-to-End Automation</i>	SC34-2750	Describes the end-to-end automation adapter for z/OS and how it enables end-to-end automation and how it connects to Service Management Unite Automation.
<i>Service Management Unite Automation Installation and Configuration Guide</i>	SC27-8747	Describes how to plan, install, set up, configure, and troubleshoot Service Management Unite Automation.
<i>Product Automation Programmer's Reference and Operator's Guide</i>	SC34-2714	Describes how to customize and operate product automation components (CICS, Db2, and IMS automation) with SA z/OS to provide a simple and consistent way to monitor and control all of the CICS, Db2, and IMS regions, both local and remote, within your organization.
<i>TWS Automation Programmer's and Operator's Reference Guide</i>	SC34-2749	Describes how to customize and operate TWS Automation.

## Related Product Information

For information that supports System Automation for z/OS, visit the z/OS library in IBM Documentation (<https://www.ibm.com/docs/en/zos>).

## Summary of Changes for SC34-2714-01

This document contains information that was previously presented in System Automation for z/OS V3.R5.0 Product Automation, SC34-2714-00.

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.

## Changed Information

The following information has been updated:

### **Automating Recovery for Application Components**

The CICS Automation now enables monitoring of CICS connections to the MQ and DB2 external subsystems.

### **Monitoring of CICS - External Subsystem Connections**

This topic is updated largely, including the monitored object, monitor command, and status message etc.

### **Monitoring of IMS - External Subsystem Connections**

This topic is updated largely, including the monitored object, monitor command, and status message etc.

### **INGRMCDB Routine for the Monitoring of CICS - External Subsystem Connections**

The INGRMCDB routine now supports the monitoring of connections to MQ and DB2 external subsystems.

### **Pictures of SA z/OS Best Practice policies**

The pictures of SA z/OS Best Practice policies have been moved from the USS directory to Add-on policies.





---

# Part 1. CICS Automation

This part describes:

- The principal concepts of SA z/OS with regard to CICS Automation.
- The steps that are necessary to customize and set up CICS Automation. The description focuses on those steps that are specific to CICS Automation and have to be done in addition to the definitions of the basic functions of SA z/OS that are common to all subsystems.
- CICS-specific message IDs in the MESSAGES/USER DATA policy item and common routines that request information or perform tasks associated with CICS Automation.

It contains the following chapters:

- [Chapter 1, “CICS Automation Concepts,” on page 3](#)
- [Chapter 2, “How to Set Up the Functions of CICS Automation,” on page 9](#)
- [Chapter 3, “MESSAGES/USER DATA Entries for CICS Automation,” on page 17](#)
- [Chapter 4, “CICS Automation Routines,” on page 23](#)



## Chapter 1. CICS Automation Concepts

This information describes the principal concepts of CICS Automation in SA z/OS and gives some hints relating to its customization.

### \*CICS Best Practices Policy

SA z/OS supplies a sample policy, \*CICS, that provides best practice definitions for running CICS Automation.

Diagrams of the sample policies are provided as PDF files that are located in [Add-on policies](#).

You customize CICS Automation for your specific installation by modifying the policy database in the customization dialog. See "Importing Sample Add-On Policies" in *IBM System Automation for z/OS Defining Automation Policy* and the online documentation for the \*CICS best practices policy for details about how to import the \*CICS best practices policy and modify the definitions that this import adds to your policy database.

### Customization Hints

Bear in mind the following hints when customizing the \*CICS best practices policy:

- The \*CICS best practices policy supplies the automation operator definition that is required for CICS Automation. It is located in the CICS\_AUTO\_OPS policy object for the AOP entry type.

Automated Function	Operator ID	Message Classes
CICSMSTR	AUTCICS	

Make sure that this operator ID is defined as a NetView operator.

- All CICS regions must be defined to SA z/OS as APPLICATION objects. The \*CICS best practices policy provides you with a number of sample classes and instances of entry type APL that model CICS regions. Copy and modify them to your needs:
  - The names of the APL classes in \*CICS have the prefix C\_CICS
  - The names of the APL instances in \*CICS have the prefix CICS or CPSM
- For the following CICS components, the **Application Type** field must be set to CICS:
  - CICS regions
  - CICSplex<sup>®</sup> SM Address Space (CMAS)
  - CICS Transaction Server
  - CICSplex SM Web User Interface Server
- For applications of type CICS the Subtype field must be set to the appropriate value.
- CICS regions are shut down with the CEMT PERFORM SHUTDOWN command. However, for CMAS regions, the COSD command is used for shutdown.

### Operation Hints

In the \*CICS best practices policy, relationships have been set up, for example, from monitor resources to the CICS application group to make sure that the monitor resources can rely on all required CICS functions.

Be aware that the shutdown of a CICS component may not be performed due to these defined resource dependencies. You should therefore address the shutdown request to the CICS application group instead of a CICS component to avoid hanging shutdown requests.

## 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.

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 on page 4](#).

<b>Keyword</b>	<b>Description</b>
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 an integer number. For strings including '/', for example dates, each part of the string counts as a separate token. This means a date in the format 01/01/15 is treated as three words. This must be considered when calculating the offset.

Table 2. Keywords in CICS Message Definitions (continued)

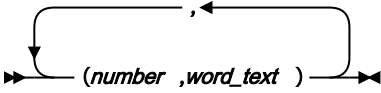
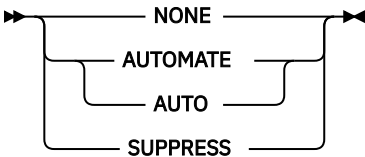
Keyword	Description
INSERT	<p>This keyword specifies the matching Insert values for CICS messages. It is optional and allows you to make decisions to control the message based on its content. Its format is:</p> <p>▶▶ (Tokens CICS Actions ) ▶▶</p> <p><b>Tokens</b></p>  <p><b>CICS Actions</b></p>  <p>Where <i>number</i> is the insert number to be checked and <i>word_text</i> is the text that is 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, including the national characters \$ (X'5B'), # (X'7B'), and @ (X'7C'). The values specified are not case sensitive.</p> <p>Example: Let us assume a message has an INSERT 7. The value of this INSERT can be a string made up of several words. However, the message should be issued as a WTO only when the string contains the values 'SET' and 'FORCEPURGE'. An INSERT with a value of 'SET TASK PURGETYPE(FORCEPURGE)' would qualify.</p> <p>Keyword INSERT could then be coded like this:</p> <pre>INSERT ((7,SET) (7,FORCEPURGE) AUTO)</pre> <p>It means that only when INSERT 7 has a match for values SET and FORCEPURGE, should the message be issued as a WTO.</p> <p>The actions that are available are:</p> <p><b>NONE</b> No actions are to be taken.</p> <p><b>AUTOMATE AUTO</b> WTO the message for SA z/OS to trap.</p> <p><b>SUPPRESS</b> 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>

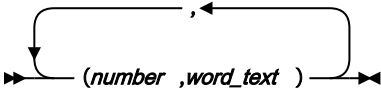
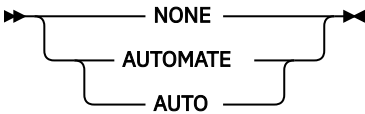
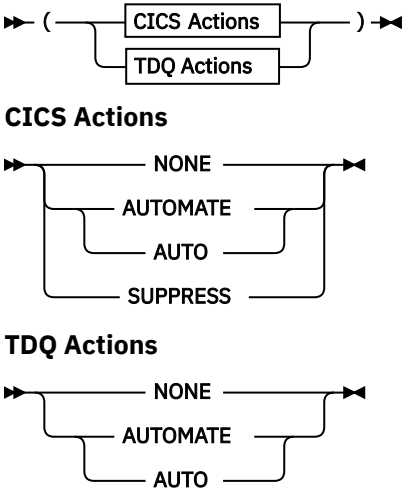
Table 2. Keywords in CICS Message Definitions (continued)	
Keyword	Description
TDQUEUE	<p>This keyword specifies the matching Transient Data Queue.</p> <p>Do not specify DFH messages that are already issued as a WTO with a TDQUEUE value.</p> <p>The format of the data is a single alphanumeric word up 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>
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 TDQ Actions ) ◄◄</p> <p><b>Tokens</b></p>  <p><b>TDQ Actions</b></p>  <p>Where <i>number</i> is the word number to be checked and <i>word_text</i> is the text to be checked. Words are counted from the beginning of the message starting from 1. A word is considered to be a contiguous set of characters that contain alphanumerics or any of the '\$#@_!?-:.,' characters. Any other characters are treated as word 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><b>NONE</b> Specifies that no action is to occur for the matching message.</p> <p><b>AUTOMATE AUTO</b> WTO the message for SA z/OS to trap.</p> <p>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>  <p><b>CICS Actions</b></p> <p><b>TDQ Actions</b></p> <p>The actions that are available are:</p> <p><b>NONE</b> 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><b>AUTOMATE AUTO</b> WTO the message for SA z/OS to trap.</p> <p><b>SUPPRESS</b> The message is suppressed from z/OS consoles. 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. 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.

**Note:** Even when a CICS message is also written to a Transient Data Queue, you must follow the instructions above. You cannot use the instruction below, which describes how to define a message to be WTO'd from a Transient Data Queue. For example, CICS message DFHAP1900 is also written to the transient data queue CADS. But to issue the message as a WTO, you cannot use keywords TDQUEUE and TOKEN. You must use keyword INSERT for this type of DFH messages.

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.

## Defining the Reload Command

To load the updated information into CICS address spaces whenever the automation configuration is refreshed, specify the following command in the REFRESHSTART phase in the STARTUP policy of the CICS address space:

```
MVS F &SUBSJOB, reload_transaction
```

The transaction SARL can be specified as *reload\_transaction* to reload the CICS message exit policy.

STARTUP 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 the effort to specify same data on each subsystem separately.

## 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, delete the messages from the policy.

## 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.



## Chapter 2. How to Set Up the Functions of CICS Automation

This information 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 *IBM System Automation for z/OS Defining Automation Policy*.

For how to define transactions and programs, see "Transaction and Program Definitions" in *IBM System Automation for z/OS Planning and Installation*.

### Automating Recovery for Application Components

CICS Automation provides automation for transaction recovery and short-on-storage conditions, and also enables monitoring of CICS connections to the external subsystems DB2® and IBM® MQ.

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

#### MINOR RESOURCE FLAGS

Minor resources are any resources specified to be subsidiary to an application, for example, messages and statuses.

With minor resource 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. The names of these minor resources must be as follows:

Problem area	Minor resource name
Short-on-storage conditions	RCVRSOS
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	(see <a href="#">“RCVRSOS: Short-On-Storage Handling”</a> on page 20)
Transaction recovery	(see <a href="#">“ABCODETRAN: Transaction Abend Recovery”</a> on page 17), (see <a href="#">“RCVRTRAN: Transaction Recovery”</a> on page 21)

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 "MINOR RESOURCES Policy Item" in *IBM 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 RESOURCES 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	Critical		Frequent		Infrequent	
	Number	Interval (hh:mm:ss)	Number	Interval (hh:mm:ss)	Number	Interval (hh:mm:ss)
CICS1.TRAN.PAYR	2	01:00:00	2	05:00:00	2	24:00:00

For more details, see "MINOR RESOURCES Policy Item" in *IBM System Automation for z/OS Defining Automation Policy*.

## Selecting the Abend Codes

The abend codes that recovery is to take place for 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.PAYR entry in the Message Processing panel and associate codes with this entry as follows:

Code 1	Code 2	Code 3	Value Returned
*	AEI0	*	INCLUDE
*	AKC3	*	INCLUDE
*	*	*	EXCLUDE

For more details, see [“ABCODETRAN: Transaction Abend Recovery”](#) on page 17.

## Specifying Recovery Actions

Use the RCVRTRAN keyword in the MESSAGES/USER DATA policy item to specify the actions to be taken when transaction abends occur.

For more details, see [“RCVRTRAN: Transaction Recovery”](#) on page 21.

## Monitoring of CICS - External Subsystem Connections

CICS Automation allows the monitoring of CICS connections to the external subsystems DB2<sup>®</sup> and IBM<sup>®</sup> MQ. The CICS command CEMT INQUIRE DB2CONN or CEMT INQUIRE MQCONN is issued regularly after each monitor interval to query the status of the CICS DB2 or CICS MQ connection.



**Attention:** Connections between CICS and MQ can only be monitored in this way using the IBM<sup>®</sup> MQ – CICS adapter. Connections via the IBM<sup>®</sup> MQ – CICS bridge cannot be monitored with CICS automation. For more information about the setup of the adapter, see [Setting up the CICS-WebSphere MQ adapter](#) in the CICS Transaction Server for z/OS<sup>®</sup> Version 5.3 product documentation.

The CICS - external subsystem connection monitoring function is optional, that is, definitions are necessary to enable it.

Implementation of the CICS - external subsystem connection monitoring function includes the following:

- A monitor resource for the connection to be monitored between a CICS and a DB2<sup>®</sup> or IBM<sup>®</sup> MQ external subsystem
- Status messages
- Relationships

**Note:** You can also establish connection monitoring using CPSM.

## CICS - External Subsystem Connection Monitor Resource

The CICS - external subsystem connection monitor resource has to be defined in the MTR policy object with the following characteristics:

Field Name	Field Value
Monitored Object	External subsystem (DB2 or MQ), MVS <sup>™</sup> subsystem ID, <i>extsysid</i> , of the external subsystem or the external subsystem group name
Monitored Jobname	<i>jobname</i> of the CICS subsystem
Monitor command	INGRMCDB (DB2) / INGRMCDB MQ

Field Name	Field Value
Monitoring Interval	<i>hh:mm</i>

The specified monitor command INGRMCDB is executed after each monitoring interval and can be expanded with an additional parameter. This parameter can either be empty, specified with DB2 or with MQ and is needed to differentiate whether a connection to a DB2 external subsystem or a MQ external subsystem is analyzed.

If the additional parameter is not specified or is specified with DB2, INGRMCDB issues the CEMT INQUIRE DB2CONN command and analyzes the status of the connection between the CICS and a DB2 external subsystem in the response to this command.

If the additional parameter is specified with MQ, INGRMCDB issues the CEMT INQUIRE MQCONN command and analyzes the status of the connection between the CICS and an MQ external subsystem.

If an external subsystem ID or an external subsystem group name is specified in the Monitored Object field, the monitor command only analyzes the status of a connection to this specified target. If the external subsystem ('DB2' or 'MQ') is entered as the monitored object, any connection to an arbitrary external subsystem is accepted.

Depending on the monitor results, INGRMCDB ends with the following return codes, which are mapped to the related health status for the monitor resource:

Return Code	Health Status	Description
1	BROKEN	No external subsystem connection is defined in CICS
2	FAILED	CEMT INQUIRE DB2CONN or CEMT INQUIRE MQCONN command failed
3	NORMAL	CICS is connected to the external subsystem
5	MINOR	CICS is not connected to the external subsystem

### Status Messages

Apart from active monitoring by issuing the INGRMCDB monitor command after each monitoring interval, the CICS - external subsystem monitoring function of SA z/OS also performs passive monitoring by updating the health status whenever it receives one of the following status messages:

- DB2® messages:

```
DFHDB2023I date time applid The CICS-DB2 attachment has connected to
DB2 subsystem db2-id{ |group}db2-group
```

```
DFHDB2025I date time applid The CICS-DB2 attachment has disconnected
from DB2 subsystem db2-id{ |group}db2-group
```

```
DFHDB2037 date time applid DB2 {subsystem |group} db2-id {is not active.
|has no active members.} The CICS-DB2 attachment facility is waiting.
```

- IBM® MQ messages:

```
DFHMQ0307I date time applid Successful connection to queue manager
qmgr-name release vrrr { |group} qsg-name
```

```
DFHMQ0331I date time applid Adapter shutdown completed.
```

```
DFHMQ0334I date time applid Adapter shutdown successful.
```

These messages indicate whether the attempt to connect the CICS region *applid* to the DB2 external subsystem *db2-id* or whether the attempt to connect the CICS region to the MQ external subsystem was successful. In response to these messages, the health status is set to NORMAL if it was successful or MINOR if it was not.

## Relationships

To propagate the health status for the monitor resource to the health status of the application, you must define a HASMONITOR relationship between the CICS region and the monitor resource.

The monitor resource is kept active only during the UP time of the CICS application and the external subsystem application group. This is achieved with an additional HASPASSIVEPARENT and a FORCEDOWN/WhenObservedDown relationship between the monitor resource and both the CICS application and the external subsystem application group. This ensures that the monitor resource can rely on all required functions.

## Event Monitoring

CICS Automation of SA z/OS allows the monitoring of events that are issued by CICSplex System Manager (CICSplex SM).

When an event is received, the event severity is mapped to the health status of a related monitor resource and, in addition, defined recovery commands are issued.

The event monitoring function is optional, that is, some definitions are necessary to get it up and running. The implementation of the event monitoring function includes the following:

- The definition of a monitor resource for each event that is to be monitored
- Relationships

The event monitoring function requires the infrastructure for the event-based CICS link and health monitoring using CICSplex SM objects. For details, see "Health Monitoring using CICSplex SM" in *IBM System Automation for z/OS Customizing and Programming*.

## Setting Up an Event Monitor Resource

You must define a passive monitor resource in the policy database for each event that is to be monitored.

You must make these definitions for the monitored object that has the following naming convention:

```
prefix.target.resource.name
```

Where:

### **prefix**

The value of the PREFIX parameter of the INGCPST command. For details, see "INGCPST" in *IBM System Automation for z/OS Programmer's Reference*.

### **target**

The name of the CICS system that is the target of the event (as in the CICSplex SM EVENT view).

### **resource**

The type of resource for which the event was generated (as in the CICSplex SM EVENT view). For events that are generated automatically by CICSplex SM, the resource is ! !.

### **name**

The name of the event.

For events that are generated as a result of evaluating an analysis or status definition, the event name is the same as the definition name.

For events that are generated automatically by CICSplex SM, the name is one of the following (as in the CICSplex SM EVENT view):

- !!SAMOPS System unavailable
- !!SAMSDM System dump
- !!SAMTDM Transaction dump
- !!SAMSOS Short-on-storage
- !!SAMMAX Maximum tasks

- !!SAMSTL System stall

Make the following definitions for the monitored object:

1. Define codes for message ING150I with the MESSAGES/USER DATA policy item of the monitor resource to map the event severities to health states of the monitor resource.

Specify the CICSplex SM severity of the received events in the **Code 1** field. Use the first token of the **Value Returned** field to specify a recovery command that is to be executed. The specified token is used to select the associated command from the list of defined commands for this message. The second token is taken as the health status for the monitor resource. For example:

Code 1	Code 2	Code 3	Value Returned
VLS			* NORMAL
LS			* WARNING
LW			WCMD WARNING

2. Define recovery commands for the different CICSplex SM event severities. Define the commands for message ING150I with the MESSAGES/USER DATA policy item of the monitor resource.

Use the **Ps/Select** field of the Command Processing panel of message ING150I to correlate the command to the first token of the **Value Returned** in the code definition.

Ps/Select	AutoFn/*	Command Text
WCMD		MYRECV

The second token of the **Value Returned** field is the status of the new health status that the monitor has determined, for example:

**NORMAL**

The monitor has obtained good results from the object, or objects, that it is watching

**WARNING**

The monitor detected a certain degree of degradation in the operation of the monitored object.

Continuing with the example, the severities VLS and LS do not execute recovery commands because there are no commands defined for the selection \*. Severity LW however has the selection WCMD specified, which you probably have defined commands for.

For more details about code processing, see INGMON in *IBM System Automation for z/OS Programmer's Reference*.

**Example: CICS Link Monitor Resource**

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.

Monitoring such a link is a suitable example for event monitoring.

To monitor events for a connection with the name CON1 to the target CICS1TOR, specify the monitored object as:

```
CPSM.CICS1TOR.CONNECT.CON1
```

## Relationships

To propagate the health status of the monitor resource to the health status of the application, you must define a HASMONITOR relationship between the CICS region and the monitor resource.

The monitor resource is kept active only during the UP time of the CICS application with both of the following relationships:

- A HASPASSIVEPARENT and a FORCEDOWN/WhenObservedDown relationship between the monitor resource and the CICS region
- An additional HASPASSIVEPARENT relationship to the connected CICS region

The monitor resource depends on the CICSplex SM event listener with a HASPARENT and a FORCEDOWN/WhenObservedDown relationship. This ensures that the monitor resource can rely on all required functions.

## Health Monitoring

CICSplex SM employs the concept of *status programs*. You can make definitions so that a program is executed periodically and sets one of the CICSplex SM severities.

This can produce events that can be monitored with the monitoring infrastructure described in [“Event Monitoring”](#) on page 13.

The naming convention for the monitored object in this case is:

```
prefix.target.PROGRAM.name
```

## 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.

CICS Automation of SA z/OS allows the monitoring of those links or connections. This is done by registering 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, the event severity is mapped to the health status of a relating monitor resource and in addition, defined recovery commands are issued.

The link monitoring function is optional, that is, some definitions are necessary to get it up and running.

The implementation of the link monitoring function includes the following:

- The definition of a monitor resource for each link
- Code definitions for message ING150I
- Command definitions for message ING150I
- Relationships

The link monitoring function requires the infrastructure for the event-based CICS link and health monitoring by using CICSplex System Manager (CICSplex SM) objects. For details, see "Health Monitoring using CICSplex SM" in *IBM System Automation for z/OS Customizing and Programming*.

## CICS Link Monitor Resource

A link monitor resource has to be defined in the MTR policy object with the following characteristic:

Field Name	Field Value
Monitored Object	Name of the monitored CPSM object according to the naming conventions, for example, CPSM.CICS1TOR.CONNECT.CON1

## Code Definitions for Message ING150I

Define codes for message ING150I with the MESSAGES/USER DATA policy item of the monitor resource to map the event severities to health states of the monitor resource.

Specify the CPSM severity of the received events in the **Code 1** field. Use the first token of the **Value Returned** field to specify a recovery command to be executed. The specified token is used to select the relating command from the list of defined commands for this message. The second token is taken as the health status for the monitor resource. For example:

Code 1	Code 2	Code 3	Value Returned
VLS	*	*	* NORMAL
LS	*	*	* WARNING
LW	*	*	* WARNING

## Command Definitions for Message ING150I

Define recovery commands for the different CPSM event severities. The commands are to be defined for message ING150I using the MESSAGES/USER DATA policy item of the monitor resource. Use the selection field to correlate the command to the first token of the **Value Returned** field in the code definition.

## Relationships

To propagate the health status of the monitor resource to the health status of the application, you must define a HASMONITOR relationship between the CICS region and the monitor resource.

The monitor resource is kept active only during the UP time of the CICS application with a HASPASSIVEPARENT and a FORCEDOWN/WhenObservedDown relationship between the monitor resource and the CICS region and an additional HASPASSIVEPARENT relationship to the connected CICS region. The monitor resource depends on the CPSM event listener with a HASPARENT and a FORCEDOWN/WhenObservedDown relationship. This ensures that the monitor resource can rely on all required functions.



## Chapter 3. 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 that this information must be entered in. There are, however, some CICS application-specific automation parameters that must be specified as entries in the MESSAGES/USER DATA policy item of the appropriate application.

This chapter contains detailed descriptions of these automation entries. However, a general understanding of the MESSAGES/USER DATA policy item is assumed. For details, see "MESSAGES/USER DATA policy item" in *IBM System Automation for z/OS Defining Automation Policy*.

### 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
"ABCODETRAN: Transaction Abend Recovery" on page 17.	Use this ID to define actions to be taken for transaction abend codes.
"CICSINFO: Display Information" on page 18.	Use this ID to define the effect of INGCICS REQ=INFO.
"LISTSHUT: Transaction Purging During Shutdown" on page 19.	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 20.	Use this ID if you want CICS Automation to take action for short on storage conditions.
"RCVRTRAN: Transaction Recovery" on page 21.	Use this ID to define actions to be taken when this specific transaction has abended.

### ABCODETRAN: Transaction Abend Recovery

Use this keyword to specify whether to initiate recovery actions for certain transactions, abend codes, and programs.

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 attributes will only apply to this transaction.

##### CODE

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

##### *tran*

The transaction ID.

## CICSINFO: Display Information

### ***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 actions to be taken if recovery is allowed are defined with the RCVRTRAN keyword.
2. The transaction name is specified either in the keyword ABCODETRAN.*tran* or as the value of the first 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.
3. If a match is not found when checking the code definitions of ABCODETRAN[.*tran*], the value INCLUDE is assumed as the default value.
4. Ensure that the abend transaction message is issued as a WTO. For more details, see [“Defining CICS Messages” on page 4](#)

## Example 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 transactions and abend codes.

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

## 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 System Automation for z/OS Operator's Commands*.

Use User-Defined Processing and enter the following data:

Keyword	Data
CICSCMD	( <i>description</i> , <i>CICS_command</i> )

## Keyword and Parameter Definitions

### ***description***

This is text that is placed before the output of *CICS\_command*. It can be used to identify the command output in the output stream. The description can be any string, but must be enclosed in quotation marks.

### ***CICS\_command***

This is the command that is to be executed. It is appended to a MODIFY subsystem and issued as an MVS command to an EMCS console. The output is collected and displayed. The command can be any

valid CICS transaction that writes to an MVS console. The command must be enclosed in quotation marks.

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 REQ=INFO command and also for function key PF10 of the DISPINFO panel.

### Examples of Usage

Keyword	Data
CICSCMD	('DISPLAY ACTIVE TASKS', 'CEMT I TA')
CICSCMD	('DISPLAY SYSTEM INFORMATION', 'CEMT I SYS')

## 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 23.

## RCVRSOS: Short-On-Storage Handling

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

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:

Ps/Select	AutoFn/*	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 command list, EVEERDMP, that can be specified for the CMD attribute when a dump is to be produced.
2. The critical value of the RCVRSOS thresholds of the subsystem is used to determine how often the specified commands are allowed to be issued within a specific time frame. The RCVRSOS thresholds must be defined in the customization dialog with the application's MINOR RESOURCES policy item.
3. Pass processing is supported. The pass counter is reset as soon as the short-on-storage situation is reported to be relieved.
4. The defined commands for message ID RCVRSOS are only issued if, in addition, command EVEES100 is defined to be issued in response to messages DFHSM0131, DFHSM0132, DFHSM0133 and DFHSM0134.

It is, however, advisable to replace the technique for the automation of short-on-storage situations with event-based CICSplex monitoring.

## 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:

Ps/Select	AutoFn/*	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:

Ps/Select	AutoFn/*	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 RESOURCES 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 called from the NetView automation table by the following messages:
  - DFHAC2231
  - DFHAC2232
  - DFHAC2233
  - DFHAC2236
  - DFHAC2245
  - DFHAC2247
  - DFHAC2248
  - DFHAC2249
  - DFHAC2250
  - DFHAC2251

## RCVRTRAN: Transaction Recovery

- DFHAC2252
- DFHAC2253

Ensure that these abend transaction messages are issued as a WTO. For more details, see [“Defining CICS Messages”](#) on page 4.

### Examples of Usage

The following command definition for message ID RCVRTRAN indicates that the command shown is to be executed for all transactions that do not have a specific command definition for message ID RCVRTRAN.*transid*.

Ps/Select	AutoFn/*	Command Text
		MSG OP1,TRAN &EHKVAR1 FAILED

The following example shows the user definitions that are needed to issue a WTO for the abend transaction messages listed in [usage note 3](#):

Keyword	Data
TDQUEUE	CSMT
OFFSET	1

## Chapter 4. CICS Automation Routines

This section is intended to help system administrators to use CICS Automation functions.

It describes the following routines that should be executed from the automation policy:

### **CICSPURG**

Purge transactions at CICS shutdown, see [“CICSPURG: Purge Transactions”](#) on page 23.

### **EVEERDMP**

Creates a system DUMP of the CICS address space, see [“EVEERDMP: CICS Dump”](#) on page 24.

### **EVEES100**

Execute defined recovery actions for short-on-storage conditions, see [“EVEES100: Recovery Function for Short-On-Storage Conditions”](#) on page 24.

### **EVEERTRN**

Handles transaction recovery, see [“EVEERTRN”](#) on page 24.

## CICSPURG: Purge Transactions

This command purges running transactions at the time it is issued.

It consults the LISTSHUT MESSAGES/USER DATA policy item (see [“LISTSHUT: Transaction Purging During Shutdown”](#) on page 19) to determine which transactions are eligible for purging.

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 shut down.

### Format

```
CICSPURG [subsys]
```

### Keyword and Parameter Definitions

#### ***subsys***

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

### Comments and Usage Notes

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

### Examples of Usage

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

Ps/Select	AutoFn/*	Command Text
1		MVS F &SUBSJOB,CEMT PERFORM SHUTDOWN
2		CICSPURG

## EVEERDMP: CICS Dump

EVEERDMP creates 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 job name for this CICS.

#### *subsys*

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

### Examples of Usage

Ps/Select	AutoFn/*	Command Text
		EVEERDMP &SUBSAPPL

## EVEES100: Recovery Function for Short-On-Storage Conditions

This routine performs CICS short-on-storage recovery actions that are defined for message ID RCVR SOS.

### Format

```
EVEES100
```

### Comments and Usage Notes

This automation routine can be defined in the automation policy to respond to the following messages for CICS short-on-storage events:

<i>Table 3. Message IDs and Severity for EVEES100</i>	
Message ID	Severity
DFHSM0131	CRITICAL
DFHSM0132	NORMAL
DFHSM0133	CRITICAL
DFHSM0134	NORMAL

It can also be invoked from the NetView automation table.

## EVEERTRN

The EVEERTRN routine handles transaction recovery. It captures each message listed below with severity NORMAL.

### Format

```
➤ EVEERTRN ➤
```



## Comments and Usage Notes

The EVEERTRN automation routine is intended to respond to the following messages by issuing the transaction recovery commands that are defined for the RCVRTRAN message ID:

```

DFHAC2231 date time applid Transaction tranid running program program name term
          termid has lost contact with its coordinator system during syncpoint and
          has abended with code ASP1. The unit of work is shunted until contact is
          restored}. EXCI job = {exci_id. condmsg
DFHAC2232 date time applid Transaction tranid running program program name term
          termid has lost contact with its coordinator system during syncpoint and
          has abended with code ASP0. All updates will be unilaterally
          committed}. EXCI job = {exci_id. condmsg
DFHAC2233 date time applid Transaction tranid running program program name term
          termid has lost contact with its coordinator system during syncpoint and
          has abended with code ASPP. All updates will be unilaterally backed
          out}. EXCI job = {exci_id. condmsg
DFHAC2236 date time applid Transaction tranid abend secondary abcode in program
          program name term termid. Updates to local recoverable resources will be
          backed out}. EXCI job = {exci_id. condmsg
DFHAC2245 date time applid A CICS-generated syncpoint request could not be completed
          normally because a connected system has requested that the unit of work be
          rolled back. Transaction tranid running program program name term termid
          has been abnormally terminated with code ASPF}. EXCI job = {exci_id. condmsg
DFHAC2247 date time applid Transaction tranid running program program name term
          termid has requested rollback, but was using a type of processing for which
          rollback is not supported. The transaction has been abnormally terminated
          with code ASP8 {. EXCI job = {exci_id. condmsg
DFHAC2248 date time applid Transaction tranid running program program name term
          termid has failed with abend ASP7 following the failure of a local
          resource owner in the prepare phase of syncpoint. Updates will be backed
          out}. EXCI job = {exci_id. condmsg
DFHAC2249 date time applid Transaction tranid running program program name term
          termid has failed with abend ASP7 following the failure of a remote system
          in the prepare phase of syncpoint. Updates will be backed
          out}. EXCI job = {exci_id. condmsg
DFHAC2250 date time applid the coordinator system has indicated that the current unit of work
          is to be backed out. Transaction tranid running program program name term
          termid has been abnormally terminated with abend ASP3}. EXCI job = {exci_id.
          condmsg
DFHAC2251 date time applid Transaction tranid running program program name term
          termid has failed with abend ASPQ. Syncpoint commit processing has failed
          while communicating with a remote system}. EXCI job = {exci_id. condmsg
DFHAC2252 date time applid Transaction tranid in program program name term termid
          has lost contact with its coordinator system during syncpoint processing.
          No updates have been performed by this system; it has abended with code
          ASPR}. EXCI job = {exci_id. condmsg
DFHAC2253 date time applid Transaction tranid running program program name term
          termid has failed with abend ASP2 due to the links to the remote systems
          being in an invalid state. Updates will be backed out}. EXCI job = {exci_id.
          condmsg

```

Note that these messages are not normally issued to the system console, so if transaction recovery is required the messages should be included in the MESSAGES/USER DATA policy so that the CICS message exit forces CICS to WTO them.

For more details, see [“How to Define Transaction Recovery”](#) on page 10.



---

## Chapter 5. INGCICS Operator Command

INGCICS is the CICS-related SA z/OS operator command that can be invoked by operators or via PIPEs.

---

### INGCICS: Issue List of Defined Transactions and View the Output

---

#### **Purpose**

The INGCICS command lets you:

- Issue any console-enabled CICS transaction
- Broadcast messages to all or selected CICS users
- Issue a list of defined transactions and view the output
- Display the output of CICS transactions in full-screen or pipeable line mode

For a detailed description, see INGCICS command in *IBM System Automation for z/OS Operator's Commands*.



## Chapter 6. Monitor Command

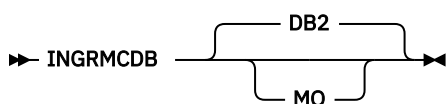
The following routine should be used as a MONITOR command when defining monitor resources for CICS Automation.

### INGRMCDB Routine for the Monitoring of CICS - External Subsystem Connections

This routine should be used as the monitor command for a monitor resource. It is issued after each monitoring interval to check whether CICS is connected to a DB2® or IBM® MQ external subsystem. To achieve this, the CICS command CEMT INQUIRE DB2CONN or CEMT INQUIRE MQCONN is issued to query the status of the CICS DB2 or CICS MQ connection.

The health status is considered to be NORMAL when the status of the monitored CICS - external subsystem connection is Connected. A degraded health status is returned in all other cases.

#### Format



#### Parameters

##### DB2

This enables the monitoring for a connection to a DB2 external subsystem and is the default value assumed, if nothing is specified with INGRMCDB. In this case, INGRMCDB issues the CEMT INQUIRE DB2CONN command and analyzes the status of the connection between the CICS and a DB2 external subsystem in the response to this command.

##### MQ

This enables the monitoring for a connection to an MQ external subsystem. In this case, INGRMCDB issues the CEMT INQUIRE MQCONN command and analyzes the status of the connection between the CICS and an MQ external subsystem in the response to this command.

#### Restrictions

The INGRMCDB routine is assumed to be defined as the monitor command for a monitor resource with the following restrictions:

- One of the following is specified in the Monitored Object field:
  - External subsystem ('DB2' or 'MQ')
  - The MVS subsystem ID of the external subsystem
  - The external subsystem group name
- A job name is specified for the monitor resource.
- The monitor resource is related to a CICS subsystem.

#### Return Codes

Depending on the monitor results, INGRMCDB ends with the following return codes, which are mapped to the related health status for the monitor resource.

<b>Return Code</b>	<b>Health status</b>	<b>Description</b>
1	BROKEN	No external subsystem connection is defined in CICS
2	FAILED	CEMT INQUIRE DB2CONN or CEMT INQUIRE MQCONN command failed
3	NORMAL	CICS is connected to the external subsystem
5	MINOR	CICS is not connected to the external subsystem

---

## Part 2. DB2 Automation

This part gives an overview of the functions provided by DB2 Automation and how to set them up. It also provides a more detailed description of the line command requests that can be performed against DB2 subsystems.

It contains the following chapter:

- [Chapter 7, “DB2 Automation for System Automation for z/OS,” on page 33](#)





## Chapter 7. DB2 Automation for System Automation for z/OS

This information describes the principal concepts of DB2 automation in SA z/OS and gives some hints concerning its customization.

### \*DB2 Best Practices Policy

SA z/OS supplies a sample policy, \*DB2, that provides best practice definitions for running DB2 automation. It shows a solution for a DB2 data sharing group with multiple DB2 subsystems that are restartable on another system in light mode.

Diagrams of the sample policies are provided as PDF files that are located in [Add-on policies](#).

You customize DB2 automation for your specific installation by modifying the policy database in the customization dialog. See "Importing Sample Add-On Policies" in *IBM System Automation for z/OS Defining Automation Policy* and the online documentation to the \*DB2 sample policy for details about how to import the \*DB2 best practices policy and modify the definitions that this import adds to your policy database.

### Customization Hints

Bear in mind the following hints when customizing the \*DB2 best practices policy:

- All DB2 address spaces must be defined to SA z/OS as APPLICATION objects. The \*DB2 best practices policy provides you with a number of sample classes and instances of entry type APL that model the DB2 subsystem and its dependent address spaces. Copy and modify them to tailor them to your needs:
  - The names of the APL classes in \*DB2 have the prefix C\_DB2
  - The names of the APL instances in \*DB2 have the prefix DB2
- For the following address spaces of DB2, the Application Type field must be set to DB2 and the appropriate subtype:
  - DB2 Master
  - DB2 Database Services
  - DB2 Distributed Data Facility
  - DB2 Resource Lock Manager
- Applications of type DB2 and subtype MSTR have an additional DB2-specific Policy Item, DB2 CONTROL, that allows you to specify the DB2 subsystem ID and parameters for controlling DB2 automation.

### Access Authority for Automation Work Operators

DB2 commands are issued by the SA z/OS automation work operator (AUTWRKnn) that the DB2 subsystem has been assigned to.

Because these operators are assigned automatically during ACF load each of the AUTWRKnn operators that are defined to automation should be granted SYSOPR authority to DB2.

When using OPERSEC=SAFCHK, or SAFDEF (user level security), the following list of operator IDs should also be granted SYSOPR authority to DB2:

- GSSOPER
- RPCOPER
- AUTINIT1

- SYSOPER (backup task for GSSOPER)
- BASEOPER (backup task for SYSOPER and GSSOPER)

## Critical Event Monitoring

---

### Purpose

Critical Event Monitoring handles specific critical events that may occur during normal day-to-day running of DB2. These include:

Message	Description
DSNB250E, DSNB311I, DSNB312I, DSNB320I, DSNB321I, DSNB322I, DSNB323I, DSNB350I, DSNB351I	Recover failed dataspace (for data sharing only)
DSNJ002I	Switch active log data sets
DSNR004I	RESTART...UR STATUS COUNTS
DSNP007I	Data set could not be extended
DSNJ110E	Last active log data set is % full
DSNJ111E	All active log data sets full
DSNJ115I	Archive data set could not be allocated
DSNT500I/DSNT501I	Resource unavailable

Recovery commands that are required should be defined for the message in the automation's MESSAGES/USER DATA policy item for any DB2 subsystem that requires DB2 critical event message recovery.

If the DB2 subsystem is known to SA z/OS as an application of type DB2, event message recovery can be controlled by parameters entered via the DB2 CONTROL policy item for the subsystem.

### Restrictions

Critical event monitoring is only done if the DB2 subsystem is defined to SA z/OS and if recovery is enabled for the invoking message ID minor resource.

For some recovery actions SYSOPR needs SYSCTRL authority.

### Usage

For each of the Event Monitoring processes, the input parameters are validated for accuracy and any errors that are found are logged and the process is terminated. The requested subsystem is then checked to determine if automation is enabled for the major resource, and also that recovery is enabled for the invoking message ID minor resource. Once these preliminary checks have been successfully completed, the requested recovery action is performed.

### **DSNB250E, DSNB311I, DSNB312I, DSNB320I, DSNB321I, DSNB322I, DSNB323I, DSNB350I, DSNB351I: Recover Failed Dataspace**

**Description:** You may want to start table spaces in response to these messages. To do this, specify the start command in the MESSAGES/USER DATA policy item for these messages in your automation policy as follows: INGDB2 TABLE &SUBSAPPL START

### **DSNJ002I: Switch Active Log Data Sets**

**Description:** If commands are defined in the automation policy item MESSAGES/USER DATA for this message to an application of type DB2, these commands are only issued when the triggering message is for the log data set that is specified by the "Active log data set name" in the DB2 CONTROL policy item.

If the application is not of type DB2, the defined commands are issued unconditionally.

### **DSNR004I: RESTART...UR STATUS COUNTS**

**Description:** If the message reports an INDOUBT counter greater than zero at the end of a DB2 restart process, ISSUEACT is called to issue commands that are defined in the automation policy item MESSAGES/USER DATA of the DB2 application for this message.

### **DSNP007I: Data Set Could Not Be Extended**

**Description:** The created automation table statement calls ISSUEACT with its code specifications as the parameters. The code values are extracted from the message text. The data set name is passed as the CODE1 value, the return code is passed as the CODE2 value, and the connection ID is passed as the CODE3 value. If a code match is found with the ID of the triggering message in policy item MESSAGES/USER DATA, the value that is returned is used to select and issue the related commands as defined in the automation policy.

### **DSNJ110E: Last Active Log Data Set Is % Full**

**Description:** If commands are defined in the automation policy item MESSAGES/USER DATA for this message to an application of type DB2, these commands are only issued when the message reports a percentage full figure that is equal to or greater than the critical threshold that is defined in the "Log full threshold" field of the DB2 CONTROL policy item.

If the application is not of type DB2, the defined commands are issued unconditionally.

### **DSNJ111E: All Active Log Data Sets Full**

**Description:** If commands are defined in the automation policy item MESSAGES/USER DATA for this message to an application of type DB2, these commands are only issued when the number of received messages within a time period exceeds a given threshold. The time period and the threshold can be entered via the DB2 CONTROL policy item in the "Active log alerts" field and the related "Threshold" field.

If the application is not of type DB2, the defined commands are issued unconditionally.

### **DSNJ115I: Archive Data Set Could Not Be Allocated**

**Description:** If commands are defined in the MESSAGES/USER DATA automation policy item for this message for an application of type DB2, these commands are only issued when the elapsed time since they were triggered by this message is greater than a given time interval, as specified in the **Log offload interval** field in the DB2 CONTROL policy item.

If the application is not of type DB2, the defined commands are issued unconditionally.

### **DSNT500I/501I: Generate DSNT500I/DSNT501I Alert**

**Description:** The created automation table statement calls ISSUEACT with its code specifications as the parameters. The code values are extracted from the message text. The name is passed as CODE1, the reason is passed as CODE2 and the type is passed as CODE3. If a code match is found with the ID of the triggering message in policy item MESSAGES/USER DATA, the value that is returned is used to select and issue the related commands as defined in the automation policy.

## INGDB2 Automation Utility

---

INGDB2 is an automation utility that can be invoked by operators or via the AT.

The INGDB2 utility lets you:

- Start or stop a table space
- Terminate active threads
- Inform TSO users that their thread is about to be terminated
- Check for indoubt threads

Based on the condition used the following messages are captured:

<i>Table 4. Messages captured for INGDB2 utility</i>	
<b>Message ID</b>	<b>Severity</b>
ING114	CRITICAL
ING129	IMPORTANT

For a detailed description of the INGDB2 utility, see INGDB2 in *IBM System Automation for z/OS Operator's Commands*.

---

## Part 3. IMS Automation

This part describes the principal concepts of SA z/OS with regard to IMS Automation and describes the steps that are necessary to customize and set up IMS Automation. The description focuses on those steps that are specific to IMS Automation and have to be done in addition to the definition of the basic functions of SA z/OS that are common to all subsystems. Furthermore, it contains reference sections for IMS-specific message IDs in the MESSAGES/USER DATA policy item and for common routines that request information or perform tasks associated with IMS Automation.

It also describes the tasks of the operator who manages IMS subsystems through IMS Automation.

It consists of the following chapters:

- [Chapter 8, “IMS Automation Concepts,” on page 39](#)
- [Chapter 9, “How to Set Up the Special Functions of IMS Automation,” on page 45](#)
- [Chapter 10, “MESSAGES/USER DATA Entries for IMS Automation,” on page 55](#)
- [Chapter 11, “INGIMS Operator Command,” on page 61](#)
- [Chapter 12, “Monitor Commands,” on page 63](#)
- [Chapter 13, “Using IMS Automation,” on page 67](#)



## Chapter 8. IMS Automation Concepts

This section describes the principal concepts of IMS Automation in SA z/OS and gives some hints relating to its customization.

### \*IMS Best Practices Policy

SA z/OS supplies a sample policy, \*IMS, that provides best practice definitions for running IMS Automation.

The \*IMS sample policy builds in FDR to provide best practices for IMS Automation in an IMSplex. FDR provides superior, sysplex-aware automated recovery within IMS. It is the recommended solution for high availability of IMS applications.

Diagrams of the sample policies are provided as PDF files that are located in [Add-on policies](#).

You customize IMS Automation for your specific installation by modifying the policy database in the customization dialog. See "Importing Sample Add-On Policies" in *IBM System Automation for z/OS Defining Automation Policy* and the online documentation for the \*IMS sample policy for details about how to import the \*IMS best practices policy and modify the definitions that this import adds to your policy database.

### Customization Hints

Bear in mind the following when customizing the \*IMS best practices policy:

- The \*IMS best practices policy supplies the automation operator definition that is required for IMS automation. It is located in the IMS\_AUTO\_OPS policy object for the AOP entry type.

Automated Function	Operator ID	Message Classes
IMSMSTR	AUTIMS	EVI*

Make sure that this operator ID is defined as a NetView operator.

- All IMS regions must be defined to SA z/OS as APPLICATION objects. The \*IMS best practices policy provides you with a number of sample classes and instances of entry type APL that model IMS regions. Copy and modify them to your needs:
  - The names of the APL classes in \*IMS have the prefix C\_IMS
  - The names of the APL instances in \*IMS have the prefix IMS
- For the following IMS regions, the Application Type field must be set to IMS:
  - IMS Control regions
  - IMS dependent regions for Fast Path and Message Processing regions
  - IMS DL/I Secondary Address Space
  - IMS Database Recovery Control
- Applications of type IMS have an additional IMS-specific policy item, IMS CONTROL, that allows you to specify the IMS subsystem ID and the IMS control region type and the IMSplex name.

### IMS Operation Hint

In the \*IMS best practices policy, relationships have been set up, for example, from monitor resources to the IMS application group, to make sure that the monitor resources can rely on all required IMS functions.

Be aware that the shutdown of the IMS control region may not be performed because of these defined resource dependencies. You should therefore address the shutdown request to the IMS application group instead of the IMS control region to avoid hanging shutdown requests.

## IMS Message Processing

SA z/OS can only automate messages that are issued via WTO. Most IMS system messages that are important are WTO'd, However, there are many IMS messages that are only logged internally.

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 IMS. In addition, your own user code might produce messages that are written to IMS. Some of these messages might be of interest in automation situations.

SA z/OS installs an exit to WTO these messages. SA z/OS installs an exit for the AOE Type 2 exit of IMS Control regions.

The messages that are WTO'd from this exit are defined in the MESSAGES/USER DATA policy for the subsystem or the subsystem class.

## Installing the IMS Message Exits

*IBM System Automation for z/OS Planning and Installation* details the basic installation steps to install the exits into IMS. This section details the various parameters and commands that can be used to control the exits.

### z/OS Exit Router Information

The out-of-the-box configuration of the exit uses the z/OS exit router to enable the user to specify exit modules at three exit points.

The exit program is EVIPVEX0 and has a pre-built alias of DFSAOE00. The exit points defined are:

#### 1. DFSAOE00.CMD

This is invoked whenever DFSAOE00 is called with AOE0FUNC=1 to initialize the routines and also when AOE0FUNC=2 is called with AOE0FLG2=X'80' (command entered at a terminal), X'20' (ICMD command) or X'10' (internal command).

#### 2. DFSAOE00.MSG

This is invoked whenever DFSAOE00 is called with AOE0FUNC=1 to initialize the routines and also when AOE0FUNC=2 is called with AOE0FLG2=X'08' (message segment).

#### 3. DFSAOE00.CMDRESP

This is invoked whenever DFSAOE00 is called with AOE0FUNC=1 to initialize the routines and also when AOE0FUNC=2 is called with AOE0FLG2=X'40' (command response segment).

Specification of routines to run at each exit point is done via PROGxx members of SYS1.PARMLIB or via the SETPROG EXIT command.

You may enable or disable exits dynamically at any time.

A required definition of EVIPVEX1 is needed to process the exit information for SA z/OS. However, any number of exit routines may be added at any of the points. The z/OS exit router will execute them one after the other. If exit routines are not specified for an exit point, no action will be taken.

The definitions of the required exit points are:

```
EXIT ADD EXITNAME(DFSAOE00.MSG) MODNAME(EVIPVEX1) DSNAME(ING.SINGMOD1) STATE(ACTIVE)
EXIT ADD EXITNAME(DFSAOE00.CMD) MODNAME(EVIPVEX1) DSNAME(ING.SINGMOD1) STATE(ACTIVE)
EXIT ADD EXITNAME(DFSAOE00.CMDRESP) MODNAME(EVIPVEX1) DSNAME(ING.SINGMOD1)
STATE(ACTIVE)
```

See *z/OS MVS Tuning and Initialization Reference* for additional parameters that can be supplied to the EXIT statement.

The SA z/OS exit routines EVIPVEX0 and EVIPVEX1 use the last three words of the storage pointed to by SXPLAWRK. If these values are changed unpredictable results may occur.



The SA z/OS exit router will set AOEORPLY to 1 if there are no exit routines enabled for an exit and a DFSAOUE0 module is present in the IMS system. This rule applies to each exit point independently. This means that the user can disable an exit point and still have their DFSAOUE0 module invoked. In addition if the exit point invoked is EVIPVEX1 as specified, see [“Using the SA z/OS Exit without the z/OS Exit Router”](#) on page 41 for details on invoking DFSAOUE0. Note that it is possible using the z/OS exit router to invoke DFSAOUE0 for all three exit points. This can be achieved by the default definition that does not define an exit point for DFSAOE00.CMDRESP or by disabling this exit point.

## Using the SA z/OS Exit without the z/OS Exit Router

As detailed in the installation manual, it is possible to use the SA z/OS exit in a standalone manner.

In this mode, no z/OS exit router is enabled and dynamic management of the exit is not possible. The procedure to enable this function is detailed in the System Automation installation manual. It basically re-defines the DFSAOE00 alias from the EVIPVEX0 module to the EVIPVEX1 module.

The SA z/OS exit will set AOEORPLY to 1 if there is a DFSAOUE0 module present in the IMS system and any of the following conditions are met:

- AOE0FUNC=2 and AOE0FLG2=AOE0MSGs
- AOE0FUNC=2 and AOE0FLG2=AOE0TCMD or AOE0ICMD or AOE0INTC

In effect this means that every system Message or Command will also invoke DFSAOUE0 if it is present. Command Responses will not invoke DFSAOUE0. It is not possible to invoke DFSAOUE0 for command responses in this mode of exit operation. See [“z/OS Exit Router Information”](#) on page 40 or [“Calling the SA z/OS Exit from Your DFSAOE00 Module”](#) on page 41 for alternatives.

Due to an IMS restriction, when DFSAOUE0 is present in the environment only the first segment of a multi-segment message will be presented to DFSAOE00. Therefore, tokens that are defined in the policy database must be present in the first segment to be matched. However, all segments will be passed to DFSAOUE0.

**Note:** Make sure that the ING.SINGMOD1 library is not concatenated ahead of the library that you have redefined the alias in. If it is, the z/OS exit router module will be enabled at the DFSAOE00 exit point.

## Calling the SA z/OS Exit from Your DFSAOE00 Module

If neither of the above methods suits your installation, it is possible to directly call the EVIPVEX1 module from your DFSAOE00 exit routine.

Input parameters are:

- Register 1 points to the SXPL (standard Exit Parameter List) as supplied by IMS to the DFSAOE00 exit.

A sample call is as follows:

```
LA   R1,SXPL           ; Load address of SXPL
L    R15,=V(EVIPVEX1) ; get address of routine
BALR R14,R15          ; invoke the exit
```

You are responsible for the correct setting of the AOEORPLY field. Note that after calling EVIPVEX1, the AOEORPLY field will be set as specified in [“Using the SA z/OS Exit without the z/OS Exit Router”](#) on page 41.

### Used storage

The DFSAOE00 exit uses the last two words of the storage that is pointed to by SXPLAWRK. The last word is reserved for internal SA z/OS control information. If the values are changed, unpredictable results may occur.

The EVIPVEX1 module uses the last three words of the storage that is pointed to by AOE0WRKA, the dynamic 256-byte work area of DFSAOE00. Make sure that DFSAOE00 does not use more than 244 bytes of this area. Otherwise, EVIPVEX1 will change the remaining buffer and hence unpredictable results may occur.

## Using the IMS Automation Message Exit

To enable the IMS message exit to process messages, the messages must be defined in the policy database. The MESSAGES/USER DATA policy item is used to define the messages.

If there are no messages defined to use the exits, the SA z/OS exit will not process any messages.

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

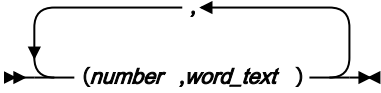
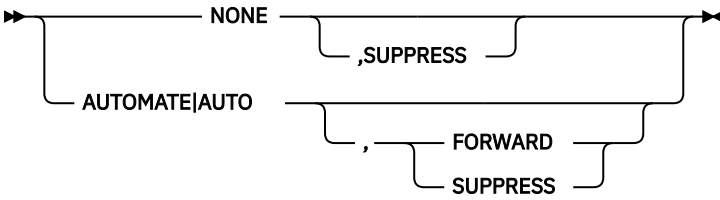
### Defining IMS Messages

There are several special keywords that, when added to the message with the USR action, cause the message information to be loaded into the appropriate exit for processing.

The keywords and their descriptions are described in [Table 5 on page 42](#).

Table 5. Keywords in IMS 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.
MSGDISP	<p>This keyword specifies the default actions for a message ID. The format of the data is as follows:</p> <p>➤ ( — <b>IMS Actions</b> — ) ➤</p> <p><b>IMS Actions</b></p> <p>The actions that are available are:</p> <p><b>NONE</b> The message is not acted on by the exit. Use this option to set a default of no actions when used with the TOKEN keyword.</p> <p><b>AUTOMATE AUTO</b> WTO the message for SA z/OS to trap.</p> <p><b>FORWARD</b> This forwards the message segments to the TYPE 1 exit DFSAOUE0, if it exists.</p> <p><b>Restriction:</b> IMS Actions AUTOMATE, FORWARD used in combination only work for single-segment messages.</p> <p><b>SUPPRESS</b> The message is suppressed from the IMS Master Console. IMS reserves the right not to suppress messages that it determines are critical.</p> <p>The default action is AUTOMATE if no TOKENs are present. If TOKENs are present the default action is NONE.</p> <p>FORWARD and SUPPRESS are mutually exclusive.</p>

Table 5. Keywords in IMS 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 IMSActions ) ▶▶</p> <p><b>Tokens</b></p>  <p><b>IMS Actions</b></p>  <p>Where <i>number</i> is the word number to be checked and <i>word_text</i> is the text to be checked. Words are counted from the beginning of the message starting from 1. A word is considered to be a contiguous set of characters that contain alphanumerics or any of the '\$#@_!?-:.,' characters. Any other characters are treated as word 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 the value is a single alphanumeric word; no blanks or special characters are allowed. The case of the value is ignored.</p>

To define a 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 the word number that the message ID occurs at in the message in the **Data** field.
3. 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.

## Refreshing Policy Data

To load the information into IMS address spaces issue the INGAMS REFRESH command. As a part of the ACF load, each IMS that had changes to MESSAGES/USER DATA policy will be reloaded with any changes.

If you want to disable the exits, delete the messages from the policy.

## 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. 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 you do not use the TOKEN keyword 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.

## Defining IMS Actions

You can specify MSGDISP policy information on 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.

## Defining IMS Actions

To define automated actions in the SA z/OS automation policy that have to be executed by IMS, it is recommended that you specify an IMS command, prefixed with the IMS subsystem ID, and issue it with the NetView MVS command.

The subsystem ID of the IMS control region can be retrieved from the &SUBSSUBID task global variable.

When specifying the command in the application policy object of a dependent IMS region that has a HasParent relationship to the IMS control region with a defined sequence number, the command can be prefixed with the &SUBPSUBID task global variable to address it to the associated IMS control region. You must make sure that the IMS control region is first in the list of defined parents.

If a Command Prefix has been defined in the APPLICATION INFO policy item for the IMS control region, you can also use the &SUBSCMDPFX task global variable to prefix the IMS command instead of the subsystem ID.

## Example

The POSTSTART commands for the IMS control region can be specified as follows:

Type	AutoFn/*	Command Text
		MVS &SUBSSUBIDCQSET SHUTDOWN SHAREDQ ON STRUCTURE ALL
		MVS &SUBSSUBIDSTA DC
		MVS &SUBSSUBIDCHE

IMS actions can also be defined to be issued as replies to the outstanding WTOR of the IMS control region. This method is only applicable to the DC control region because the DB control region does not allow communication via an outstanding WTOR. Furthermore, issuing a sequence of actions leads to a processing bottleneck. This is because each time it issues a reply, SA z/OS first has to wait for the next outstanding WTOR before it can continue with issuing the next reply. This method is therefore not recommended.

## Chapter 9. How to Set Up the Special Functions of IMS Automation

This information explains how to set up the special functions of IMS Automation for your specific needs.

For the setting up of base functions, like starting and stopping subsystems, see *IBM System Automation for z/OS Defining Automation Policy*.

### Automating Recovery for Application Components

IMS Automation provides automated recovery for transactions and programs and also enables monitoring of:

- Online log data sets (OLDS)
- Recovery control data sets (RECON)
- VTAM® Access Method Control Block (ACB)
- IMS connections to the external subsystems DB2 and IBM MQ

### How to Define Transaction Recovery

Customization of transaction recovery consists of:

- Determining which application program (TP) transactions will have recovery automation
- Identifying the batch message region (BMP) transactions that will have recovery automation
- Specifying the error threshold level that a recovery should stop at
- Identifying specific abend codes that you want recovery procedures to occur for
- Specifying the recovery procedure, which usually consists of invoking a command, a routine, and/or sending notifications to an operator

The recovery itself is typically triggered from the NetView automation table by calling the EVIECTOX routine when certain messages arrive at NetView. EVIECTOX then consults the automation policy to find out whether recovery is to be attempted and to determine what has to be done.

The following sections illustrate the configuration process.

### Specifying the Transactions or Programs to Be Recovered

In this example, suppose that recovery is enabled for the IMS10AA application on the application level, and that you also want it enabled for transactions PAYR, DBTS, and BLNG, but not for any other transaction.

You must then define recovery automation flags using the MINOR RESOURCES policy item for IMS10AA in the customization dialog for the following four minor resources:

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

To do this, enter **f** in the **Cmd** field and the minor resource name in the **Minor Resource** field on the Minor Resource Definitions panel and press Enter. This displays the Automation Flag Processing panel for the minor resource. Here you specify which flags are set and which are not. Set the recovery automation flag to NO for TRAN and YES for the three second-level minor resources. For more information, see "MINOR RESOURCES Policy Item" in *IBM System Automation for z/OS Defining Automation Policy*.

You can fine-tune recovery automation for programs in the same way by replacing the TRAN keyword with PROG.

### 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 RESOURCES 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	Critical		Frequent		Infrequent	
	Number	Interval (hh:mm:ss)	Number	Interval (hh:mm:ss)	Number	Interval (hh:mm:ss)
IMS10AA.TRAN.PAYR	2	01:00:00	2	05:00:00	2	24:00:00

For more details, see "MINOR RESOURCES Policy Item" in *IBM System Automation for z/OS Defining Automation Policy*.

For recovery thresholds of programs, you must use PROG instead of TRAN as the first part of the thresholds' name.

### Selecting the Abend Codes

The abend codes that recovery is to take place for are specified in the MESSAGES/USER DATA policy item for IMS10AA through the following keywords:

- The ABCODETRAN keyword for transactions
- The ABCODEPROG keyword for programs

If you want to initiate recovery for transaction PAYR only when the abend code is U3033 or U907, you need to create an ABCODETRAN.PAYR entry in the MESSAGES/USER DATA policy item and use action Cod on it. This takes you to the Code Processing panel where you enter the following definitions:

Code 1	Code 2	Code 3	Value Returned
*	U3033	*	INCLUDE
*	U907	*	INCLUDE
*	*	*	EXCLUDE

For more details, see [“ABCODETRAN: Transaction Abend Recovery”](#) on page 56.

In a similar way, you can use the ABCODEPROG keyword to include certain abend codes for certain programs in recovery automation.

### Specifying Recovery Actions

You specify the commands to be issued for recovery on the Command Processing panel for the DFS554A message ID of IMS10AA.

For example:

Ps/Select	AutoFn/*	Command Text
PROG		MVS &SUBSSUBIDSTA PGM &EHKVAR2

Ps/Select	AutoFn/*	Command Text
TRAN		MVS &SUBSSUBIDSTA TRAN &EHKVAR1

For more details, see “DFS554A: Respond to Program Abend” on page 57.

## Monitoring of Online Log Data Sets (OLDS)

IMS automation allows the monitoring of the online log data sets of IMS control regions and execution of recovery actions if needed. The IMS display command is issued to analyze the status of the OLDS data sets.

If needed, OLDS data sets are started or stopped. Any OLDS data sets that have been defined as spare OLDS data sets are started last and stopped first.

The IMS display command can be triggered by incoming messages that report OLDS-related activities, or it can be issued at regular time intervals.

The implementation of the OLDS monitoring function includes the following:

- A monitor resource to monitor the number of available OLDS
- A monitor resource to monitor for excessive switching of the OLDS
- Input parameters for the OLDS monitor function, defined as user data for the message ID OLDS
- Status messages for passive OLDS monitoring
- Command definitions for the health status update that are dependent on the OLDS switch frequency
- Threshold definitions
- Automation flag settings
- Relationships

The OLDS monitoring function is optional, that is, some definitions are necessary to enable it. For activation the following implementation steps are needed:

### IMS OLDS Monitor Resource

The IMS OLDS monitor resource has to be defined with the MTR policy object with the following characteristics:

Field Name	Field Value
Monitored Object	OLDS
Monitored Jobname	<i>jobname</i> of the IMS control region
Monitor command	INGRMIOL
Monitoring Interval	<i>hh:mm</i> (optional)

Without a specified monitoring interval, the monitor command is issued initially when the monitor resource is started.

Depending on the monitor results, INGRMIOL ends with the following return codes, which are mapped to the related health status for the monitor resource.

Return Code	Health status	Description
1	BROKEN	Monitor encountered a severe error
2	FAILED	DISPLAY OLDS failed
3	NORMAL	No problem found by OLDS monitoring

Return Code	Health status	Description
4	WARNING	One of the following occurred: <ul style="list-style-type: none"> <li>• Needed to start spare OLDS to have the minimum in AVAILABLE status</li> <li>• AUTOMATIC ARCHIVE is off</li> </ul>
5	MINOR	Could not start enough spare OLDS to have the minimum in AVAILABLE status
6	CRITICAL	Number of OLDS in BACKOUT status exceeds maximum limit

### Message ID OLDS

The input parameters for OLDS monitoring have to be defined as user data for the special message ID OLDS with the MESSAGES/USER DATA policy item of the IMS control region that the monitor resource is associated with.

Keyword	Data	Description
MINIMUM	<i>nn</i>	The minimum number of OLDS that must be available at all times. The default minimum number is 50% of the normal number of OLDS, where the normal number of OLDS is given in the response of the DISPLAY OLDS command.
SPARES	<i>(nn,nn...)</i>	The spares are OLDS that IMS automation activates when the number of available OLDS drops below the minimum and it is not sufficient to start the stopped OLDS to have the minimum number of OLDS available. The names of the spares are the two-digit numbers taken from the end of the ddname. For example, DFSOLP99 is the spare named 99. Be sure that the names of the spares match the names of existing OLDS.
BACKOUT	<i>nn</i>	The maximum number of OLDS that can have an OTHER-STS of BACKOUT. Set this number to the total number of acceptable OLDS data sets with an OTHER-STS of BACKOUT.

### Status Messages

Apart from optional active monitoring with the INGRMIOL monitor command after each monitoring interval, the OLDS monitoring function can be implemented with passive monitoring.

The following IMS messages can be used to trigger health status updates and recovery actions:

```
DFS3256I OPEN/ALLOCATION FAILED ON ddname
DFS3257I ONLINE LOG NOW SWITCHED - FROM DFSOLPxxx TO ddname2
DFS3258A LAST ONLINE LOG DATA SET IS BEING USED - NEED ARCHIVE
DFS3260I ONLINE LOG DATA SET SHORTAGE - NEED ANOTHER DATA SET
```

Messages DFS3256I, DFS3257I and DFS3260I only indicate OLDS-related activities that may or may not change the health status of the monitor resource. Therefore the defined monitor command first has to be issued to analyze the actual situation and to evaluate the health status before it can be updated. To achieve this, these messages have to be defined as status messages in the MESSAGES/USER DATA policy item of the OLDS monitor resource with the status Check.

By contrast, message DFS3258A informs about a concrete problem situation. It can be directly mapped to a degraded health status by selecting CRITICAL as the new health status with the AUT action in the MESSAGES/USER DATA policy item of the OLDS monitor resource.



## Automation Flag Settings

Before starting or stopping OLDS data sets, the OLDS monitoring function first checks the Recovery automation flag of the OLDS minor resource for the IMS control region.

The recovery actions are only issued if they are allowed by the automation flag.

## IMS OLDS Switch Frequency Monitor

The IMS OLDS switch frequency monitor resource has to be defined in the MTR policy object with the following characteristics:

Field Name	Field Value
Monitored Object	OLDS_SWITCH
Monitored Jobname	<i>jobname</i> of the IMS control region
Monitor command	(none)
Monitoring Interval	<i>hh:mm</i> (optional)

The switch frequency is determined by passive monitoring of message DFS3257I.

## Command Definitions for the Health Status Update Related to the Switch Frequency

To set the health status of the monitor resource, depending on the frequency of the received message DFS3257I, define the following commands for message DFS3257I in the MESSAGES/USER DATA policy item of an IMS control region:

Ps/Select	AutoFn/*	Command Text <sup>1</sup>
INFR		INGMON OLDS_SWITCH, JOBNAME=&SUBSJOB, STATUS=WARNING, INFO=(MSG, INFR EQUENT THRESHOLDS LIMIT REACHED FOR OLDS SWITCHING)
FREQ		INGMON OLDS_SWITCH, JOBNAME=&SUBSJOB, STATUS=MINOR, INFO=(MSG, FREQUE NT OLDS SWITCHING DETECTED)
CRIT		INGMON OLDS_SWITCH, JOBNAME=&SUBSJOB, STATUS=CRITICAL, INFO=(MSG, CRI TICAL OLDS SWITCHING FREQUENCY REACHED)
ALWAYS		INGMON OLDS_SWITCH, JOBNAME=&SUBSJOB, STATUS=NORMAL, INFO=(MSG, OLDS SWITCHING FREQUENCY IS NORMAL)

1. The **Command Text** field is spread across two screens.

## Threshold Definitions

Define the threshold levels for the degradation of the health status for the minor resource DFS3257I in the MINOR RESOURCES policy item of the IMS control region.

## Relationships

To propagate the health status of the monitor resources to the health status of the application, HASMONITOR relationships have to be defined between the IMS control region and both monitor resources.

The monitor resources are kept active only during the UP time of the IMS application group. This is achieved with an additional HASPASSIVEPARENT and a FORCEDOWN/WhenObservedDown relationship between each monitor resource and the IMS application group. This ensures that the monitor resources can rely on all required functions.

## Monitoring of Recovery Control Data Sets (RECON)

IMS Automation allows the monitoring of recovery control data sets of IMS control regions.

The IMS command `RMLIST DBRC='RECON STATUS'` is issued regularly to analyze the status of the listed RECON data sets in the response to this command.

The RECON monitoring function is optional, that is, some definitions are necessary to enable it.

The implementation of the RECON monitoring function includes the following:

- A monitor resource to monitor the number of available RECON data sets
- A status message for passive RECON monitoring

## IMS RECON Monitor Resource

The IMS RECON monitor resource has to be defined in the MTR policy object with the following characteristics:

Field Name	Field Value
Monitored Object	RECON
Monitored Jobname	<i>jobname</i> of the IMS control region
Monitor command	INGRMIRE
Monitoring Interval	<i>hh:mm</i>

The monitor command is executed after each monitoring interval. It issues the following command and analyzes the status of the listed RECON data sets in the response to this command:

```
RMLIST DBRC='RECON STATUS'
```

Depending on the monitor results, INGRMIRE ends with the following return codes, which are mapped to the related health status for the monitor resource:

Return Code	Health status	Description
1	BROKEN	Monitor encountered a severe error
2	FAILED	RMLIST DBRC='RECON STATUS' failed
3	NORMAL	No problem found by RECON monitoring
4	WARNING	No COPY2 found for RECON
5	MINOR	No SPARE found for RECON

Return Code	Health status	Description
6	CRITICAL	Neither COPY2 nor SPARE, and possibly no COPY1 found for RECON

## Status Message

Apart from active monitoring with the INGRMIRE monitor command after each monitoring interval, RECON monitoring function can be implemented with passive monitoring.

The following IMS message can be used to trigger health status updates:

```
DSP0038I RECON INCONSISTENCY RECON HEADER RECORD NOT FOUND
```

When message DSP0038I is received, the defined monitor command first has to be issued to analyze the actual situation and to evaluate the health status before it can be updated. To achieve this, this message has to be defined as a status message in the MESSAGES/USER DATA policy item of the monitor resource with the status Check.

## Relationships

To propagate the health status of the monitor resource to the health status of the application, a HASMONITOR relationship has to be defined between the IMS control region and the monitor resource.

The monitor resource is kept active only during the UP time of the IMS application group with an additional HASPASSIVEPARENT and a FORCEDOWN/WhenObservedDown relationship between the monitor resource and the IMS application group. This ensures that the monitor resource can rely on all required functions.

## Monitoring of VTAM ACB

IMS automation allows status checking of the VTAM Access Method Control Block (ACB) and the enablement of logons.

The IMS command DISPLAY ACTIVE DC is issued regularly after each monitor interval and the response to this command is analyzed.

This monitor function is only useful for an IMS control region, not for a DB control region.

The DC monitoring function is optional, that is, some definitions are necessary to get it up and running.

The implementation of the DC monitoring function includes the following:

- A monitor resource to monitor the status of the VTAM ACB and the enablement of logons
- A status message for passive DC monitoring

## IMS DC Monitor Resource

The IMS DC monitor resource has to be defined in the MTR policy object with the following characteristics:

Field Name	Field Value
Monitored Object	DC
Monitored Jobname	<i>jobname</i> of the IMS control region
Monitor command	INGRMIDC
Monitoring Interval	<i>hh:mm</i>

The monitor command is executed after each monitoring interval. It issues the following IMS command and analyzes the status of the VTAM ACB and the LOGONS enablement:

```
DISPLAY ACTIVE DC
```

If no monitoring interval is specified, the defined monitor command is only issued once initially after the monitor resource has been started.

Depending on the monitor results, INGRMIDC ends with the following return codes, which are mapped to the relating health status for the monitor resource:

Return Code	Health status	Description
1	BROKEN	Monitor encountered a severe error
2	FAILED	DISPLAY ACTIVE DC failed
3	NORMAL	VTAM ACB is OPEN and LOGONS enabled
4	WARNING	LOGONS are not enabled

### Status Message

Apart from active monitoring with the INGRMIDC monitor command after each monitoring interval, passive monitoring is used to update the health status of the monitor resource.

The following IMS message is used to trigger health status update:

```
DFS2111I VTAM ACB CLOSED.
```

When message DFS2111I is received, the health status should be set to WARNING. To achieve this, this message has to be defined as a status message in the MESSAGES/USER DATA policy item of the monitor resource with the status WARNING.

### Relationships

To propagate the health status of the monitor resource to the health status of the application, you have to define a HASMONITOR relationship between the IMS control region and the monitor resource.

The monitor resource is kept active only during the UP time of the IMS application with an additional HASPASSIVEPARENT and a FORCEDOWN/WhenObservedDown relationship between the monitor resource and the IMS control region. This ensures that the monitor resource can rely on all required functions.

## Monitoring of IMS - External Subsystem Connections

IMS automation allows the monitoring of IMS connections to the external subsystems DB2 and IBM MQ. The IMS command DISPLAY SUBSYS is issued regularly after each monitor interval and the response to this command is analyzed concerning the status of the connection to an external subsystem.



**Attention:** Connections between IMS and MQ can only be monitored in this way using the IBM MQ - IMS™ adapter. Connections via the IBM MQ - IMS™ bridge cannot be monitored with IMS automation. For more information about setting up the adapter, see the [Setting up the IMS adapter](#) topic in IBM MQ documentation.

The IMS - external subsystem connection monitoring function is optional, that is, definitions are necessary to enable it.

The implementation of the IMS - external subsystem connection monitoring function includes the following:

- A monitor resource for each connection to be monitored between an IMS control region and a DB2 or IBM MQ external subsystem.
- Status messages
- Relationships

## IMS - External Subsystem Connection Monitor Resource

The IMS - external subsystem connection monitor resource has to be defined in the MTR policy object with the following characteristics:

Field Name	Field Value
Monitored Object	MVS subsystem ID, <i>extsysid</i> , of the external subsystem
Monitored Jobname	<i>jobname</i> of the IMS control region
Monitor command	INGRMIDB
Monitoring Interval	<i>hh:mm</i>

The specified monitor command INGRMIDB is executed after each monitoring interval. It issues the DISPLAY SUBSYS command for the specified external subsystem and analyzes the status of the connection between both subsystems in the response to this command.

Depending on the monitor results, INGRMIDB ends with the following return codes, which are mapped to the related health status for the monitor resource:

Return Code	Health status	Description
1	BROKEN	No external subsystem connection is defined in IMS
2	FAILED	DISPLAY SUBSYS <i>extsysid</i> command failed
3	NORMAL	IMS is connected to the external subsystem
4	WARNING	The IMS connection to the external subsystem is pending
5	MINOR	IMS is not connected to the external subsystem
6	CRITICAL	No external subsystem connection is defined in IMS, but recovery is outstanding

## Status Messages

Apart from active monitoring by issuing the INGRMIDB monitor command after each monitoring interval, the IMS - external subsystem monitoring function of SA z/OS also performs passive monitoring by updating the health status whenever it receives one of the following status messages:

- DB2 messages:

```
DFS0801I SUBSYSTEM CONNECTION FOR xxxxxxxx COMPLETED
DFS3611 EXTERNAL SUBSYSTEM xxxx CONNECTION TERMINATED RC=y
DSNM003I IMS/TM xxxx FAILED TO CONNECT TO SUBSYSTEM yyyy RC=rc
```

- IBM MQ messages:

```
DFS0801I SUBSYSTEM CONNECTION FOR xxxxxxxx COMPLETED
DFS3611 EXTERNAL SUBSYSTEM xxxx CONNECTION TERMINATED RC=y
CSQQ002E IMS/TM yyyy failed to connect to queue manager qqqq, MQRC=mqrc
```

These messages indicate whether the attempt to connect the IMS control region to the DB2 or MQ subsystem was successful. In response to these messages, the health status is set to NORMAL if it was successful and MINOR if it was not.

Because these messages are issued to the IMS/TM master terminal operator, you have to define them to the IMS message exit in order that they be issued as WTOs to the console. For details, see [“Using the IMS Automation Message Exit”](#) on page 42. SA z/OS provides statements in the NetView automation table that is built that ensure an update of the monitor's health status.

### Relationships

To propagate the health status for the monitor resource to the health status of the application, you must define a HASMONITOR relationship between the IMS control region and the monitor resource.

The monitor resource is kept active only during the UP time of the IMS application group and the external subsystem application group. This is achieved with an additional HASPASSIVEPARENT and a FORCEDOWN/WhenObservedDown relationship between the monitor resource and both the IMS application group and the external subsystem application group. This ensures that the monitor resource can rely on all required functions.

## Chapter 10. MESSAGES/USER DATA Entries for IMS Automation

Because IMS automation is integrated into SA z/OS, you must enter any information for IMS automation in the policy database via the customization dialog.

In most cases, the customization dialog itself restricts you to the format that this information must be entered in. There are, however, a number of IMS-specific automation parameters that must or can only be specified as entries in the MESSAGES/USER DATA policy item of the appropriate application policy object. In these cases, the customization panels provide no information about the keywords and the format of their parameters.

This chapter contains detailed descriptions of the automation entries that you have to define for IMS-specific keywords. However, a general understanding of the MESSAGES/USER DATA policy item is assumed. For more information, see "MESSAGES/USER DATA policy item" in *IBM System Automation for z/OS Defining Automation Policy*.

### IMS-Specific MESSAGES/USER DATA Keywords

Most of the following keywords must be specified in the MESSAGES/USER DATA item of the respective control region and apply to DB control regions as well as to IMS control regions.

They need to be entered in the **Message ID** field on the Message Processing panel.

### ABCODEPROG: Respond to IMS Application Abends

Use ABCODEPROG keyword to define actions to be taken for program abends of program-driven batch message processing (BMP) regions, IMS Fast Path (IFP) regions, Java message processing program (JMP) regions, Java batch processing program (JBP) regions, and Message processing programs (MPPs).

Transaction driven BMPs use the ABCODETRAN keyword to determine recovery actions.

Use code processing on the Message Processing panel:

Code 1	Code 2	Code 3	Value Returned
*	<i>acode</i>	<i>progid</i>	INCLUDE or EXCLUDE

**Note:** The ABCODEPROG keyword may be followed with a dot and a program ID *progid*.

### Keyword and Parameter Definitions

#### ABCODEPROG [ .*progid*]

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

#### CODE

Use this keyword to define which abends should be included or excluded from recovery.

**\***

An asterisk as the first positional parameter is required for compatibility with the ABCODETRAN entry. Code an asterisk as shown.

#### *acode*

The abend code. An asterisk (\*) can be used for generic specifications. System abend codes should be prefixed with an S, such as SOC1.

#### *progid*

The program name.

**INCLUDE|EXCLUDE**

Indicates whether to initiate a recovery sequence for this program and abend code combination. Use INCLUDE to initiate a recovery and EXCLUDE if you do not want a recovery initiated. INCLUDE is the default.

**Comments and Usage Notes**

If the ABCODEPROG entry is omitted, no recovery takes place and a warning message is issued. The program name can either be specified as ABCODEPROG . *progid* or as the first value of the CODE attribute. Use ABCODEPROG . *progid* when you want all of the specifications to apply to one specific program. Use the CODE attribute when you want to code several transactions.

**Example of Usage**

In the following example, recovery is not attempted for program SAMPLE4 when the abend code is U0778. In all other cases, recovery takes place.

Code 1	Code 2	Code 3	Value Returned
*	U0778	SAMPLE4	EXCLUDE
*	*	*	INCLUDE

**ABCODETRAN: Transaction Abend Recovery**

Use this keyword to include transaction abend codes in recovery or exclude them from recovery. This includes both transaction-driven message processing regions (MP) and transaction-driven batch message processing (BMP) regions.

Use code processing on the Message Processing panel:

Code 1	Code 2	Code 3	Value Returned
<i>tran</i>	<i>abend</i>	<i>pgm</i>	INCLUDE or EXCLUDE
*	U3033	*	INCLUDE

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

**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 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 either specified as ABCODETRAN.*tran* or as 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.

## Example of Usage

In the following example, recovery takes place for transaction PAYR on IMS10AA if the abend code is U903 or U3033. No recovery takes place for any other transaction or abend code.

Code 1	Code 2	Code 3	Value Returned
*	U3033	*	INCLUDE
*	U907	*	INCLUDE
*	*	*	EXCLUDE

## DFS554A: Respond to Program Abend

This keyword is used to define commands or replies to be issued in response to the application program abend message DFS554A.

If the DFS554A message indicates that a transaction has stopped, the commands or replies that have been defined for the selection name TRAN are issued to restart the transaction.

Similarly, if the DFS554A message indicates that a program has stopped, the commands or replies that have been defined for the selection name PROG are issued to restart the program.

## Variables

Apart from the task global variables that are provided by AOCQRY, you can use the following variables when specifying the command or reply. The variables are replaced by their corresponding value during runtime.

Variable	Value
&EHKVAR1	Transaction name
&EHKVAR2	Program name
&EHKVAR3	Job identifier
&EHKVAR4	Region identifier
&EHKVAR5	User abend code
&EHKVAR6	System abend code

## Example of Usage

Ps/Select	AutoFn/*	Command Text
PROG		MVS &SUBSSUBIDSTA PGM &EHKVAR2
TRAN		MVS &SUBSSUBIDSTA TRAN &EHKVAR1

## IMSINFO: Display Information

These commands are issued when the INGIMS REQ=INFO or INGIMS REQ=XINFO command is used to display the state of the selected IMS Control or DBCTL region. The commands are issued via the IMS

subsystem ID on an MVS EMCS console and the resulting messages are either displayed on the INGIMS panel or written to the user's NetView console.

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

Use user processing on the Message Processing panel:

Keyword	Data
IMSCMD	('description','IMS_command')

## Keyword and Parameter Definitions

### description

This is text that is placed before the output of *IMS\_command*. It can be used to identify the command output in the output stream. The description can be any string, but must be enclosed in quotation marks.

### IMS\_command

This is the command that is to be executed. It is appended to the IMS subsystem ID and issued as an MVS command. The output is collected and displayed. The command can be any valid IMS Type 1 or Type 2 command. If you use a Type 2 command, be sure to also use INGIMS REQ=XINFO. The commands must be enclosed in quotes, but if the command contains single quotes then use double quotes as the delimiter.

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

## Comments and Usage Notes

This policy is required for the correct operation of the INFO request of the INGIMS command, and also for the PF10 function key of the DISPINFO panel.

## Example of Usage

Keyword	Data
IMSCMD	('CQS DATA', 'DIS CQS')
IMSCMD	('Comms', 'DIS ACT DC')

## TCO: Allocate data set with Commands for Time-Driven Procedures

This entry allows you to specify the name of the data set holding the TCO members and commands to be issued to initiate, change, start, or stop time-driven procedures for any IMS operation.

Use reply processing on the Message Processing panel:

Pass/Selection	Retry Count	Reply Text
INIT		DFSTCF LOAD DFSTCF .
SPEC		DFSTCF LOAD &EHKVAR1 .
START		/START LTERM DFSTCF1 .
STOP		/PSTOP LTERM DFSTCFI .

## Keyword and Parameter Definitions

### INIT

This entry specifies the IMS command required to start the initial time driven procedure DFSTCF.

**SPEC**

This entry specifies the IMS command required to change from the current TCO script to a script name entered from the operator interface.

**START**

This entry specifies the IMS command required to start TCO.

**STOP**

This entry specifies the IMS command required to stop TCO.

**Comments and Usage Notes**

1. Use the user processing option on the Message processing panel to define TCO data set name. The specification is optional.

Keyword	Data
DSN	<i>data set name</i>

2. For more information on TCO, refer to the *IMS Operations Guide*.
3. &EHKVAR1 is the name entered from the IMS automation operator interface TCO function.

**TCOMEMBERS: Define TCO Members**

This keyword serves to create a list of members that appear in a pop-up IMS Automation TCO Member Load panel.

The panel is shown in [Figure 1 on page 59](#).

```

EVIKMT10      IMS Automation: TCO Member Load
Resource/Domain => IMS711C4/APL/KEY1
Page: 1 of 1
Date: 05/23/02
Time: 13:01
Domain: IPSNM

TCO Status . . . . . : Available

Member      Select one of the user supplied TCO members
Alterna    - STRTLNES      START LINES
Maximum    - ASGNTRAN      ASSIGN TRANSACTIONS
           - DISPROG      DISPLAY ACTIVE PROGRAMS

F1=Help    F3=Cancel

Command ===>
F1=Help    F2=End      F3=Return   F4=IMS Menu   F6=Roll

```

Figure 1. Example IMS Automation TCO Member Load Pop-up Panel

Use user processing on the Message Processing panel:

Keyword	Data
NAME	<i>(membername, 'comment')</i>

**Format**

► TCOMEMBERS — NAME=( *membername* ,'*comment* ') ◄

## Keyword and Parameter Definitions

***membername***

This is the 8-character name of the member previously defined in the IMS TCO member library. The library is associated with the DFSTCF DD statement in the IMS start up JCL.

***comment***

This is a comment, up to 32 characters long.

## Comments and Usage Notes

For more information on TCO, refer to [“Working with TCO Functions”](#) on page 70.

## Example of Usage

Keyword	Data
NAME	(STRTLNES, 'START LINES')
NAME	(ASGNTRAN, 'ASSIGN TRANSACTIONS')

---

## Chapter 11. INGIMS Operator Command

INGIMS is IMS-related SA z/OS command that issues IMS console commands and that can be invoked by operators or via PIPES.

---

### INGIMS: Issue List of Defined Transactions and View the Output

---

The INGIMS command lets you:

- Issue any console-enabled IMS type 1 command
- Issue any IMS type 2 command if an IMSplex name is provided
- Broadcast messages to all or selected IMS users
- Issue a list of defined transactions and view the output
- Display the output of IMS transactions in full-screen or pipeable line mode

For a detailed description, see INGIMS command in *IBM System Automation for z/OS Operator's Commands*.



## Chapter 12. Monitor Commands

The following routines should be used as a MONITOR command when defining monitor resources for IMS Automation.

### INGRMIDB Routine for the Monitoring of IMS - External Subsystem Connections

This routine should be used as the monitor command for a monitor resource. It is issued after each monitoring interval to check whether IMS is connected to a DB2 or IBM MQ external subsystem. To achieve this, the IMS command DISPLAY SUBSYS is issued for the external subsystem to query the status of the IMS - external subsystem connection.

The health status is considered to be NORMAL when the status of the monitored IMS - external subsystem connection is Connected. A degraded health status is returned in all other cases.

#### Format

➤ INGRMIDB ➤

#### Restrictions

The INGRMIDB routine is assumed to be defined as the monitor command for a monitor resource with the following restrictions:

- The MVS subsystem ID of the external subsystem, *extsysid*, is specified as the monitored object.
- A job name of a subsystem of category IMS with subcategory CTL is specified for the monitor resource.
- The monitor resource is related to a IMS subsystem.

#### Return Codes

Depending on the monitor results, INGRMIDB ends with the following return codes, which are mapped to the related health status for the monitor resource:

Return Code	Health status	Description
1	BROKEN	No external subsystem connection is defined in IMS
2	FAILED	DISPLAY SUBSYS <i>extsysid</i> command failed
3	NORMAL	IMS is connected to the external subsystem
4	WARNING	The IMS connection to the external subsystem is pending
5	MINOR	IMS is not connected to the external subsystem
6	CRITICAL	No external subsystem connection is defined in IMS, but recovery is outstanding

### INGRMIOL Routine for OLDS Monitoring

This routine should be used as the monitor command for a monitor resource.

It is issued after each monitoring interval to check the status of the online log data sets (OLDS) of an IMS control region and to take recovery actions if needed. To achieve this, the IMS command DISPLAY

OLDS is issued to analyze the status of the listed OLDS data sets. If needed, OLDS data sets are started or stopped. Any OLDS data sets that have been defined as spare OLDS data sets are started last and stopped first.

The health status of the monitored OLDS data sets is considered to be NORMAL when the number of available OLDS data sets is not less than a given minimum number, and the AUTOMATIC ARCHIVE is switched on. In all other cases a degraded health status is returned.

**Format**

➡ INGRMIOL ⬅

**Restrictions**

- The monitor resource must be for an IMS control region that is defined as an IMS subsystem of subcategory CTL.
- OLDS is specified as the monitored object.
- Automation actions in INGRMIOL are only taken if the recovery automation flag for minor resource OLDS is on.

**Comments and Usage Notes**

The automation settings for OLDS monitoring are taken from user data for the special message ID OLDS, which is defined in the MESSAGES/USER DATA policy item of the IMS control region with the following keyword-data pairs:

<b>Keyword</b>	<b>Data</b>	<b>Description</b>
MINIMUM	<i>nn</i>	The minimum number of OLDS data sets that must be in (AVAILABLE + IN USE) status.  Half of the number of listed OLDS data sets in the response of the DISPLAY OLDS command is assumed as the default value.
SPARES	<i>(nn,nn...)</i>	These specified spare OLDS data sets are started last or stopped first when OLDS data sets have to be started or stopped in order to have the minimum number of OLDS data sets in status AVAILABLE.
BACKOUT	<i>nn</i>	The maximum number of OLDS that are allowed to be in a BACKOUT status.

All of these settings are optional.

**Return Codes**

Depending on the monitor results, INGRMIOL ends with the following return codes, which are mapped to the related health status for the monitor resource:

<b>Return Code</b>	<b>Health status</b>	<b>Description</b>
1	BROKEN	The monitor encountered a severe error.
2	FAILED	DISPLAY OLDS failed.
3	NORMAL	No problem found by OLDS monitoring.



Return Code	Health status	Description
4	WARNING	One of the following occurred: <ul style="list-style-type: none"> <li>• Needed to start spare OLDS to have the minimum in AVAILABLE status</li> <li>• AUTOMATIC ARCHIVE is off</li> </ul>
5	MINOR	Could not start enough spare OLDS to have the minimum in AVAILABLE status.
6	CRITICAL	The number of OLDS in BACKOUT status exceeds maximum limit.

## INGRMIRE Routine for RECON Monitoring

This routine should be used as the monitor command for a monitor resource. It is issued after each monitoring interval to check the status of the recovery control data sets (RECON) of an IMS control region. To achieve this, the IMS command RMLIST DBRC='RECON STATUS' is issued to analyze the status of the listed RECON data sets in the response to this command.

The health status of the monitored RECON data sets is considered to be NORMAL when three RECON data sets are found in the status COPY1, COPY2 and SPARE. A degraded health status is returned in all other cases.

### Format

➤ INGRMIRE ➤

### Restrictions

- The monitor resource must be for an IMS control region that is defined as an IMS subsystem of subcategory CTL.
- RECON is specified as the monitored object.

### Return Codes

Depending on the monitor results, INGRMIRE ends with the following return codes, which are mapped to the related health status for the monitor resource:

Return Code	Health status	Description
1	BROKEN	The monitor encountered a severe error.
2	FAILED	RMLIST DBRC='RECON STATUS' failed.
3	NORMAL	No problem found by RECON monitoring.
4	WARNING	No COPY2 found for RECON.
5	MINOR	No SPARE found for RECON.
6	CRITICAL	Neither COPY2 nor SPARE found, and possibly no COPY1 found for RECON.

## INGRMIDC Routine for DC Monitoring

This routine should be used as the monitor command for a monitor resource.

It is issued after each monitoring interval to check whether the VTAM ACB of an IMS control region is OPEN and LOGONS are ENABLED. To achieve this, the response to the IMS command DISPLAY ACTIVE DC is analyzed.

The health status is considered to be NORMAL when the VTAM ACB of the IMS control region is OPEN and LOGONS are ENABLED. A degraded health status is returned in all other cases.

### Format

➤ INGRMIDC ➤

### Restrictions

- The monitor resource must be for an IMS control region that is defined as an IMS subsystem of subcategory CTL. It should not be used for a DB control region.
- DC is specified as the monitored object.

### Return Codes

Depending on the monitor results, INGRMIDC ends with the following return codes, which are mapped to the related health status for the monitor resource.

Return Code	Health status	Description
1	BROKEN	The monitor encountered a severe error.
2	FAILED	DISPLAY ACTIVE DC failed.
3	NORMAL	VTAM ACB is OPEN and LOGONS enabled.
4	WARNING	LOGONS are not enabled.

# Chapter 13. Using IMS Automation

This information explains how to use the IMS Automation panels and to work with subsystems.

It is assumed that you have used and are familiar with the SA z/OS operator interface. This chapter describes characteristics that are unique to IMS Automation.

## Using the Main Menu

The main menu panel lists all of the tasks available with the operator interface, as shown in [Figure 2](#) on [page 67](#).

```
EVIK0000          SA z/OS - Command Dialogs
Domain ID   = IPSFM   ----- IMS   -----   Date = 03/18/09
Operator ID = OPER1   System = KEY3   Time = 14:38:54

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

Action =>  ___  1. Inquire      Display an IMS control reg.
              2. Start        Start an IMS subsystem
              3. Shutdown     Shutdown an IMS subsystem
              4. Triggers     Display trigger conditions
              5. Service Periods Perform scheduling functions
              6. Master Terminal Perform Master Terminal Commands

              8. Broadcast     Send message to users
              9. TCO Management Load/Start/Stop TCO
             10. Dependent Regions Manage Dependent Regions
             99. Local functions Provide access to user defined local

functions

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

Figure 2. IMS Automation Main Menu

You can specify a resource using the following fields:

### Resource

To define one or more IMS Subsystems to be processed, type in the name of a resource. This should be in the format name/type/system. This name will be used for all functions until a new one is entered on one of the Major Option panels or selected on one of the List Selection panels.

To display a list of all known IMS resources type a question mark (?) for the resource. You can then select a subsystem from this list (refer to [“Selecting and Viewing Subsystems”](#) on page 68). To obtain this list for a sysplex that is not the current one, specify the corresponding sysplex in the **Target** field before you press Enter.

**Note:** Many of the menu options are not valid for remote sysplexes.

### Target

This field allows you to specify the system name, the domain ID or the sysplex name where you would like the required IMS function to be performed. To limit the resource list to a specific system, use the \*/APL/system notation.

If you specify a sysplex name and then enter a question mark in the **Resource** field you call a list of all the IMS control regions in the specified sysplex. For the current sysplex, you need not enter the sysplex name.

You can then select one of the following options by entering it in the **Action** field. The options that are marked with an asterisk (\*) are not valid for remote sysplexes. For more details about the SA z/OS command that is called for each option, see *IBM System Automation for z/OS Operator's Commands*.

### 1 (Inquire)\*

Use this option to retrieve detailed information for an IMS resource. See [“Getting Detailed Status” on page 69](#).

### 2 (Start)

Use this option to initiate the startup process of a resource. This option calls the INGREQ command.

### 3 (Shutdown)

Use this option to initiate the shutdown process of a resource. This option calls the INGREQ command.

### 4 (Triggers)

Use this option to display the triggers associated with a resource. This option calls the DISPTRG command.

### 5 (Service Periods)

Use this option to display or override the schedule that is associated with a resource. This option calls the INGSCHED command.

### 6 (Master Terminal)\*

Use this option to issue a command to a specific subsystem. This option calls the INGIMS REQ=CMD command.

### 8 (Broadcast)\*

Use this option to send a message to a specific subsystem. This option calls the INGIMS REQ=BROADCAST command.

### 9 (TCO Management)\*

Use this option to manage the Time Controlled Operations (TCO) functions of a specific subsystem. See [“Working with TCO Functions” on page 70](#) for details.

### 10 (Dependent Regions)\*

Use this option to display information about IMS dependent regions that you can issue commands against. See [“Displaying IMS Dependent Region Information” on page 71](#) for details.

### 99 (Local functions)\*

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

**Note:** When option 99 is selected from the IMS Automation main menu, the EVIEU000 module is called. This module is not part of SA z/OS, but should be provided by you as a user-defined application.

**Important:** Option 99 is only valid for the local sysplex. You cannot access a remote sysplex with any of these functions.

## Selecting and Viewing Subsystems

---

This section describes how to select a resource from a list of available resources and how to display detailed information about a subsystem.

### Selecting a Subsystem

You can specify the resource you want to work with by simply entering its name in the **Subsystem** or **Resource** field of the respective panel.

On the IMS-specific panels you can call up a list of the available IMS control regions by entering a question mark at position one of the **Subsystem** or **Resource** field. On the panels that belong to basic SA z/OS commands (INGREQ, INGSCHED, DISPTRG), you can use an asterisk (\*) as a wildcard.

[Figure 3 on page 69](#) shows a list of IMS control regions generated with the question mark function.

```

INGKYSTS                      SA z/OS - Command Dialogs          Line 1    of 34
Domain ID   = IPSFM           ----- Selection Panel -----      Date = 08/22/05
Operator ID = OPER            Sysplex = KEY1PLEX                          Time = 06:27:50

CMD: S Select
CMD Name      Type System   Compound   Desired   Observed   / scroll   Nature
-----
s  IMS641C4    APL  KEY2     INHIBITED  AVAILABLE  UNKNOWN
-  IMS711C4    APL  KEY1     INAUTO    UNAVAILABLE STARTING
-  IMS711DL    APL  KEY1     INAUTO    AVAILABLE  STARTING
-  IMS711F1    APL  KEY1     SATISFACTORY UNAVAILABLE SOFTDOWN
-  IMS711M1    APL  KEY1     SATISFACTORY UNAVAILABLE SOFTDOWN
-  IMS711RC    APL  KEY1     INAUTO    AVAILABLE  STARTING
-  IMS712CX    APL  KEY1     SATISFACTORY AVAILABLE  AVAILABLE
-  IMS712DL    APL  KEY1     SATISFACTORY AVAILABLE  AVAILABLE
-  IMS712F1    APL  KEY1     PROBLEM   UNAVAILABLE HARDDOWN
-  IMS712M1    APL  KEY1     PROBLEM   UNAVAILABLE HARDDOWN
-  IMS712RC    APL  KEY1     SATISFACTORY AVAILABLE  AVAILABLE
-  IMS713C4    APL  KEY1     AWAITING  UNAVAILABLE AVAILABLE
-  IMS713DL    APL  KEY1     INAUTO    UNAVAILABLE STOPPING

Command ==>
PF1=Help    PF2=End      PF3=Return   PF6=Roll
PF9=Refresh PF10=Previous PF11=Next    PF12=Retrieve
    
```

Figure 3. Selection Panel for IMS Resources

The list contains all IMS subsystems of the KEY1PLEX sysplex (see line 3 of the panel) that are defined to SA z/OS. You can use it not only to select a subsystem (by entering s in the **CMD** column), but also to get an overview of the sysplex. Columns 5 through 7, for example, contain status information for the subsystems. For more details on the different status types, see "Statuses Supplied by the Automation Manager" in *IBM System Automation for z/OS User's Guide*.

You can scroll horizontally through the list to the right by pressing PF11, and to the left by pressing PF10. The first three columns, which make up the resource name of the subsystem, are fixed.

## Getting Detailed Status

To view the status of a specified subsystem in detail, select option **1** on the main menu panel and press Enter.

This displays the panel shown in [Figure 4 on page 70](#)

## Selecting and Viewing Subsystems

```
AOFKINFO          SA z/OS - Command Dialogs          Line 1    of 221
Domain ID   = IPSFM  ----- DISPINFO -----      Date = 10/26/07
Operator ID = OPER                                     Time = 15:42:45

Subsystem ==>  IMS1CTL      System ==>  KEYC      System name, domain ID
                                           or sysplex name
Subsystem   : IMS1CTL      on System : KEYC

Description  : IMS Control Region

Inform list : SDF

Class chain : C_IMS_DCCTL  Class for IMS DCCTL regions
              C_IMSCTL    Class for IMS Control regions

Job Name    : IMS9A1C4
ASID       : 005D

Job Type    : MVS

Category    : IMS - CTL

Current status : UP
Last Monitored : 12:17:16 on 10/25/07
Last Changed  : 13:12:01 on 10/25/07
Last Message  :
                AOF571I 13:12:01 : IMS1CTL SUBSYSTEM STATUS FOR JOB IMS9A1C4 IS UP
                - UP MESSAGE RECEIVED

Monitor      : INGPJMON
Monitor Status : ACTIVE
Monitor Interval : None specified

Primary WTOR reply IDs : 46

Last termination : 10:26:28 on 08/27/07   Type : NORM
Last start       : 13:12:01 on 10/25/07   Type : NORM

ARM Element Name : None

Command ==>
PF1=Help      PF2=End      PF3=Return    PF4=INGINFO    PF6=Roll
PF8=Forward   PF9=Refresh   PF10=IMS Info PF12=Retrieve
```

Figure 4. Detailed Subsystem Information Panel for an IMS Control Region

## Working with TCO Functions

The master terminal functions provide a fullscreen panel interface to perform Time Controlled Operations (TCO) functions.

Select option 9, TCO Management, to display the IMS TCO Status panel, as shown in [Figure 5 on page 71](#).

**Note:** Use the new IMS Automation command INGIMS to achieve the same functionality: for example, INGIMS resource REQ=TCO

```

EVIKYTC0          SA z/OS - Command Dialogs      Line 1      of 7
Domain ID = IPSFP  ----- IMS TCO Status ----- Date = 05/04/07
Operator ID = OPER                               Time = 18:48:24
Control Reg.= IMS2CTL/APL/KEY4
CMD: A Load      B Start      C Stop

                                           / scroll
CMD Name      Type      Status      Description
-----
- DFSTCF      LTERM      ACTIVE
- BITEST1     MEMBER     DISPLAY MESSAGE QUEUE *QBUF* PLU
- BITEST2     MEMBER     DISPLAY MESSAGE QUEUE *QBUF*
- BITEST3     MEMBER     DISPLAY MESSAGE QUEUE *QBUF* PLU
- DFSTCF      MEMBER     DEFAULT MEMBER
- DFSTC0      MEMBER     HUGOA ALTERNATE MEMBER
- SVDfstC     MEMBER     ALTERNATE MEMBER

Command ==>
PF1=Help      PF2=End      PF3=Return      PF6=Roll
               PF9=Refresh      PF12=Retrieve
    
```

Figure 5. IMS TCO Status Panel

The Time Controlled Operations (TCO) interface lets you issue TCO commands from the IMS Automation interface.

From this panel you can:

- Load a specific member
- Start or stop TCO processing for a logical terminal

## Load a Specific Member

Enter command code A against an object of type MEMBER to load that member. The member is loaded into TCO via the IMS control region outstanding reply.

The reply text that is used is defined in the ACF via the TCO entry and has a selection of SPEC. Typically it is defined as DFSTCF LOAD &EHKVAR1 . in the ACF, where the EHKVAR1 variable is set to the member name that the line command is executed against.

## Start or Stop TCO Processing

Enter command code B or C against an object of type LTERM to start or stop TCO processing for that member.

This invokes INGIMS with the command of /START LTERM *name* or /STOP LTERM *name*, where *name* is the name of the logical terminal.

## Displaying IMS Dependent Region Information

Option 10 displays details about the IMS dependent regions that are being processed by the specified IMS Control Region.

See [Figure 6 on page 72](#). All dependent regions that are currently active are displayed. Those that are defined to SA z/OS are displayed as SA z/OS resources. The remainder have only their job names and IMS information displayed.

**Note:** Use the new IMS Automation command INGIMS to achieve the same functionality: for example, INGIMS *resource* REQ=DEP

## Selecting and Viewing Subsystems

```

EVIKYDP0          SA z/OS - Command Dialogs          Line 1      of 5
Domain ID = IPSFP  ---- Dependent Regions ----      Date = 08/03/07
Operator ID = OPER                               Time = 12:44:37
Control Reg.= IMSCTL/APL/KEY4
CMD: A Update    B Start    C Stop    D INGRELS    E INGVOTE    F INGINFO
   H DISPTRG    I INGSCHED  N /ASSIGN  P /PSTOP    / scroll
CMD Name        Type System Reg.Id ITyp Trans/Step Program  IMSStatus
-----
_  IMSDBRC      APL  KEY4          DBRC
_  IMSDLS      APL  KEY4          DLS
_  IMSFP1      APL  KEY4    1    FPME NO MSG.  DFSIVP4
_  IMSMP1      APL  KEY4    2    TP           WAITING
_  IMSPPI      APL  KEY4    3    BMP  IMS941PP EVIRYPPI

Command ==>
PF1=Help    PF2=End    PF3=Return    PF6=Roll
PF9=Refresh PF10=Previous PF11=Next    PF12=Retrieve

```

Figure 6. IMS Dependent Regions Panel

For each resource the following information is shown:

### Name

The name of the resource. This is the job name from the /DIS ACTIVE display if the dependent region is not defined to SA z/OS.

### Type

The type of the resource. This is blank if the dependent region is not defined to SA z/OS.

### System

The name of the system where the resource resides. This is blank if the dependent region is not defined to SA z/OS.

The following fields are displayed on a series of screens that you can move through with the PF10 or PF11 key:

### Reg.Id

The IMS dependent region number.

### ITyp

The second type column is the IMS dependent region type.

### Trans/Step

Transaction code being processed by region, or NONE, if there are no regions of that type.

### Program

Name of the program processing in the region.

### IMSStatus

Status of the region, which can be one of the following:

Status	Meaning
UNCERTAIN	The IMS and SA statuses do not match. In particular, IMS does not know about the job name represented by the SA Resource and yet SA sees the resource as Available. This is a possible problem with the IMS dependent region.
ACTIVE-DBCMD	A /DBD or a /DBR command is in progress and waiting for the region to terminate before the /DBD or /DBR can complete.
AVAILABLE	The active threads are available. The region is available to schedule an application.
SCHEDULED	The application program is being scheduled.
TERMINATING	The application program is being terminated.



Status	Meaning
UNAVAILABLE	An active DBT thread is unavailable. An application is using the region, even though the application is not currently scheduled. This region is therefore not available to any other application.
WAITING	The MPP region is waiting for work.
WAIT-AOI	An AO application issued a GMSG call with the WAITAOI subfunction specified, but there are no messages for the AO application to retrieve.
WAIT-BLOCKMOVER	An application control block cannot be loaded because the ACB block mover is busy.
WAIT-CMD/ PENDING	A /DBDUMP, /DBRECOVERY, or /START command is in progress.
WAIT-INPUT	The application program is in WAIT-FOR-INPUT (WFI) mode.
WAIT-INTENT	The application program's intent for a database conflicts with the use of the database by a scheduled program.
WAIT-INTENT/POOL	Indicates that either the application program's intent for a database conflicts with the use of the database by a scheduled program, or a temporary shortage of DMB, PSB, or PSB work area pool space exists.
WAIT-INTENT SCHD	The IMS transaction scheduler detected an application scheduling intent, for example, Load Balancing.
WAIT-I/O PREVEN	A BMP region that accesses a GSAM database cannot schedule until I/O prevention has completed.
WAIT-MESSAGE	The application program is in a pseudo WAIT-FOR-INPUT (WFI) mode. The application is scheduled and is waiting for a message.
WAIT-POOLSPACE	A temporary shortage of DMB, PSB, or PSB work area pool space exists.
WAIT-SWITCHOVER	The alternate system is tracking the active system.
WAIT-SYNCPOINT	The application in the region is now in SYNC POINT.
WAIT-EPCB POOL	A temporary shortage of EPCB pool space exists.
WAIT-RRS PC	The application program has a protected conversation with an OTMA client that is processing a synchronization point (or sync point). Sync point can continue after the OTMA client issues either an SRRCMIT or SRRBACK call. Alternatively, the application program is part of a cascaded family and is processing a sync point. APPC/OTMA SMQ Enablement uses RRS cascaded transaction support to synchronize the backend and frontend system.

**IMS Classes**

One of the classes that is associated with the region. The region can have from one to four classes, whose values range from 1 to 999.

**Compound**

The compound status of the resource. This is a summary of all statuses of the resource and provides a single value that tells you whether the resource is OK or it has a problem.

**Observed**

The current status of the resource as observed by the automation agent.

**Desired**

The status that the automation manager is trying to move the resource to. It can be either available or unavailable.

**Automation**

The status representing the automation agents automation for the resource.

**Startable**

Indicates whether it is possible to start the resource if the automation manager is asked to do so at this point in time.

## Selecting and Viewing Subsystems

### Health

The health status of the resource. Shows N/A if the resource is not connected to a monitor.

### Auto

Shows the automation flag that is maintained by the automation manager. No automation is done for the resource by the automation manager when the flag is off.

### Hold

Shows the hold flag that is maintained by the automation manager.

### Starttype

The preset start type that will be used the next time the resource is made available (started). This value is set by INGSET and will override any TYPE value that is specified (or defaulted to) in the next INGREQ start request.

### Stoptype

The preset stop type that will be used the next time the resource is made unavailable (shut down). This value is set by INGSET and will override any TYPE value specified (or defaulted to) in the next INGREQ stop request. However, a stop type of FORCE, wherever specified, will always be honored.

### Trigger

The trigger that is associated with the resource.

### Category

Shows the category of the resource, always IMS for dependent regions.

### Subtype

The subtype of the resource. Available subtypes are: CTL, TP, DBRC, DLS, FP, BMP, FDR.

### Description

Descriptive information about the resource.

Different colors are used to indicate when a particular status is viewed as abnormal, as follows:

Color	Meaning
Blue	The desired status is UNAVAILABLE.
Pink	The compound status is DENIED or INHIBITED.
Red	The compound status is PROBLEM. The observed status is HARDDOWN or PROBLEM.
Yellow	The compound status is DEGRADED. The observed status is not in line with the desired status.

You can use one of the following command codes to invoke another command dialog:

#### A

Update resource settings (SA z/OS resources only).

#### B

Start resource (make available). SA z/OS resources will invoke INGREQ, non-SA z/OS regions will invoke INGIMS with a command of /START REGION xxxxx, where xxxxx is the job name.

#### C

Stop resource (make unavailable). SA z/OS resources will invoke INGREQ, non-SA z/OS regions will invoke INGIMS with a command of /STOP REGION JOBNAME xxxxx, where xxxxx is the job name.

#### D

Show resource relationships. (This invokes INGRELS).

#### E

Show resource requests or votes. (This invokes INGVOTE).

#### F

Show resource details. (This invokes INGINFO).

**H**

Show trigger details for resource. (This invokes DISPTRG).

**I**

Show schedules for resource. (This invokes INGSCHED).

**N**

Invoke the /ASSIGN command. Use the assign command to assign additional classes to the region. It invokes INGIMS with the command /ASSIGN CLASS *xx* REGION *nn*, where *nn* is the region number. You can overwrite the command to supply the classes.

**P**

Invoke the /PSTOP command. Use the pstop command to stop a transaction. It invokes INGIMS with the command /PSTOP REGION *nn* TRANSACTION *xx*, where *nn* is the region number and *xx* is taken from the **Trans/Step** column. You can overwrite the command before executing it.

**Note:** This action code will not check whether the control region or the dependent region supports the command. You can adjust the command to suit your purpose if the TRANSACTION format of the command is not suitable.

Additionally you can use the slash character (/) to make the selected line the first line of the display.

You can also use the FIND(F), RFIND(RF), and SORT subcommands on this panel.



---

## Chapter 14. IMS Automation Routines

The following IMS automation routines are available:

- EVIECTOX
- EVISTRCT
- EVISTRMN

---

### EVIECTOX

#### Purpose

This routine can be used to perform recovery processing for transaction and program abends in response to the application program abend message DFS554A.

The routine performs the following actions:

- Parses all data from the DFS554A
- Checks the appropriate automation flag
- Checks the code definitions for exclusions from recovery for the message types ABCODETRAN or ABCODEPROG
- Performs threshold checking for the triggering DFS554A message
- Captures received message with severity NORMAL.

The minor resource name that is used for automation flag checking and threshold checking is either *TRAN.tran* or *PROG.progid*.

If allowed, the recovery commands or replies for message DFS554A with the selection names PROG or TRAN are issued.

EVIECTOX should be called from the NetView automation table.

#### Syntax

►► EVIECTOX ◄◄

---

### EVISTRCT

#### Purpose

Captures received message (CQS0205E) with severity CRITICAL.

#### Syntax

►► EVISTRCT ◄◄

#### Usage

The EVISTRCT automation routine is intended to respond to the following message:

```
CQS0205E STRUCTURE structurename IS FULL
```

## EVISTRMN

---

### Purpose

This routine resets the posted IMS sysplex event to SDF.

This routine should be invoked from the NetView automation table.

### Syntax

➤ EVISTRMN ➤

### Usage

The automation routine EVISTRMN is intended to respond to the following message:

```
CQS0206I CQS structurename percentage BELOW THRESHOLD LEVEL
```

---

# Part 4. OMEGAMON Automation

This part gives an overview of the functions provided by OMEGAMON Automation and how to set them up.





---

## Chapter 15. \*ITM Best Practices Policy

SA z/OS supplies a sample policy, \*ITM, that provides best practice definitions for running OMEGAMON Automation.

A broad range of OMEGAMON monitors exist for a variety of z/OS products. These monitors are extremely valuable for checking the health of those products. Availability and performance data can be visualized on the IBM Tivoli Enterprise Portal (TEP). This policy provides a sample configuration to keep the OMEGAMON monitors, the Tivoli OMEGAMON Manager (TOM), the Tivoli Enterprise Monitoring Server (TEMS) highly available and to easily recycle them after configuration changes.

Diagrams of the sample policies are provided as PDF files that are located in [Add-on policies](#).

You can customize OMEGAMON Automation for your specific installation by modifying the policy database in the customization dialog. See "Enabling the SA z/OS Monitoring Agent" in *IBM System Automation for z/OS Planning and Installation*, "Importing Sample Add-On Policies" in *IBM System Automation for z/OS Defining Automation Policy*, and the online documentation for the \*ITM best practices policy for details about how to import the \*ITM best practices policy and modify the definitions that this import adds to your policy database.



---

## Appendix A. 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:

- [“The Program-to-Program Interface” on page 83](#)
- [“Program-to-Program Interface Components in NetView and CICS” on page 84](#)
- [“NetView Requests Using the Program-to-Program Interface” on page 84](#)
- [“CICS Requests Using the Program-to-Program Interface” on page 87](#)
- [“Programming Interface” on page 89](#)
- [“Customizing CICS Definitions” on page 102](#)
- [“Definition Members” on page 103](#)
- [“Security Checking Using CICS” on page 106](#)

### The 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.

The NetView 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 1, “CICS Automation Concepts,” on page 3](#) provides you with step-by-step procedures that tell you how to install these components so that the interface can be implemented.

### 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 105](#). The command processors are described in [“EVESNCCI: NetView to CICS Communication Interface” on page 90](#).

### 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 94](#), and an initialization member which is described in [“EVESPINM: CICS PPI Initialization Member” on page 104](#).

## 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.

## Program-to-Program Interface Components in NetView and CICS

Figure 7 on page 84 illustrates the CICS Automation program-to-program interface components in NetView and in CICS.

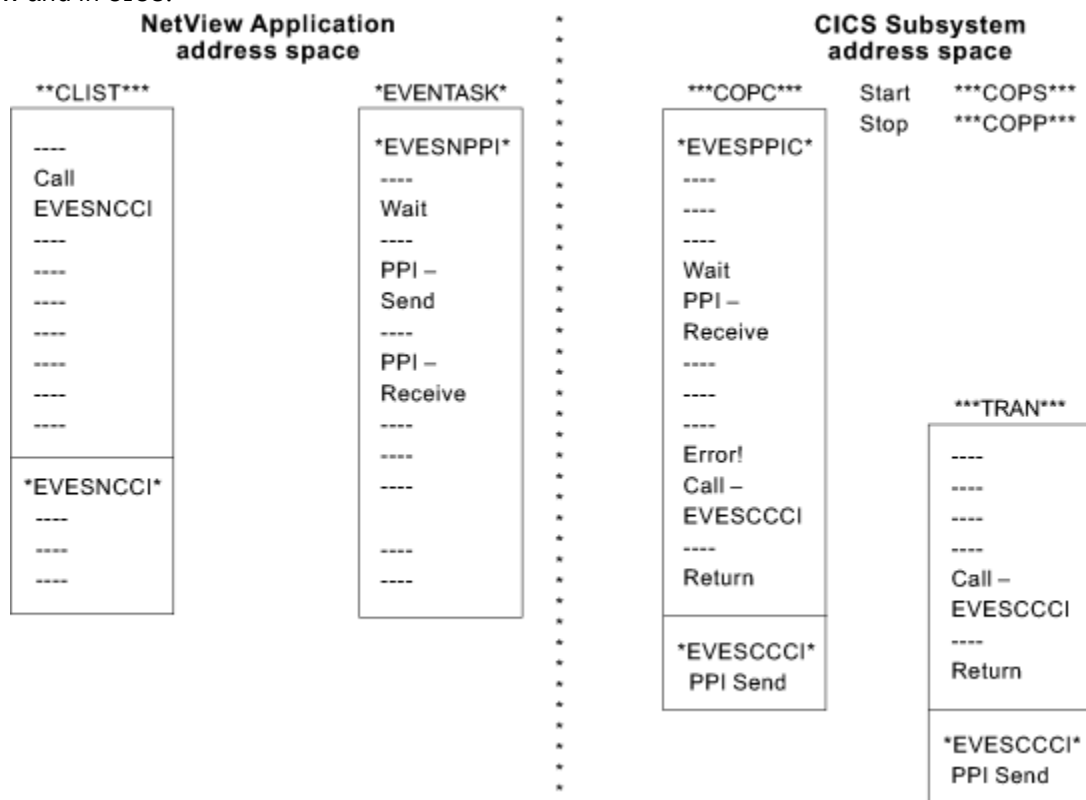


Figure 7. Program-to-Program Interface Components in NetView and CICS

Two of the programs that are shown in Figure 7 on page 84 have not been previously mentioned:

- EVESNPPI is the NetView subtask program.
- EVESPPIC is the CICS long-running receiver program.

These are internal programs that are not described in this manual.

## NetView Requests Using the Program-to-Program Interface

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 90.

The following requests from NetView are described in this section:

- [“CONVERSE from NetView” on page 85](#)
- [“SEND from NetView” on page 86](#)
- [“CANCEL from NetView” on page 87](#)

## 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.

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 follows:

### 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 that is 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.

[Figure 8 on page 86](#) illustrates the flow of events initiated by an EVESNCCI CONVERSE request.

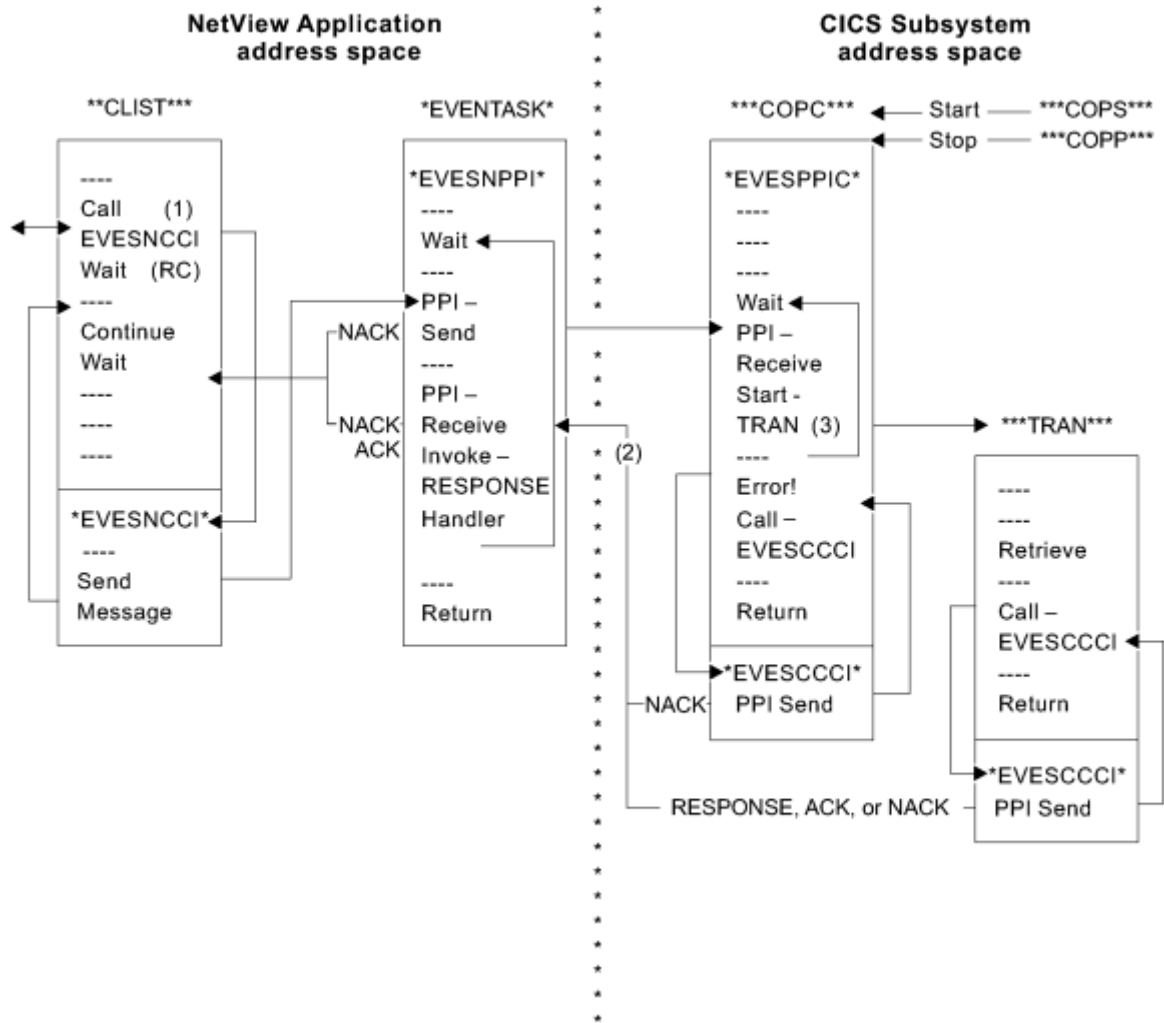


Figure 8. An EVESNCCI CONVERSE Request

**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.

**SEND from NetView**

A SEND request from a NetView command list starts a CICS transaction on the same host. No response is returned. EVESNCCI returns a message and return code 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 [Figure 9](#) on page 87.

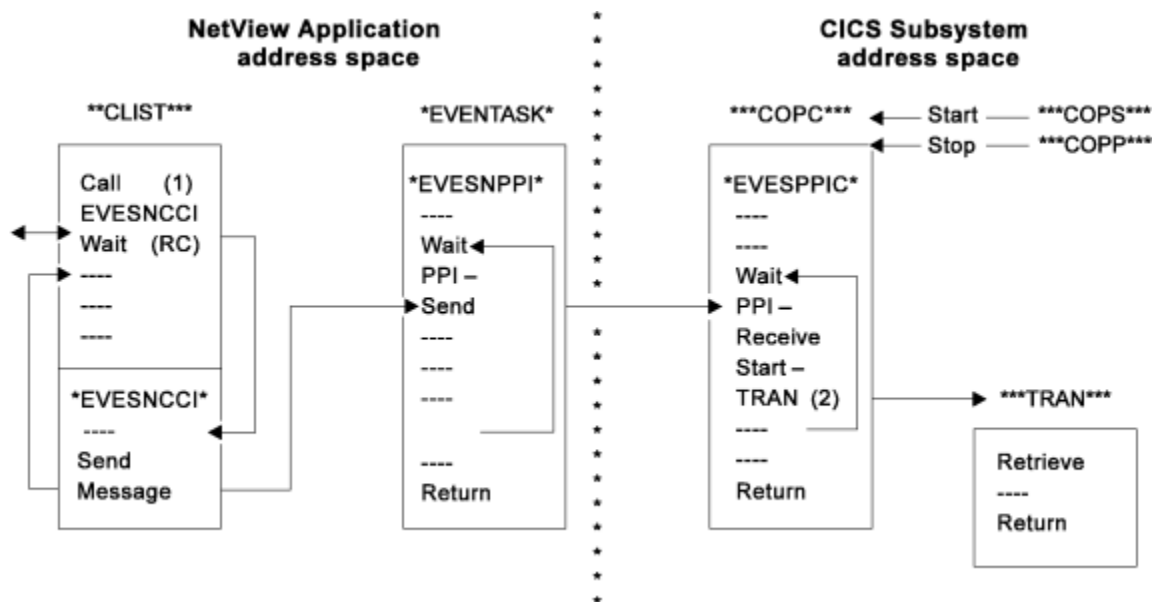


Figure 9. An EVESNCCI SEND Request

#### 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 90 (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.

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 that are 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

EVESNCCI is used for these requests. This section only provides an overview of EVESNCCI. The programming details, such as command syntax and return codes, are provided in [“EVESNCCI: CICS to NetView Communication Interface”](#) on page 94.

## 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 10 on page 88](#).

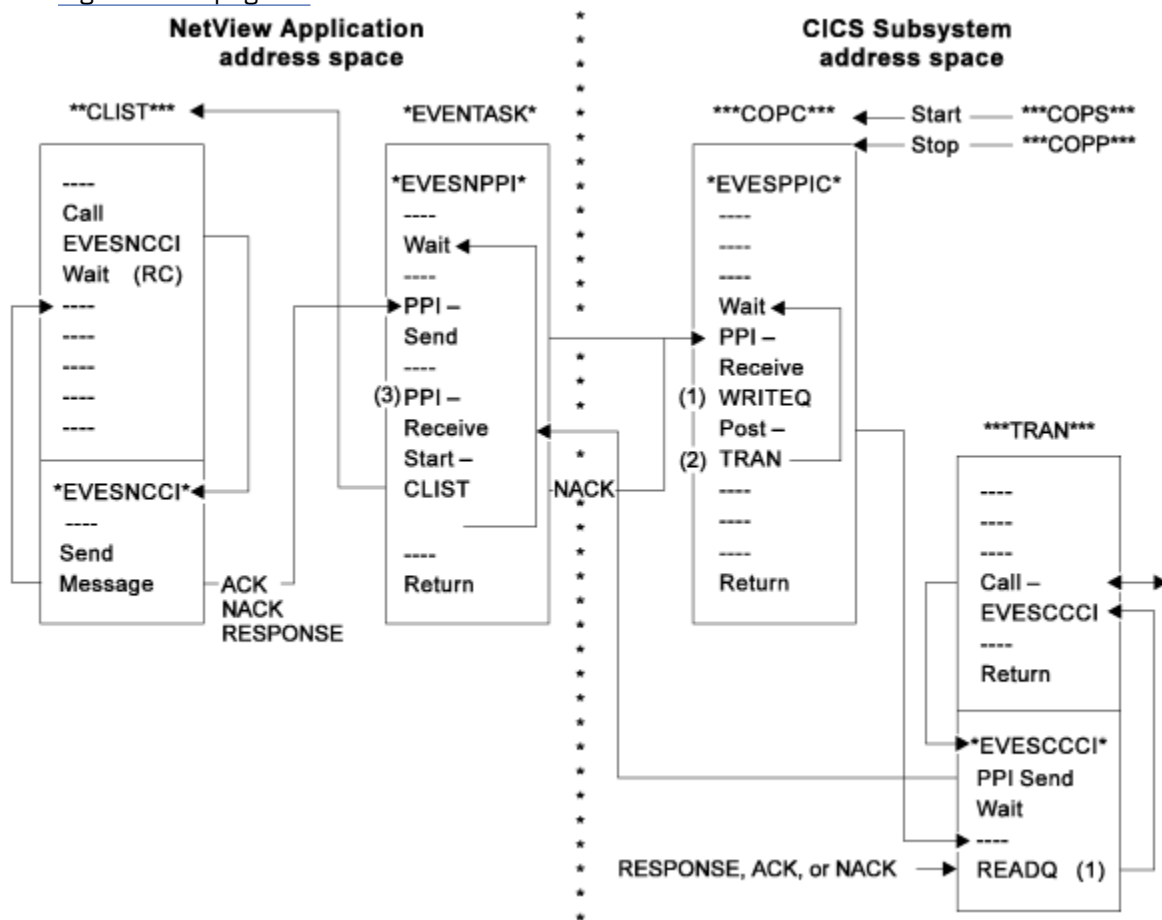


Figure 10. An EVESCCCI CONVERSE Request

### 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.

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 [Figure 11](#) on page 89.

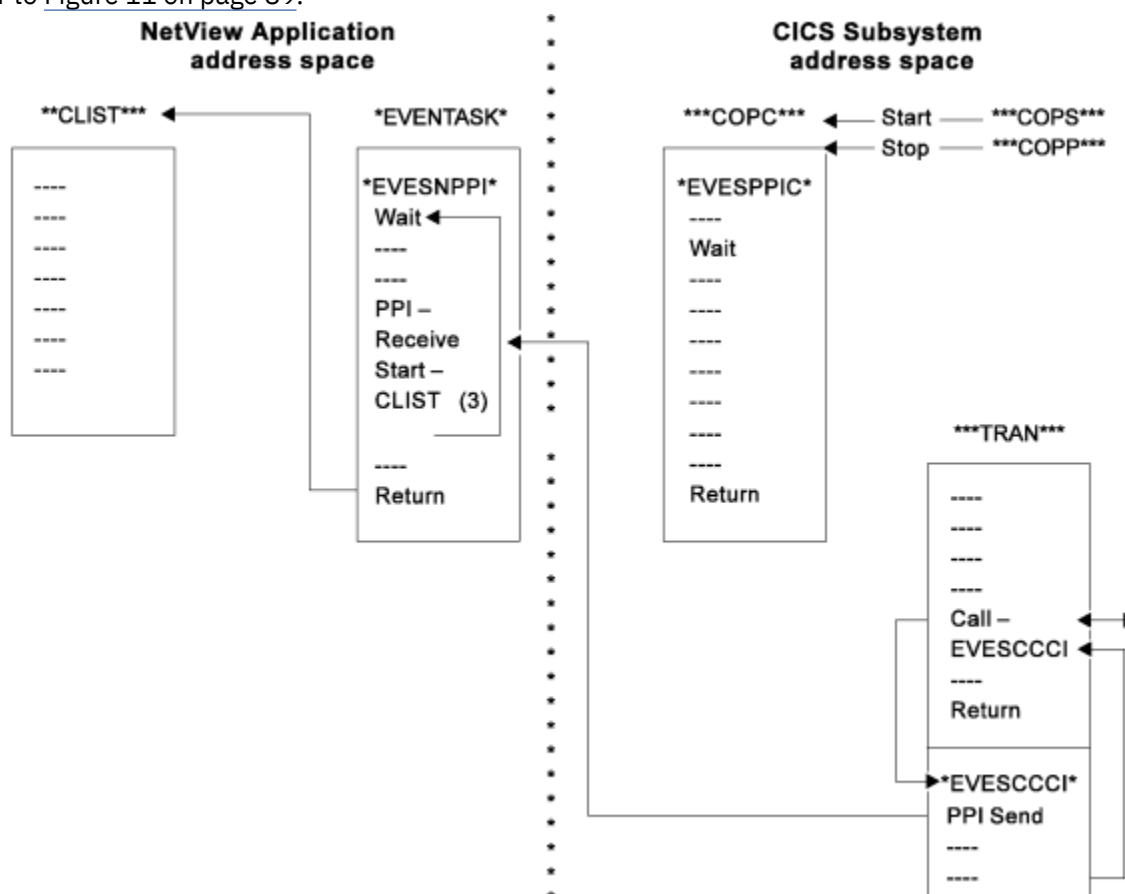


Figure 11. An EVESCCCI SEND Request

## Programming Interface

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

### **“EVESNCCI: NetView to CICS Communication Interface” on page 90**

Allows you to:

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

### **“EVESCCCI: CICS to NetView Communication Interface” on page 94.**

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 97.**

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=subnm
          ,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 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
```

**In CICS (EVESPINM):**

```
EVESPINM 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 32,656 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.
8. The following matrix shows the required (R), optional (O), and invalid (-) keywords for the various command types:

TYPE=	NAME	FUNC	DATA	OPID	DOMA	REQI	SEGM	ID
C	R	R	O	O	O	-	O <sup>1</sup>	O <sup>4</sup>
S	R	R	O	-	-	-	O <sup>1</sup>	O <sup>4</sup>
R	R	R	O	-	-	R	O <sup>1</sup>	O <sup>4</sup>

<b>TYPE=</b>	<b>NAME</b>	<b>FUNC</b>	<b>DATA</b>	<b>OPID</b>	<b>DOMA</b>	<b>REQUI</b>	<b>SEGM</b>	<b>ID</b>
N	R	R	O <sup>2</sup>	¬	¬	R	¬	¬
A	R	R	¬	¬	¬	R	¬	¬
X	¬	¬	¬	¬	¬	¬	¬	R
none	¬	¬	O	¬	¬	¬	R <sup>3</sup>	R <sup>4</sup>

**Notes:**

- a. Only SEGMENT=F can be used with these TYPEs.
- b. The length of the data must be less than 101 bytes.
- c. Only SEGMENT=M or SEGMENT=L can be used if no TYPE is specified.
- d. 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<sub>xxx</sub> 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<sub>xxx</sub> 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 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<sub>xxx</sub> REQUEST IN EVESNCCI

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

**RC=24**

ERROR ON EVESX<sub>nnn</sub> 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 progname
EVE142E FUNCTION funcname NOT FOUND IN memname
EVE171E progname : ERROR IN progname(tran), CODE = cccc
EVE175E progname : FUNCTION funcname NOT FOUND IN EVESPINM
EVE181E progname : 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,"||,
```

## EVESCCCI: CICS to NetView Communication Interface

```
"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.

## 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 8 on page 96](#) for a SEND request parameter list example):

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 97](#) 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 returns the following fields to the caller of the CONVERSE request.

<i>Table 7. EVESCCCI Fields Returned to Caller from CONVERSE Request</i>			
<b>Name</b>	<b>Disp.</b>	<b>Length</b>	<b>Contents and description</b>
OUTRESPA	044	4	Address of response area
OUTRESPL	048	4	Length of response area
OUTRTRNC	052	4	Binary return code: <b>0</b> Successful request <b>4</b> Timeout on CONVERSE <b>8</b> Program-to-program interface not available (see OUTABNDC for error code) <b>12</b> Incorrect parameter list <b>16</b> Internal processing error (see OUTABNDC for error code)
OUTABNDC	056	4	Error code (character) • OTRTRNC = 8 <b>C003</b> Program-to-program interface not active <b>C015</b> CICS LOAD/LINK failure <b>C017</b> No CICS storage available <b>C2XX</b> Program-to-program interface request error • OTRTRNC = 16 <b>A...</b> CICS abend codes <b>C012</b> CICS READQ failure <b>C016</b> CICS POST failure <b>C018</b> CICS FREEMAIN failure <b>C960</b> Incorrect TS item length <b>C961</b> RQE chain corrupted

**Note:** Refer to “EVEMPINT: EVESCCCI Parameter List Copy Book” on page 97 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 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:

<i>Table 8. EVESCCCI SEND Request Parameter List</i>			
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 97 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 90.

**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//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)  <b>Note:</b> If the name is less than eight characters, pad it with blanks.

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). <b>It is the responsibility of the caller of EVESCCCI to free the response area using EXEC CICS FREEMAIN.</b>
OUTRESPL	048	04	Contains the length of the response area addressed by the OUTRESPA field.

Name	Disp.	Length	Contents and description
OUTRTRNC	052	04	<p>Contains the return code which will have one of the following binary values:</p> <p><b>000</b> Successful request.</p> <p><b>004</b> Timeout on CONVERSE.</p> <p><b>008</b> Program-to-program interface not available (see OUTABNDC for error codes). A transaction dump is provided depending on the error code.</p> <p><b>012</b> Incorrect parameter list. No transaction dump is provided.</p> <p><b>016</b> Internal processing error (see OUTABNDC for error codes). A transaction dump is provided.</p>

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><b>C003</b> 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><b>C015</b> 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><b>C017</b> No CICS storage is available. No transaction dump is provided.</p> <p><b>C2xx</b> A program-to-program interface request error occurred, where xx 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><b>A***</b> A CICS abend has occurred. A transaction dump is provided. An error message is logged for most A*** errors.</p> <p><b>C012</b> An unexpected error has occurred on a READQ command. A transaction dump is provided. An error message is logged.</p> <p><b>C016</b> An unexpected error has occurred on a POST command. A transaction dump is provided. An error message is logged.</p> <p><b>C018</b> An unexpected error has occurred on a FREEMAIN command. A transaction dump is provided.</p> <p><b>C961</b> Internal error. Incorrect TS item length. A transaction dump is provided. An error message is logged.</p> <p><b>C962</b> 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

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

**Examples of Usage**

**Example 1**

```

*****
*          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!
*
*      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!
*
*      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!
*
*****
* 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(OUTH) 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!
*
* RESPTXT DC C'RESPONSE TEXT FROM CICS'
*
* DFHEISTG ,
*
* CISPOUTP DS CL(OUTH) 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

##### About this task

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 105 for details.

## Step 1b (Optional): Member EVESPINM

### About this task

A sample member is delivered in SINGSAMP (see [“EVESPINM: CICS PPI Initialization Member”](#) on page 104 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”](#) on page 102.
- You are using the program-to-program interface for your own transactions.

In these cases, do the following:

### Procedure

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

### About this task

A PPI receiver is required to allow CICS subsystems to communicate with NetView.

## Step 3: Define a CICS PPI Receiver Task

### About this task

A PPI receiver is required to allow NetView to communicate with CICS subsystems. This is an optional step. 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 104.**

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 105**

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.

## Definition Members

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 105.

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.

### 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 is as follows:

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

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” on page 105](#), 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 104](#).

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

## 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 94](#).
2. The function, such as CEMT.
3. The operator ID that the program runs under, for example, AUTCPPI.
4. The command list or command processor that is used for this function.

### 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=CEMT,   FUNCTION NAME
          TRANSID=COMT     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 104.

## 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.

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 104.
- 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 PLTPUSER.

**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 System Automation for z/OS Operator's Commands*.

---

## Appendix B. Notices

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

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

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

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

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

*Intellectual Property Licensing  
Legal and Intellectual Property Law  
IBM Japan Ltd.  
19-21, Nihonbashi-Hakozakicho, Chuo-ku  
Tokyo 103-8510, Japan*

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

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

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

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

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

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

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

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

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

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

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

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

#### COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work must include a copyright notice as follows:

© (your company name) (year).

Portions of this code are derived from IBM Corp. Sample Programs.

© Copyright IBM Corp. \_enter the year or years\_.

## Trademarks

---

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

## Terms and conditions for product documentation

---

Permissions for the use of these publications are granted subject to the following terms and conditions.

### Applicability

These terms and conditions are in addition to any terms of use for the IBM website.

## **Personal use**

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

## **Commercial use**

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

## **Rights**

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.



# Glossary of IMS, CICS, and Other Terms

---

This glossary defines special IMS and CICS terms used in this book 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 this book.

## A

### **abend**

Abnormal end of task.

### **ACB**

Access method control block (VTAM and VSAM).

### **access method**

A technique for moving data between main storage and input/output devices.

### **ACK**

Acknowledgement.

### **ANSI**

American National Standards Institute.

### **AOST**

Automated Operator Station Task.

### **APAR**

See [authorized program analysis report](#).

### **application program**

A program written for or by a user that applies to the user's work.

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.

### **automation**

Computer system control of operation processes.

### **authorized program analysis report (APAR)**

A request for correction of a problem caused by a defect in a current unaltered release of a program.

## B

### **batch**

An accumulation of data to be processed.

### **batch message processing**

In IMS/VS, a batch processing program that accesses online data bases and message queues.

### **BMP**

Batch Message Processing Region.

## C

### **CCTL**

Coordinator Controller.

### **CEC**

See [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.

**central processing complex (CPC)**

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 four-processor CPC, or it can be partitioned into the equivalent of two 3090 model 200s, each of which runs as a CPC with its own operating system.

**CICS**

Customer Information Control System.

**CLIST**

See [command list](#).

**CNM**

Communications Network Management.

**command**

In IMS and CICS, an instruction similar in format to a high-level programming language statement. 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.

**command list (CLIST)**

A list of commands and statements designed to perform a specific function for the user. Command lists can be written in REXX or in NetView Command List Language.

**concurrent**

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

**CPC**

See [central processing complex](#).

**D**

**database**

A collection of data fundamental to a system.

**database backout**

The function of removing changes made to user data sets by in-flight transactions.

**database recovery**

The function of restoring the user data sets, starting with a backup copy and applying all changes made to each data set after the backup was taken.

**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.

**DBCTL**

Data Base Control.

**DEDB**

Data Entry Data Base.

**DLISAS**

Data Language Interface Separate Address Space (IMS Batch).

**domain**

In IMS, a set of subsystems on a specific NetView domain defined by the system programmer in the control file.



## E

### **end user**

In IMS or CICS, anyone using IMS or 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.

## F

### **Fast Path Message Region**

In IMS, a region that executes programs that require good response characteristics and that have large transaction volumes. Message processing is grouped for load balancing and synchronized for database integrity and recovery.

### **focal point system**

In IMS, a system in which multiple subsystems are interconnected. One subsystem serves as a focal point of control, and the others are referred to as intermediate or distributed systems.

## H

### **HM**

Help Message.

### **HSBID**

Hot Standby Identifier.

### **HSSP**

High-Speed Sequential Processing.

## I

### **initial program load (IPL)**

The initialization procedure that causes an operating system to commence operation.

### **initialization**

Actions performed by IMS or CICS to construct the environment in the IMS or CICS region to enable IMS or CICS applications to be run.

A process started by SA z/OS to construct the environment that automation will occur in.

### **installation**

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.

The task of making a program ready to do useful work. This task includes generating a program, initializing it, and applying PTFs to it.

### **INSTALL/IVP**

Install/Installation Verification Procedure.

### **Integrated Resource Lock Manager (IRLM)**

In IMS automation, this facility is used as a lock manager, both as a single lock manager and in a data sharing environment.

### **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 IMS or CICS systems, and may be used for user application to user application communication, or for transparently executing IMS or CICS functions on a remote IMS or CICS system.

### **IPL**

See initial program load.

**IRC**

See [interregion communication](#).

**IRLM**

See [Integrated Resource Lock Manager](#).

**ISC**

See [intersystem communication](#).

**IVP**

Installation Verification Procedure.

**K****keyword**

A symbol that identifies a parameter.

A part of a command operand that consists of a specific character string.

An operand in a CEDA definition. Key-sequenced data set: a VSAM database organization.

**L****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.

**lock manager**

Feature of IMS automation responsible for serializing the recovery process in areas where multiple subsystems can invoke recovery actions.

**M****member**

See partitioned data set.

**MPP**

Message Processing Program.

**MSC**

See [Multiple Systems Coupling](#).

**MSDB**

Main Storage Data Base.

**MTO**

Master Terminal Operator.

**Multiple Systems Coupling (MSC)**

An IMS/VS feature that permits geographically dispersed IMS/VS systems to communicate with each other.

**N****NACK**

Negative Acknowledgement.

**NCCF**

See [Network Communications Control Facility](#).

**network**

An interconnected group of nodes.

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.

**Network Communications Control Facility (NCCF)**

IBM licensed program consisting of a base for command processors that can monitor, control, and improve network operations.

**non-XRF (non-XRF IMS)**

Represent IMS in a non-XRF configuration.

**NPDA**

Network Problem Determination Aid/Application.

**O****OLDS**

Online Log Data Set.

**online**

Pertaining to a user's ability to interact with a computer.

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.

**panel**

In IMS automation, the set of information displayed on a single screen of the user interface.

**parameter**

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

**partitioned data set (PDS)**

A data set in direct access storage that is divided into partitions, called members, each of which can contain a program, part of a program, or data. Synonymous with program library.

**PDS**

See [partitioned data set](#).

**preprocessor**

Routine in IMS automation that enables the programmer to define unique GLOBALV names to store the state value of certain processes.

**processor**

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

**program temporary fix (PTF)**

A temporary solution or bypass of a problem diagnosed by IBM field engineering as the result of a defect in a current unaltered release of the program.

**PTF**

See [program temporary fix](#).

**PUT**

Program update tape.

**R****RACF**

See [Resource Access Control Facility](#).

**RDS**

Restart Data Set.

**RECON**

Recovery Control.

**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 IMS or CICS intercommunication, a system that the local IMS or 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.

**Resource Access Control Facility (RACF)**

A licensed program that provides for access control by identifying and verifying users to the system, authorizing access to DASD data sets, logging detected unauthorized access attempts, and logging detected accesses to protected data sets.

**RMF**

Resource Management Facility.

**roll**

In IMS automation, the option to begin or rollover to another NetView session. This action is assigned to the PF6 key.

**S****SDF**

Status Display Facility. The display facility for SA z/OS.

**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.

**single-point-of-control**

Feature of IMS automation enabling the operator to monitor and control IMS subsystems from a single NetView console.

**SIT**

See [system initialization table](#).

**SLDS**

System Log Data Set.

**SMU**

Security Maintenance Utility.

**SNA**

See [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 IMS or CICS by the system operator.

**status code**

In IMS/VS, a two-character code in the program communication block (PCB) mask that indicates the results of a DL/1 call.

**subsystem**

A secondary or subordinate system.

A resource defined to SA z/OS and CICS Automation.

**system**

In IMS or CICS, an assembly of hardware and software capable of providing the facilities of IMS or 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.

**T****task**

(ISO) A basic unit of work to be accomplished by a computer.

Under IMS or CICS, the execution of a transaction for a particular user.

**TCO**

Timer-Controlled Operations.

**terminal**

A point in a system or communication network at which data can either enter or leave.

In IMS or 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.

**U****update**

To modify a file with current information.

**V****VSCR**

Virtual Storage Constraint Relief.

**VTAM**

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

**W****WTOR**

Write To Operator with Reply.

**X****XRF**

Extended recovery facility, a software function that minimizes the effects of various failures on the end users.



---

# Index

## Special Characters

- \*CICS [3](#)
- \*DB2 [33](#)
- \*IMS best practices policy [39](#)

## A

- ABCODEPROG [46](#), [55](#)
- ABCODETRAN [11](#), [17](#), [46](#), [56](#)
- abend codes [46](#)
- abend codes for CICS applications [11](#)
- ACK response [85](#)
- application components, automating recovery for
  - CICS Automation [9](#)
  - IMS Automation [45](#)
- applications
  - policy items
    - CICS CONTROL [84](#)
    - MESSAGES/USER DATA [9](#)
    - MINOR RESOURCE FLAGS [9](#)
    - MINOR RESOURCES policy item [10](#), [46](#)
- automating recovery for application components
  - CICS Automation [9](#)
  - IMS Automation [45](#)
- automation
  - operators [3](#)
- automation operator [39](#)
- autoroutine
  - EVIETOX [77](#)
  - EVISTRCT [77](#)
  - EVISTRNMN [78](#)

## C

- calling SA z/OS exit from DFSAOE00 module [41](#)
- CANCEL from NetView [87](#)
- CICS application samples [3](#)
- CICS Automation
  - automating recovery for application components [9](#)
  - CICS DB2 connection monitor resource [11](#)
  - CICS DB2 connection monitoring status messages [12](#)
  - concepts [3](#)
  - customization hints [3](#)
  - dump, EVEERDMP [24](#)
  - event monitor resource definitions [13](#)
  - event monitoring [13](#)
  - INGCICS operator command [27](#)
  - INGRMADB monitor command [11](#)
  - link monitoring [15](#)
  - message exit, using [4](#)
  - MESSAGES/USER DATA keywords [17](#)
  - monitor command [29](#)
  - monitoring of CICS DB2 connections [11](#)
  - operation hints [3](#)
  - passive monitoring [12](#)
  - relationships for event monitoring [15](#)

- CICS Automation (*continued*)
  - relationships for monitoring CICS DB2 connections [13](#)
  - routines [23](#)
  - short-on-storage recovery [24](#)
- CICS Automation concepts [3](#)
- CICS best practices policy [3](#)
- CICS DB2 connections
  - monitor resource [11](#)
  - monitoring of [11](#)
  - monitoring status messages [12](#)
- CICS dump [24](#)
- CICS link monitoring [15](#)
- CICS messages
  - defining [4](#)
  - INSERT keyword [4](#)
  - keywords [4](#)
  - MSGDISP keyword [4](#)
  - OFFSET keyword [4](#)
  - TDQUEUE keyword [4](#)
  - TOKEN keyword [4](#)
- CICS receiver program [84](#)
- CICS transactions
  - recovery [10](#)
- CICSINFO [18](#)
- CICSPlex SM address space (CMAS) [3](#)
- CICSPURG command [23](#)
- commands
  - CICSPURG [23](#)
  - EVEERDMP [24](#)
  - EVEES100 [24](#)
  - INGIMS [61](#)
  - monitor [63](#)
- concepts, CICS Automation [3](#)
- CONVERSE
  - from CICS [88](#)
  - from NetView [85](#)
- COPC [83](#)
- critical event monitoring, DB2 automation [34](#)
- customization hints
  - CICS Automation [3](#)

## D

- DB2 application samples [33](#)
- DB2 automation
  - critical event monitoring [34](#)
  - INGDB2 utility [36](#)
- DB2 best practices policy [33](#)
- defining CICS messages [4](#)
- defining IMS actions [44](#)
- defining IMS messages [42](#)
- defining reload command [8](#)
- dependent region
  - information, displaying [71](#)
- detailed status [69](#)
- DFS554A [46](#), [57](#)
- DFS AO E00 module, calling SA z/OS exit from [41](#)

displaying  
  dependent region information [71](#)  
DISPTRG [67](#)  
dump [24](#)

## E

EVEERDMP command [24](#)  
EVEERTRN automation routine  
  EVEERTRN [24](#)  
EVEES100 command [24](#)  
EVEMPINT—EVESCCCI parameter list copy book [97](#)  
event monitoring for CICS Automation [13](#)  
event-driven functions  
  critical event monitoring [34](#)  
EVENTASK [83](#), [102](#), [105](#)  
EVESCCCI - CICS to NetView communication interface [94](#)  
EVESNCCI—NetView to CICS Communication Interface [90](#)  
EVESNPPI [84](#)  
EVESPINM [103](#), [104](#)  
EVESPPIC [84](#)  
EVIETOX automation routine [77](#)  
EVISTRCT automation routine [77](#)  
EVISTRMN automation routine [78](#)  
exit router information, z/OS [40](#)

## H

health monitoring [15](#)

## I

IMS actions, defining [44](#)  
IMS Automation  
  automating recovery for application components [45](#)  
IMS automation message exit, using [42](#)  
IMS automation panels  
  IMS TCO Status [70](#)  
  Subsystem Information [69](#)  
IMS Automation panels  
  search function [67](#)  
IMS message exits, installing [40](#)  
IMS message processing [40](#)  
IMS messages, defining [42](#)  
IMS regions [39](#)  
IMS transactions  
  recovery [45](#)  
IMSINFO [57](#)  
INGCICS operator command [27](#)  
INGDB2 utility for DB2 automation [36](#)  
INGIMS [61](#)  
INGRMCDDB monitor command [11](#)  
INGSCHED [67](#)  
INSERT keyword for CICS messages [4](#)  
installing IMS message exits [40](#)

## K

keyboard [xi](#)  
keywords in CICS message definitions [4](#)

## L

LISTSHUT [19](#)

## M

main menu [67](#)  
master terminal functions  
  TCO interface [70](#)  
message exit [4](#)  
message exits, installing [40](#)  
message IDs, partial and performance [8](#), [43](#)  
message processing, IMS [40](#)  
MESSAGES/USER DATA keywords  
  ABCODEPROG [46](#), [55](#)  
  ABCODETRAN [11](#), [17](#), [46](#), [56](#)  
  CICS Automation [17](#)  
  DFS554A [46](#), [57](#)  
  IMSINFO [57](#)  
  LISTSHUT [19](#)  
  RCVRSOS [20](#)  
  RCVRTRAN [21](#)  
  TCO [58](#)  
  TCOMEMBERS [59](#)  
minor resources  
  definitions for component recovery [9](#)  
monitor command, CICS Automation [29](#)  
monitor command, INGRMCDDB [11](#)  
monitor commands [63](#)  
monitor resource, CICS DB2 connections [11](#)  
monitoring  
  CICS DB2 connections [11](#)  
  CICS health [15](#)  
  event monitor resource definitions for CICS Automation [13](#)  
  events for CICS Automation [13](#)  
  online log data sets [47](#)  
  passive, CICS Automation [12](#)  
  recovery control data sets [50](#)  
  VTAM ACB [51](#)  
monitoring status messages, CICS DB2 connections [12](#)  
MSGDISP keyword  
  for CICS messages [4](#)

## N

NACK response [85](#)  
NetView subtask program [84](#)

## O

OFFSET keyword  
  for CICS messages [4](#)  
OLDS  
  monitoring [47](#)  
operation hints  
  CICS Automation [3](#)

## P

partial message IDs and performance [8](#), [43](#)  
passive monitoring, CICS Automation [12](#)  
performance, partial message IDs and [8](#), [43](#)



- policy data, refreshing [8](#), [43](#)
- PPI (see program-to-program interface) [83](#)
- PPI receiver task [103](#)
- processing, IMS messages [40](#)
- program-to-program interface
  - CICS components [83](#)
  - CICS requests
    - CONVERSE [88](#)
    - SEND [89](#)
  - communication components [84](#)
  - EVENTASK [83](#), [102](#), [105](#)
  - EVESPINM
    - customization [103](#)
  - NetView requests
    - CANCEL [87](#)
    - CONVERSE [85](#)
    - SEND [86](#)

## R

- RCVRSOS [20](#)
- RCVRTRAN [21](#)
- RECONS
  - monitoring [50](#)
- recovery
  - abend codes [46](#)
  - abend codes for CICS applications [11](#)
  - CICS transactions [10](#)
  - DFS554A [46](#)
  - IMS transactions [45](#)
  - minor resource flags for CICS applications [10](#)
  - minor resources [45](#)
  - short-on-storage condition [9](#)
  - thresholds [46](#)
  - thresholds for CICS applications [10](#)
- recovery for application components, automating
  - CICS Automation [9](#)
  - IMS Automation [45](#)
- refreshing policy data [8](#), [43](#)
- relationships
  - CICS event monitoring [15](#)
  - monitoring of CICS DB2 connections [13](#)
- reload command, defining [8](#)
- RESPONSE response [85](#)
- routines
  - CICS Automation [23](#)

## S

- SA z/OS exit
  - calling from DFSAOE00 module [41](#)
  - using without z/OS exit router [41](#)
- SA z/OS operator
  - commands
    - DISPTRG [67](#)
    - INGSCHED [67](#)
- security checking [106](#)
- segment support in NetView [90](#)
- selecting a subsystem [68](#)
- SEND
  - from CICS [89](#)
  - from NetView [86](#)
- short-on-storage recovery function, EVEES100 [24](#)

- shortcut keys [xi](#)
- subsystems
  - selecting [68](#)
- system initialization table [106](#)

## T

- TCO [58](#)
- TCO functions [70](#)
- TCOMEMBERS [59](#)
- TDQUEUE keyword for CICS messages [4](#)
- Time Controlled Operations (TCO) interface [70](#)
- TOKEN keyword
  - for CICS messages [4](#)
- transient data queue exit [4](#)

## U

- using
  - CICS Automation message exit [4](#)
- using SA z/OS exit without z/OS exit router [41](#)

## V

- VTAM ACB, monitoring [51](#)

## X

- XRF environment
  - IMSINFO keyword [57](#)

## Z

- z/OS exit router information [40](#)
- z/OS exit router, using SA z/OS exit without [41](#)







SC34-2714-01

