

System Automation for OS/390



CICS Automation Programmer's Reference and Operator's Guide

Version 2 Release 1

System Automation for OS/390



CICS Automation Programmer's Reference and Operator's Guide

Version 2 Release 1

Note!

Before using this information and the product it supports, be sure to read the general information under "Notices" on page ix.

Fourth Edition (October 2000)

This edition applies to System Automation for OS/390 Version 2 Release 1 (5645-006), and to AOC/MVS CICS Automation Version 1 Release 4 (5685-151), IBM licenced programs, and to all subsequent releases and modifications until otherwise indicated in new editions.

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

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

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

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

Internet: s390id@de.ibm.com
World Wide Web: <http://www.s390.ibm.com/os390/>

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

© Copyright International Business Machines Corporation 1990, 2000. All rights reserved.

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

Contents

Figures v

Tables vii

Notices ix

Programming Interface Information ix

Trademarks x

About This Book. xi

Who Should Use This Book xi

What's in This Book xi

Notation for Format Descriptions xi

Related Publications xii

 The System Automation for OS/390 Library xii

 Related Product Information xiii

Part 1. Introducing CICS Automation 1

Chapter 1. Principal Concepts of SA OS/390 3

Automation Policies 3

Goal-Driven Automation 4

Dependencies, Request Propagation, and Desired State 4

Persistency of Requests and Conflicting Requests 6

Triggers 7

Service Periods 8

Application Groups 8

SA OS/390 and the NetView Message Automation Table 9

Chapter 2. Special Functions of CICS Automation 11

Link Monitoring 11

Health Checking 11

State/Action Tables 11

Recovery 13

Program-to-Program Interface 13

 NetView Components 13

 CICS Components 13

 Communication Components 13

Part 2. Customizing CICS Automation 15

Chapter 3. Customizing CICS Automation 17

Installation-Related Steps 17

 Step 1: Update MPFLSTxx in SYS1.PARMLIB 17

Tailor Your Network NetView 17

 Step 1: Tailor Your Network NetView for CDRM Recovery 17

Define CICS Automation to CICS 17

 Step 1: SIT or Startup Overrides 18

 Step 2: Program List Table Definitions 18

 Step 3: Define Consoles 18

 Step 4: Transaction and Program Definitions 19

 Step 5: DFHRPL and the CICS Automation Library 20

CICS Automation Definitions in NetView 20

 Step 1: Basic CICS Automation Common Policy Definitions 20

 Step 2: Basic CICS Application Definitions 21

 Step 3: The Program-to-Program Interface Initialization Member (EVENTASK) 22

 Step 4: Adding a New CICS Automation Status File Record 22

 Step 5: Installing CICSplex SM REXX API 23

Extended CICS Definitions 23

 Step 1: Health Check Programs 23

 Step 2: Program-to-Program Interface Initialization Member (CICS) 23

 Step 3: Add or Change the CICS Transient Data Messages 24

 Step 4: Echoplex Back-End Programs 24

 Step 5: Security Considerations 24

CICS Automation Definitions for CICSplex System Manager (CPSM) 25

 Automating Coordinating Address Space (CAS) Startup and Shutdown 25

 Automating CICSplex SM Address Space (CMAS) Startup and Shutdown 25

Migration and Coexistence 26

 Migration 26

 Coexistence between V1R4 and V2R1 28

Chapter 4. How to Set Up the Special Functions of CICS Automation 29

Defining the SDF States for CICS Automation 29

Automating Recovery for Transactions and Problem Areas 30

 How to Define Transaction Recovery 31

 How to Set Up the State/Action Tables 35

 How to Set Up Health Checking 36

 How to Set Up Link Monitoring 37

 Setting Up Echoplexing 38

 Security Checking Using CICS 38

Adding Local Applications to the CICS Automation Operator Interface 39

Using Linemode Functions 40

 Health Checking 41

 SIT Override 41

 Link Monitoring 41

 Message options 41

 CICSPOST 41

 CEMTPPI 41

How to Implement Remote Site Recovery for VSAM RLS (CICS TS Function Only)	41
Special Considerations for RACF-Protected Subsystems	42

**Chapter 5. MESSAGES/USER DATA
Entries for CICS Automation 45**

Translating Format Descriptions into MESSAGES/USER DATA Entries	45
CICS-Specific MESSAGES/USER DATA Keywords	50
ABCODESYSTM—System abend recovery	52
ABCODETRAN—Transaction Abend Recovery	54
HEALTHCHK—Health Checking	56
LISTSHUT—Transaction Purging During Shutdown	58
RCVRAUTO—Autoinstall Time Limit	59
RCVRSOS—Short-On-Storage Handling	60
RCVRTRAN—Transaction Recovery	62
RCVRVIOL—Storage Violation Handling	63

**Chapter 6. CICS Automation Routines,
Commands, and Definition Members . . . 65**

Subroutines	66
CICSQRY—Name Lookup	67
CICSRCMD—Request a CICS Function	70
Commands and Common Routines	70
CEMTPPI—CEMT PPI Short Syntax	72
CICSDLY—Change the Shutdown Delay Time	73
CICSPOST—Post An External Event	74
CICSPURG—Purge Transactions	75
CICRSYC—CICS Resync	76
CICSSHUT—Shutdown Processor	77
EVEED003—Critical Message Handler for the Status Display Facility	79
EVEERDMP—CICS Dump	80
EVEEMIGR—Migrate Subsystem to CICS/TS	81
EVEEY00S—Common State Handler for State/Action Tables	82
CICSHLTH—Linemode Health Checking	83
CICSOVRD—Linemode SIT Override	85
CICSLM—Linemode Link Monitor	87
CICSMHDR—Linemode Message Options	90
CMASSHUT—CICSplex SM Address Space (CMAS) Shutdown	92
Definition Members	92
EVESPINM—CICS PPI Initialization Member	94
EVENTASK—NetView PPI Initialization Member	96
EVESCMT3—Message Exit Table for CICS	98

Part 3. Using CICS Automation 101

**Chapter 7. Using Panels and Working
with Subsystems 103**

Using CICS Automation Panels	103
Panel Characteristics	103
Using the Main Menu	104

Using Fast Path	106
Selecting and Viewing Subsystems	106
Selecting a Subsystem	106
Getting Detailed Status	107

**Chapter 8. Starting and Stopping
Resources 113**

Startups	113
Shutdowns	116

**Chapter 9. Monitoring Your CICS
Subsystems 117**

Link Monitoring	117
Displaying Links	118
Health Checking	121

Chapter 10. Broadcasting Messages 125

Chapter 11. Support Functions 127

Cold Start Indicator	127
V2R1 Target System	127
V1R3 Target System	128
Program-to-Program Interface	129
Message Options	130

**Chapter 12. The Status Display
Facility 133**

**Appendix. CICS Automation and the
Program-to-Program Interface 137**

Program-to-Program Interface Components in NetView and CICS	137
NetView Requests Using the Program-to-Program Interface	138
CONVERSE from NetView	138
SEND from NetView	140
CANCEL from NetView	141
CICS Requests Using the Program-to-Program Interface	142
CONVERSE from CICS	142
SEND from CICS	143
Programming Interface	144
EVESNCCI—NetView to CICS Communication Interface	146
EVESNRSP - Common Response Handler from CICS	151
EVESCCCI - CICS to NetView Communication Interface	152
EVEMPINT—EVESCCCI Parameter List Copy Book	155

Glossary of CICS and Other Terms 161

Index 165

Figures

1. Example of Start Dependencies	5	20. Input Panel for the INGREQ Command	113
2. Example of Conflicting Requests	6	21. Verification Panel for INGREQ	115
3. Example of a Request Involving a Group	9	22. Input Panel for INGREQ Command	116
4. Short-on-Storage State/Action Table Example	12	23. CICS Automation Monitoring Panel	117
5. Defining Minor Resources for Transactions	32	24. Monitoring Links Panel	118
6. Automation Flag Panel.	32	25. Display Links Panel	119
7. Thresholds Definitions Panel.	33	26. Health Checking Panel	122
8. Code Processing Panel	34	27. Broadcast Messages Panel	125
9. Command Processing Panel	34	28. Broadcast Notification—Cancelled Example	126
10. Defining a Health Check Program to CICS Automation	37	29. Support Functions	127
11. Message Processing Panel of the Customization Dialogs 1	46	30. Update Panel of INGLIST	128
12. CMD Processing Panel of the Customization Dialogs	47	31. Set Cold Start Indicator Panel (Pre-CICS TS V1R1 Subsystem)	129
13. User-Defined Data Panel of the Customization Dialogs	48	32. Set Cold Start Indicator Panel (CICS TS V1R1 Subsystem and Above)	129
14. Message Processing Panel of the Customization Dialogs 2	49	33. Program-to-Program Interface	130
15. Code Processing Panel of the Customization Dialogs	50	34. Set Message Id Display Panel	131
16. Common Characteristics of CICS Automation Panels	103	35. Status Display Facility Main Panel	134
17. CICS Automation Main Menu	104	36. The CICS Monitor Panel	135
18. Selection Panel for CICS Resources	107	37. Program-to-Program Interface Components in NetView and CICS.	138
19. Subsystem Information Panel	108	38. An EVESNCCI CONVERSE Request	140
		39. An EVESNCCI SEND Request	141
		40. An EVESCCCI CONVERSE Request	143
		41. An EVESCCCI SEND Request	144

Tables

1. System Automation for OS/390 Library	xii	7. EVEEMIGR Return Codes.	81
2. Related Products Books	xiii	8. EVESCCCI CONVERSE Request Parameter List	152
3. CICS-Specific Correspondences between ACF Keywords and Policy Objects/Items	26	9. EVESCCCI Fields Returned to Caller from CONVERSE Request	153
4. Minor Resource Names for Problem Areas	30	10. EVESCCCI SEND Request Parameter List	154
5. CICSQRY Return Codes	68		
6. CICSRCMD Return Codes	70		

Notices

References in this publication to IBM[®] products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. 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 of the intellectual property rights of IBM may be used instead of the IBM product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, are the responsibility of the user.

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

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

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

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

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

Programming Interface Information

This book documents programming interfaces that allow the customer to write programs to obtain the services of System Autuation for OS/390.

Trademarks

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

CICS	DB2	NetView
CICS/ESA	IBM	OS/390
CICS/MVS	IMS/ESA	RACF
CICSplex	MVS	VTAM

NetView is a trademark of Tivoli Systems Inc. in the United States, or other countries, or both.

About This Book

This book describes how to customize and operate CICS® Automation. CICS Automation is a feature of System Automation for OS/390 that provides a simple and consistent way to monitor and control all of the CICS regions, both local and remote, within your organization. This automation feature automates, simplifies, and standardizes console operations and the management of component, application, and production related tasks.

Who Should Use This Book

This book is intended for two kinds of users/user groups:

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

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

Installing and customizing CICS Automation requires a programmer's understanding of NetView®, CICS, SA OS/390, and CICS Automation, 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 subsystems.

These users will mainly need part 1 and part 3.

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

What's in This Book

This book contains the following:

Part 1. Introducing CICS Automation

Explains some main concepts of SA OS/390 and describes the special functions of CICS Automation.

Part 2. Customizing CICS Automation

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

Part 3. Using CICS Automation

Describes the operator interface of CICS Automation.

Notation for Format Descriptions

The reference sections of this manual contain format descriptions of commands and of entries in the SA OS/390 policy database. The notation used for these descriptions is as follows:

- Items shown in braces { } represent alternatives. You must choose one. For example,

{A|B|C}

indicates that you must specify one item only: A, B, or C.

- Items shown in brackets [] are optional. You may choose one. For example,

[A|B|C]

indicates that you may enter A, B, or C, or you may omit the operand.

- A series of three periods (...) indicates that a variable number of items may be included in the list.
- An underscored item shows the default that the system will choose if you do not specify an item. For example,
[A|B|C]

indicates that if no operand is specified, B is assumed.

- Lowercase italicized items are variables; substitute your own value for them.
- Uppercase items must be entered exactly as shown.
- Parentheses must be entered as shown.
- Where operands can be abbreviated, the abbreviations are shown in capital letters. For example, ALL can be entered as A or ALL.
- Commas are used as delimiters between parameters. The last parameter does not require a comma after it. Because of this, we place the comma in front of a parameter to show that if you add this parameter, you need a comma as, for example, in
XYZ [A[,B[,C]]]

However, the comma actually goes after the preceding parameter and needs to be on the same line as that parameter.

Related Publications

The System Automation for OS/390 Library

The following table shows the information units in the System Automation for OS/390 library:

Table 1. System Automation for OS/390 Library

Title	Order Number
<i>System Automation for OS/390 General Information</i>	GC33-7036
<i>System Automation for OS/390 Licensed Program Specifications</i>	SC33-7037
<i>System Automation for OS/390 Planning and Installation</i>	SC33-7038
<i>System Automation for OS/390 Customizing and Programming</i>	SC33-7035
<i>System Automation for OS/390 Defining Automation Policy</i>	SC33-7039
<i>System Automation for OS/390 User's Guide</i>	SC33-7040
<i>System Automation for OS/390 Messages and Codes</i>	SC33-7041
<i>System Automation for OS/390 Operator's Commands</i>	SC33-7042
<i>System Automation for OS/390 Programmer's Reference</i>	SC33-7043
<i>System Automation for OS/390 CICS Automation Programmer's Reference and Operator's Guide</i>	SC33-7044
<i>System Automation for OS/390 IMS Automation Programmer's Reference and Operator's Guide</i>	SC33-7045
<i>System Automation for OS/390 OPC Automation Programmer's Reference and Operator's Guide</i>	SC23-7046

The System Automation for OS/390 books (except Licensed Program Specifications) are also available on CD-ROM as part of the following collection kit:

IBM Online Library OS/390 Collection (SK2T-6700)

This softcopy collection includes the IBM Library Reader, a program that enables you to view online documentation.

SA OS/390 Homepage

For the latest news on SA OS/390, visit the SA OS/390 homepage at <http://www.s390.ibm.com/products/sa/>

Related Product Information

The following table shows the books in the related product libraries that you may find useful for support of the SA OS/390 base program.

Table 2. Related Products Books

Title	Order Number
<i>MVS/ESA MVS Configuration Program Guide and Reference</i>	GC28-1817
<i>MVS/ESA Planning: Dynamic I/O Configuration</i>	GC28-1674
<i>MVS/ESA Support for the Enterprise Systems Connection</i>	GC28-1140
<i>MVS/ESA Planning: APPC Management</i>	GC28-1110
<i>MVS/ESA Application Development Macro Reference</i>	GC28-1822
<i>MVS/ESA SP V5 System Commands</i>	GC28-1442
<i>MVS/ESA SPL Application Development Macro Reference</i>	GC28-1857
<i>OS/390 Hardware Configuration Definition: User's Guide</i>	SC28-1848
<i>OS/390 Information Roadmap</i>	GC28-1727
<i>OS/390 Information Transformation</i>	GC28-1985
<i>OS/390 Introduction and Release Guide</i>	GC28-1725
<i>OS/390 JES Commands Summary</i>	GX22-0041
<i>OS/390 Licensed Program Specifications</i>	GC28-1728
<i>OS/390 Printing Softcopy Books</i>	S544-5354
<i>OS/390 Starting Up a Sysplex</i>	GC28-1779
<i>OS/390 Up and Running!</i>	GC28-1726
<i>Planning for the 9032 Model 3 and 9033 Enterprise Systems Connection Director</i>	SA26-6100
<i>Resource Access Control Facility (RACF) Command Language Reference</i>	SC28-0733
<i>S/390 MVS Sysplex Overview – An Introduction to Data Sharing and Parallelism</i>	GC23-1208
<i>S/390 MVS Sysplex Systems Management</i>	GC23-1209
<i>S/390 Sysplex Hardware and Software Migration</i>	GC23-1210
<i>S/390 MVS Sysplex Application Migration</i>	GC23-1211
<i>S/390 Managing Your Processors</i>	GC38-0452
<i>Tivoli/Enterprise Console User's Guide Volume I</i>	GC31-8334
<i>Tivoli/Enterprise Console User's Guide Volume II</i>	GC31-8335
<i>Tivoli/Enterprise Console Event Integration Facility Guide</i>	GC31-8337
<i>Tivoli for OS/390 NetView V1R3 Administration Reference</i>	SC31-8222
<i>Tivoli for OS/390 NetView V1R3 Application Programmer's Guide</i>	SC31-8223

Table 2. Related Products Books (continued)

Title	Order Number
<i>Tivoli for OS/390 NetView V1R3 APPN Topology and Accounting Agent Guide</i>	SC31-8224
<i>Tivoli for OS/390 NetView V1R3 Automation Guide</i>	SC31-8225
<i>Tivoli for OS/390 NetView V1R3 AON Customization Guide</i>	SC31-8662
<i>Tivoli for OS/390 NetView V1R3 AON User's Guide</i>	GC31-8661
<i>Tivoli for OS/390 NetView V1R3 Bridge Implementation</i>	SC31-8238
<i>Tivoli for OS/390 NetView V1R3 Command Reference Vol. 1</i>	SC31-8227
<i>Tivoli for OS/390 NetView V1R3 Command Reference Vol. 2</i>	SC31-8227
<i>Tivoli for OS/390 NetView V1R3 Customization Guide</i>	SC31-8228
<i>Tivoli for OS/390 NetView V1R3 Customization: Using Assembler</i>	SC31-8229
<i>Tivoli for OS/390 NetView V1R3 Customization: Using Pipes</i>	SC31-8248
<i>Tivoli for OS/390 NetView V1R3 Customization: Using PL/I and C</i>	SC31-8230
<i>Tivoli for OS/390 NetView V1R3 Customization: Using REXX and the NetView Command List Language</i>	SC31-8231
<i>Tivoli for OS/390 NetView V1R3 Data Model Reference</i>	SC31-8232
<i>Tivoli for OS/390 NetView V1R3 Installation and Administration Guide</i>	SC31-8236
<i>Tivoli for OS/390 NetView V1R3 Messages and Codes</i>	SC31-8237
<i>Tivoli for OS/390 NetView V1R3 MultiSystem Manager User's Guide</i>	SC31-8607
<i>Tivoli for OS/390 NetView V1R3 NetView Graphic Monitor Facility User's Guide</i>	GC31-8234
<i>Tivoli for OS/390 NetView V1R3 NetView Management Console User's Guide</i>	GC31-8665
<i>Tivoli for OS/390 NetView V1R3 User's Guide</i>	SC31-8241
<i>Tivoli for OS/390 NetView V1R3 Planning Guide</i>	GC31-8226
<i>Tivoli for OS/390 NetView V1R3 RODM and GMFHS Programmer's Guide</i>	SC31-8233
<i>Tivoli for OS/390 NetView V1R3 Security Reference</i>	SC31-8606
<i>Tivoli for OS/390 NetView V1R3 SNA Topology Manager and APPN Accounting Manager Implementation Guide</i>	SC31-8239
<i>Tivoli Management Platform Reference Guide</i>	GC31-8324
<i>TSO/E REXX/MVS User's Guide</i>	SC28-1882
<i>TSO/E REXX/MVS Reference</i>	SC28-1883
<i>VM/XA SP GCS Command and Macro Reference</i>	SC23-0433
<i>VSE/SP Unattended Node Support</i>	SC33-6412
<i>VTAM Messages and Codes</i>	SC31-6493
<i>VTAM Network Implementation Guide</i>	SC31-6404
<i>VTAM Network Implementation Guide</i>	SC31-6434

Part 1. Introducing CICS Automation

This part describes principal concepts of SA OS/390, including some NetView-related information, and gives an overview of the additional facilities offered by CICS Automation.

Chapter 1. Principal Concepts of SA OS/390

This section sketches some fundamentals of SA OS/390[®]. For more detailed information see the SA OS/390 documentation.

Automation Policies

System automation primarily deals with starting and stopping applications in accordance with their interrelationships. These interrelationships include dependencies of applications on other applications as well as being a component application of an application complex. Also, system automation supports permanent availability of an application by moving the application to another system in case of an unrecoverable abend (see “Application Groups” on page 8).

All applications and systems that you want to include in automation must be defined to SA OS/390 in an automation *policy database*. This database contains the objects to be managed by SA OS/390, and the rules according to which automation of these objects proceeds. You access the policy database from the so-called *customization dialogs*. The customization dialogs are described in *System Automation for OS/390 Defining Automation Policy*.

The objects that are defined in the policy database are called *policy objects* or *entries*. Applications and systems, for example, are policy objects. Every policy object belongs to an *entry type* which is identified by a three letter code; thus, applications belong to the entry type APL.

Policy objects have automation-related properties and are associated with one another; these properties and connections are called *policy items*. For example, there is a policy item STARTUP for applications that specifies how SA OS/390 is to start the application.

What you enter in the policy database are policy objects. However, the objects that can be automated are not these policy objects, but so-called *resources*, which are automatically generated from the policy objects.

This is especially important in the case of applications, since the resources that correspond to an application always represent a *subsystem*, that is, a combination of the application with a system on which it is intended to run; thus, one application can correspond to several subsystems. These resources are generated when an application is linked to a system in the policy database. Note also that some properties and connections are defined on the application (policy object) level (see “Triggers” on page 7) and handed down to all corresponding resources, while others are specified at the resource level (see “Dependencies, Request Propagation, and Desired State” on page 4), and therefore only apply to that resource.

The names of the resources have the following format:

resource_name/entry_type[/system_name]

The most common entry types are APL (application), APG (application group), and SYS (system). The system name is omitted when the resource is associated with a sysplex, and not a single system.

The policy database must be converted into an *automation control file* (ACF) in order to be accessible to SA OS/390.

Goal-Driven Automation

A basic concept of SA OS/390 is to distinguish between the *desired* state of a resource and (broadly speaking) its *actual* state. Every resource has a desired state, which is either AVAILABLE or UNAVAILABLE; AVAILABLE is the default. This desired state, which is also called the automation *goal*, can be different from the actual state; a resource whose desired state is to be running (AVAILABLE), can actually be down. SA OS/390 always tries to keep the actual state in line with the desired state, but sometimes this is not possible.

SA OS/390 is called *goal driven* because all requests that can be made to it from the outside refer to the desired state of the target resource. When an operator passes a start request for a resource to SA OS/390, this is a request to set the desired state of the resource to AVAILABLE. It is up to SA OS/390 to decide whether (1) this is at all possible, and if so, whether (2) the actual state can be modified accordingly:

1. Making a request does not automatically lead to a change of the desired state of the target resource. Rather, SA OS/390 compares the *priority* of the new request with that of the last successful request. Only when the new request has a higher priority does SA OS/390 change the desired state of the resource. Note that this presupposes that the old request is still available. For more details on this topic, see “Persistence of Requests and Conflicting Requests” on page 6.
2. The latter decision mainly depends on the *dependencies* between the target resource and other resources, and on the *triggers* that may have been associated with it. Dependencies and triggers are defined in the policy database. For more information, see “Dependencies, Request Propagation, and Desired State”, and “Triggers” on page 7.

Dependencies, Request Propagation, and Desired State

One of the main tasks of system automation when starting or stopping a resource is to consider the dependencies that exist between the resource to be started/stopped and other resources. Certain resources can only be started when certain other resources are already running (start dependencies), and certain resources can only be stopped when certain other resources are already down (stop dependencies). Note that start and stop dependencies are in principle independent of each other, although if A can only be started when B is running, then it will, as a rule, not be possible to stop B unless A has been stopped beforehand.

Such dependencies can be specified in the policy database. The only restriction is that the dependent and the supporting resource must belong to the same sysplex (they need *not* reside on the same system). SA OS/390 takes dependencies into account when it is requested to start or to stop a resource. By default, it will try to start/stop all resources on which the target resource of the request directly or indirectly depends. The mechanism by which this is accomplished is called *request propagation*. It is best explained by an example.

Example 1: Let A, B, and C be resources so that A can only be started when B is running, and B can only be started when C is running. C is supposed

to have no start dependencies. Suppose, furthermore, that A, B, and C are all actually down, and that this conforms to their desired state (which is UNAVAILABLE).

Finally, assume that A, B, and C are not associated with any trigger (for the significance of this, see “Triggers” on page 7), and that there are no requests pending for any of the three resources (see “Persistence of Requests and Conflicting Requests” on page 6).

This situation is displayed in Figure 1. The labels of the arrows specify the dependency type. **MakeAvailable/WhenAvailable** is the format in which SA OS/390 specifies that the dependent (lower) resource, which is referred to by **MakeAvailable**, can only be started when the supporting (upper) resource, referred to by **WhenAvailable**, is running.

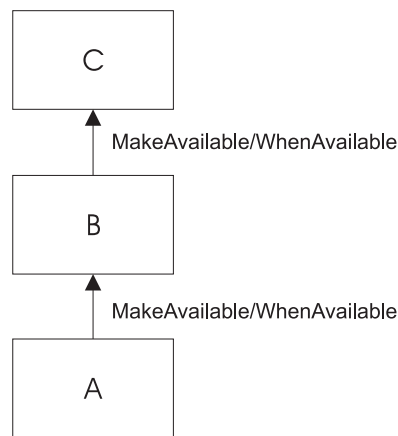


Figure 1. Example of Start Dependencies

When SA OS/390 receives a request to start A, the following chain of events will occur:

1. The request is propagated:
 - a. Since A can only be started when B is running, a start request is put to B.
 - b. Since B can only be started when C is running, a start request is put to C.
2. In response to these requests, the desired state of all three resources is changed to AVAILABLE.
3. SA OS/390 tries to change the actual state of the resources according to their desired state:
 - a. At first, only C, which has no start dependencies, can be started. B and A cannot be started because C and B are not yet running.
 - b. Then B will be started, because C is now available.
 - c. Finally, A is started.

The propagated requests are usually called *votes* instead of requests.

In example 1, the request propagation is uniform; the desired state of all three resources is set to AVAILABLE because the condition of the dependency relationships is **WhenAvailable** in both cases. This is not always the case, as the following example shows.

Example 2: Modify example 1 to the effect that B can only be started when C is *unavailable*, and that C is running, in accordance with its desired state AVAILABLE, when the request comes in.

To reflect this modification, the upper arrow label of Figure 1 on page 5 would have to be changed to **MakeAvailable/WhenDown**. This expresses that the dependent (lower) resource can only be started when the supporting (upper) resource is unavailable (down).

In example 2, the request must be transformed when propagated from B to C, because in order to start B and then A, C must be down. Therefore, SA OS/390 would put a *stop* request to C in this case, and the desired state of C would be set to UNAVAILABLE.

By propagating requests, SA OS/390 actively supports the start or stop request. You can also switch off request propagation for a resource. If this were to be done for resource A in example 1, then A would not be started because B is not available, and SA OS/390 would do nothing to start B. In this case A would only be started after B had been started, directly or indirectly, through another request.

Persistency of Requests and Conflicting Requests

Requests (and the votes derived from them) are persistent. They are stored in SA OS/390 and continue to be taken into account until you explicitly remove them. This implies that there can be more than one request (vote) for the same resource at the same time, and these requests (votes) can be contradictory, as shown in the following example.

Example 3: Expand example 1 by a resource D, also depending on C, which can only be started if C is down. A, B, and C are as in Figure 1 on page 5; D is supposed to be down, and its desired state to be UNAVAILABLE.

Figure 2 contains a graphical presentation of example 3.

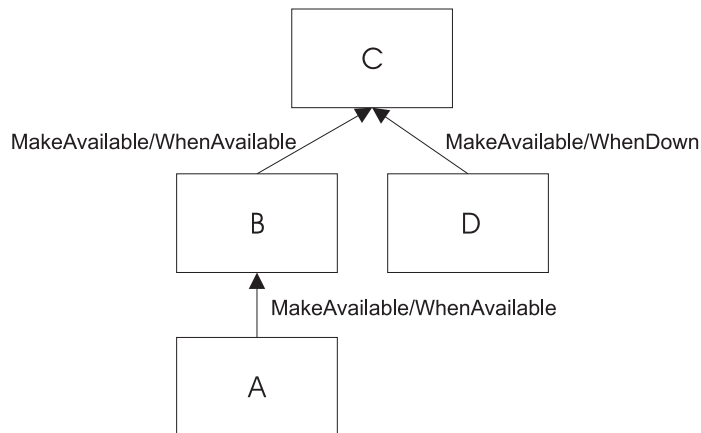


Figure 2. Example of Conflicting Requests

Now assume that first a request to start A and then a request to start D are passed to SA OS/390. The first request results in setting the desired state of C to AVAILABLE. Thereafter the propagation of the start request for D results in a vote to stop C. Since votes are persistent, the previous vote to start C is still existent,

and we have two contradictory votes for C. In such a situation, SA OS/390 uses the *priority* of the original requests to decide which one of the two wins.

When the priority of the old start vote for A is higher than that of the new vote to start D, then the desired state of D will be changed to AVAILABLE, but that of C will remain AVAILABLE; accordingly, SA OS/390 will not try to stop C, and thus D cannot be started. If, on the other hand, the vote to stop C has the higher priority, then the desired state of C is changed to UNAVAILABLE, and SA OS/390 will try to stop C in accordance with its desired state, and then to start D. When two contradictory votes have the same priority, a start vote wins over a stop vote.

The persistency concept implies that the losing vote is not automatically discarded. If, for instance, the start request for A wins, the start request for D and the propagated stop vote for C continue to be stored in SA OS/390, and can still be fulfilled after the request for A, and therefore also the start vote for C which was derived from it, have been removed by an operator. After the removal, SA OS/390 will determine the desired state of C again and will set it to UNAVAILABLE in response to the stop vote propagated from the start request for D, if no other vote is pending for C. After that, C will be stopped, and then D will be started.

Note that persistency of requests does not apply to successive requests of the same operator. In this case the second request will replace the earlier one.

Triggers

Triggers specify necessary conditions for starting or stopping an application. They are defined independently of applications. In this way the same trigger can be associated with more than one application. Triggers are defined and linked to an application in the policy database.

The conditions contained in a trigger are either startup conditions or shutdown conditions; there can be more than one startup condition, and also more than one shutdown condition. When a trigger is associated with an application, the resources generated from this application can only be started if *at least one* of the startup conditions in this trigger is satisfied; analogously, they can only be stopped if at least one of the shutdown conditions is fulfilled.

A trigger condition consists of a set of *events*. An SA OS/390 event represents an external event that is relevant to the state of the application associated with the trigger. The information that the external event has or has not occurred is passed to SA OS/390 by *setting* or *unsetting* the SA OS/390 event; this must be done by an operator or by an automation procedure. A trigger condition is only satisfied when *all* its events are set.

The following example illustrates the use of triggers and their interrelations with dependencies and request propagation.

Example 4: Expand example 1 to the effect that resource C is associated with a trigger that contains only one startup condition. This condition consists of two events, EVENT1 and EVENT2. EVENT1 is set, EVENT2 is unset.

When the request to start A arrives at SA OS/390, it will set off the same sequence of events as with example 1 up to step 2 on page 5. Since, however, the only startup condition of the trigger is not satisfied, C will not be started, and therefore B and A will not be started either. In order to start A, EVENT2 must be set, for

example, by an operator. This will lead to a re-evaluation of the startup condition. Since this condition is now satisfied, SA OS/390 will start C, and subsequently B and A.

Service Periods

So far we have always assumed that the start or stop requests are made by a human operator. However, SA OS/390 also provides the possibility to make start and stop requests at specified points in time independently of human intervention. The objects that are able to do this are called *service periods*. Service periods are defined in the policy database.

A service period is a set of time intervals, so-called *service windows*, during which an application should be available or unavailable. Service periods are defined independently of applications and can then be associated with one or more applications or application groups (see “Application Groups”). When an application is associated with a service period, the service period makes a start request for the application whenever the start time of a service window arrives; this request is canceled when the stop time of the service window arrives. You can also specify service windows during which the application should be unavailable; in this case, a stop request is made at the start, and canceled at the stop time of the service window. The following example is again an expansion of example 1.

Example 5: Resource A of example 1 is associated with a service period that contains at least one service window during which A should be available.

If the start time of this service window arrives, the same sequence of events will occur as with example 1.

An operator can temporarily modify a service period (this is called a *schedule override*). In case of a conflict between a request made by an operator and a request from a service period, the operator request wins when its priority is not lower than that of the service period request.

Application Groups

Modern applications often consist of more than one component, and these different components can be distributed among different systems. SA OS/390 provides the possibility to combine different components of an application on one or more systems within a sysplex into an *application group*. This allows you to start and stop a complex application by a single command, and to integrate it into automation processes as a whole.

Example 6: Suppose that resource B of example 1 is an application *group* with the members B1 and B2, and declare A dependent on group B (not on the individual group members), and B dependent on C. You can define B so that every request made to the group as a whole is automatically propagated to every group member.

Figure 3 on page 9 contains a graphical presentation of example 6.

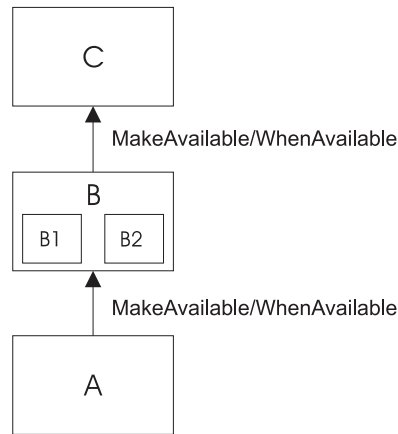


Figure 3. Example of a Request Involving a Group

Then, if you request A to be started, SA OS/390 will first, as before, propagate the request to group B and to application C. After C has been started and therefore group B can be started (step 3b on page 5 of example 1), a start vote will be propagated to every member of B. After the desired state of B1 and B2 has been set to AVAILABLE and both resources have been started, B will be considered available, and only then will SA OS/390 start A.

In this type of group (which is called BASIC) the group members form a complex entity, and therefore the group is only considered available when *all* its members are available.

The group concept is also used to move applications from their primary system to a backup system when the primary system has failed (group type MOVE). In this case the members of the group are instances of the same application on different systems. In accordance with their purpose, MOVE groups are declared available when *exactly one* of their members is available. You assign preferences to the elements in order to determine which group member is to be started when a start request is put to the group, and which group member takes over when the currently available member is no longer restartable any more.

SERVER groups are a third type of group. They are a variant of move groups and differ from these mainly in that you can specify how many of its members must be available before the group is considered available. As with move groups, you assign preferences to the members to determine which of them are to be started when a start request is put to the group, and which group members takes over when one of the currently available members is no longer restartable.

Groups can be nested. Suppose, for example, that you have a complex application that you want to be able to move from one system to another. Here you can first define two basic groups G1 and G2, each containing the application on a different system, and then define a move group that contains G1 and G2 as its members.

SA OS/390 and the NetView Message Automation Table

The implementation of SA OS/390 is based on NetView. One important area, where SA OS/390 relies on NetView functionality, is the NetView Message Automation Table (MAT). This table serves to automate operator responses to messages that are sent to NetView. It contains instructions of the general form:

When message ABC arrives then issue command XYZ.

Whenever NetView receives a message, it scans the MAT. If it finds an entry for the message, it issues the command specified in that entry.

With applications controlled by SA OS/390, the command will typically be one of the generic routines that are shipped with SA OS/390 (see *System Automation for OS/390 Programmer's Reference*). Many of these routines retrieve information from the ACF and then act according to that information.

A typical example for such information is the MESSAGES/USER DATA policy item of the APPLICATION policy object. Within the MESSAGES/USER DATA policy item, you can associate a command with a message ID (see *System Automation for OS/390 Defining Automation Policy*). If you connect this message ID with the generic routine ISSUECMD in the MAT, then NetView will execute ISSUECMD when the application sends the message in question to NetView. ISSUECMD, in its turn, will search for the message ID in the ACF entry for this application, and if the message ID is associated there with a command, it will issue this command. For more information on ISSUECMD, see *System Automation for OS/390 Programmer's Reference*.

For example, you could associate the message ID AHL031I, which is the ID of the startup message sent by the application GTF, with the command MVS \$DMRO'GTF IS NOW UP' in the MESSAGES/USER DATA policy item for GTF. Then the MAT would have to contain an entry like the following:

```
IF MSGID = 'AHL031I'  
THEN EXEC(CMD('ISSUECMD AUTOTYP=START') ROUTE(ONE *));
```

Now, when NetView receives the AHL031I message it extracts the job name from the message and calls ISSUECMD. ISSUECMD knows where to find the job name and searches the ACF for the associated application. When it finds GTF, it will look for the AHL031I entry in the MESSAGES/USER DATA policy item and will issue the command that is associated with AHL031I for GTF,
MVS \$DMRO'GTF IS NOW UP'.

For more information on the MAT, see *Tivoli NetView for OS/390 Automation Guide*. CICS Automation also has some special generic routines, see "Chapter 6. CICS Automation Routines, Commands, and Definition Members" on page 65.

Chapter 2. Special Functions of CICS Automation

CICS Automation is integrated into SA OS/390. Thus, CICS regions must be generated in the policy database as subsystems, by linking CICS applications to systems, in order to be available to CICS Automation. Triggers and service periods for CICS regions are also defined as for any other application. CICS Automation also offers some special facilities.

Link Monitoring

CICS communicates with other systems or subsystems using either intersystem communication (ISC) or interregion communication (IRC). Link monitoring verifies that the IRCs and ISCs are active. This is done by issuing CEMT INQUIRE at certain intervals to check the status of these connections. When a link failure is detected, link monitoring will perform automatic recovery actions.

Basic link monitoring only ensures that the link is active on the origin subsystem. With the so-called echo facility, you can also control the remote subsystem at the other end of the link. This facility is supported for links to CICS and IMS.

Health Checking

Through health checking, you execute programs that check the state of certain components of an application running under CICS. CICS Automation initiates a health check program by sending a request across the program-to-program interface to CICS. CICS invokes the program, and sends back an Acknowledgement (ACK), indicating that the program executed as expected; or it sends a Negative Acknowledgement (NACK), indicating that the program did not execute as expected. A NACK includes data describing the error. Up to ten health check programs can be associated with any CICS application that is defined to SA OS/390.

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

State/Action Tables

The MAT (see “SA OS/390 and the NetView Message Automation Table” on page 9) contains instructions on what to do when certain messages are sent to NetView. Here the action to be taken only depends on one event (the message).

There are, however, cases where not only the actual event, but also previous events must be taken into consideration in order to determine the necessary action. Assume, for example, that a CICS subsystem reports that its storage supply is back to normal. If the preceding message had reported a storage shortage and this had triggered certain actions, these actions will have to be revoked; if, however, no error message had occurred before, the actual message should simply be ignored.

To meet this requirement, resources are associated with a *state*. When an event, usually a message, occurs that is relevant for the resource in question, the action to be taken is inferred not only from the event, but also from the actual state of the resource; that is, the same event can lead to different actions, depending on the

state. Furthermore, the state of the resource will usually be changed on account of the event and the previous state. In this way the current state can be said to reflect the history of the resource.

State/Action Tables serve to define such a state-dependent event processing. They form a two-dimensional matrix, where the rows correspond to the possible events (messages) and the columns to the possible states. The individual cells specify the action to take and the new state in the case that the respective event occurs when the resource is in the respective state.

For an example, consider the following table:

```

/*****
/* STATES:          NORMAL          SINGLE          DOUBLE          */
/*                HALTED          HALTED          */
/* STATE VALUES:  0              1              2              */
/*****
EVENT=DFH0506     EVEES101/1      NA              NA
EVENT=DFH0507     NOP            EVEES102/0     NA
EVENT=DFHSM0100I  EVEES101/1     /2             NA
EVENT=DFHSM0101I  EVEES101/1     /2             NA
EVENT=DFHSM0106I  NOP            EVEES102/0     /1
EVENT=DFHSM0107I  NOP            EVEES102/0     /1
EVENT=CICSDN      NOP            EVEES103/0     EVEES103/0
EVENT=CICSINIT    NOP            EVEES103/0     EVEES103/0

```

Figure 4. Short-on-Storage State/Action Table Example

The table must be read as follows:

- If an action must be taken and/or the status be changed, the cells contain the name of the action command list to be called and/or the code of the new state, separated by a slash.
- NOP signifies that nothing is to be done.
- NA signifies that this event/state combination cannot occur.

The sample table serves to manage the storage supply resource for CICS applications. There are three possible states for storage supply, 0=NORMAL, 1=SINGLE HALTED, and 3=DOUBLE HALTED. The following scenario illustrates how the table works:

1. Assume that the actual state of storage supply is 0.
2. Message DFH0506 arrives. This message says that CICS is short on storage.
3. Row 1, column 1 of the table is consulted.
4. The state of storage supply changes to 1.
5. The action command list EVEES101 is called. EVEES101 schedules a timer, on the expiration of which the operator is alerted, the SDF is updated, and an MVS dump taken.
6. Message DFH0507 arrives. This message says that storage supply is back to normal.
7. Row 2, column 2 of the table is consulted.
8. The state of storage supply changes to 0.
9. The action command list EVEES102 is called. EVEES102 cancels the timer and removes any SDF messages that are associated with this error occurrence.

Recovery

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

Program-to-Program Interface

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

- Initiate, from NetView, the execution of a CICS program.
- Process a response from this CICS program.
- Initiate, from CICS, 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 3. Customizing CICS Automation" on page 17 provides you with step-by-step procedures that tell you how to install these components so that the interface can be implemented.

You do not need to understand our implementation of the NetView program-to-program interface to use the CICS Automation-provided functions. However, if you are considering using the CICS Automation program-to-program interface code for your own purposes, read "Appendix. CICS Automation and the Program-to-Program Interface" on page 137.

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 96. The command processors are described in "EVESNCCI—NetView to CICS Communication Interface" on page 146, and "EVESNRSP - Common Response Handler from CICS" on page 151.

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; also there is a subroutine program which is described in "EVESCCCI - CICS to NetView Communication Interface" on page 152, and an initialization member which is described in "EVESPINM—CICS PPI Initialization Member" on page 94.

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 actual VTAM applid. The steps contained in the customization sections describe how to define the RECEIVERID and the VTAM applid.

Part 2. Customizing CICS Automation

This part describes the steps that are necessary to customize and set up CICS Automation. Furthermore, it contains reference sections for CICS specific MESSAGES keywords and for common routines which request information or perform tasks associated with CICS Automation.

Important: CICS Automation 2.1 only supports CICS V4 and higher.

Chapter 3. Customizing CICS Automation

This chapter explains how to customize NetView, CICS, and SA OS/390 for CICS Automation. The customization process mainly comprises the following steps:

1. Define CICS Automation to CICS.
2. CICS Automation Definitions in NetView.

This step mainly consists of defining the policy objects in the SA OS/390 policy database that are necessary for CICS Automation.

Installation-Related Steps

Step 1: Update MPFLSTxx in SYS1.PARMLIB

Add the following entries in MPFLSTxx in SYS1.PARMLIB to trap all DFH , EVE, and EYU prefix messages:

```
DFH*,SUP(NO),AUTO(YES)
EVE*,SUP(NO),AUTO(YES)
EYU*,SUP(NO),AUTO(YES)
```

This CICS Automation requirement forwards these messages to NetView.

Tailor Your Network NetView

Step 1: Tailor Your Network NetView for CDRM Recovery

If you have a separate network NetView, then add the following to the network MAT:

```
IF MSGID='IST093I' & TOKEN(2)=RESOURCE .
  THEN EXEC(CMD('EVEEZ000 ' RESOURCE) ROUTE(ALL *));
```

Warning

The network NetView message automation table is used if two copies of NetView are running in the same system, one for SA OS/390 and one for network automation.

Define CICS Automation to CICS

This section describes the basic CICS Automation definitions that take place on CICS. Because of the differences in CICS versions and releases, we attempt to avoid providing installation or definition instructions specific to CICS. Therefore, you should refer to the CICS documentation while performing these steps, especially the following:

- Either the *CICS/MVS[®] Resource Definition (Macro)* or the *CICS/ESA[®] Resource Definition (Macro)* publication.
- Either the *CICS/MVS Resource Definition (Online)* or the *CICS/ESA Resource Definition (Online)* publication.

These steps are performed on each CICS region.

Note: The TS queues EVEVCQUE and COLEVEEQ used by SA OS/390 CICS must not be defined as remote in your TST (temporary storage table).

Step 1: SIT or Startup Overrides

1. On each CICS, ensure that the system initialization table (SIT) or startup overrides include the following:

```
PLTPI=xx,   where xx is the suffix to the startup PLT
PLTSD=yy,   where yy is the suffix to the shutdown PLT
MSGVL=1,
BMS=(STANDARD|FULL)
```
2. Because CICS Automation maintains a long-running task in each CICS, review the AMXT, CMXT, and MXT values.
3. Possibly, you must add CN as your last startup override, whether from SYSIN or through the JCL, in order to support the CICS Automation operator interface SIT override option. However, this is not necessary if you have coded the start commands in the STARTUP item of the APPLICATION policy object in the following style:

```
MVS S EYUMAS,JOBNAME=EYUMAS1A,CICS=EYUMAS1A,PARM='SYSIN,START=AUTO&EHKVAR1'
```

This is also the way the start commands are predefined in the sample databases.

Step 2: Program List Table Definitions

1. Add the TYPE=ENTRY definitions shown in the following example to the post initialization program list table (PLT) for each CICS after the entry for DFHDELIM (as in phase 2).

```
DFHPLT TYPE=INITIAL,SUFFIX=xx
DFHPLT TYPE=ENTRY,PROGRAM=EVESTIEX
DFHPLT TYPE=ENTRY,PROGRAM=EVESTISP
DFHPLT TYPE=FINAL
```

2. Add the TYPE=ENTRY definitions shown in the following example to the shutdown program list table (PLT) for each CICS.

```
DFHPLT TYPE=INITIAL,SUFFIX=yy
DFHPLT TYPE=ENTRY,PROGRAM=EVEPLTT
DFHPLT TYPE=ENTRY,PROGRAM=EVESYLMQ
DFHPLT TYPE=ENTRY,PROGRAM=DFHDELIM
DFHPLT TYPE=FINAL
```

3. Assemble the PLT table.
4. If you must define a new table, add the following definitions:

```
DEFINE PROGRAM(DFHPLTxx) LANGUAGE(ASSEMBLER) GROUP(yourgroup)
DEFINE PROGRAM(DFHPLTyy) LANGUAGE(ASSEMBLER) GROUP(yourgroup)
```

Step 3: Define Consoles

1. Define a console with either a CONSOLE number of 00 or a CONSNAME of INTERNAL (the CONSOLE and CONSNAME parameters are mutually exclusive). This enables transactions COPS (start program-to-program interface) and COPP (stop program-to-program interface). Verify that the SAF security profile associated with console 00 has sufficient authority to issue the COPP and COPS transactions.
2. Define consoles for autotasks to enable shutdown and broadcast functions.
In an EMCS environment the autotask console names are determined, in order of precedence as follows:

- a. If you are using AOCGETCN (that is, using the profiles shipped with the product) the name is determined by AOFNPMASK. For more information, see *System Automation for OS/390 Customizing and Programming* or *System Automation for OS/390 Defining Automation Policy*.
- b. If you are using NetView 3.1 or above, the CONSNMAME parameter on the PROFILE statement in the task profile determines the EMCS console name. For more information, see *NetView for MVS V3 Administration and Security Reference*.
- c. By default the autotask name is used for the EMCS console name.

The SHUTOPER automated function of the base determines the autotask name used for shutdown; the autotasks defined for the CICS automated functions (see *System Automation for OS/390 Defining Automation Policy*) are used for broadcast.

In addition, a console has to be defined for each NetView operator that may wish to inquire or control a CICS region. For CICS Transaction Server V1.3 and later releases this can be accomplished via the CICS Console Auto-Install function. Further information on this function can be found in the *CICS Installation/Customization* manuals. For pre-Transaction Server V1.3 CICS systems, the following two PTFs provide a mechanism to define a pool of consoles: PQ09813 for CICS TS V1R1 and V1R2 and PQ09812 for CICS V4.

Step 4: Transaction and Program Definitions

This step describes how to define the standard CICS Automation transactions and programs to CICS. To this purpose, the DFHCSDUP program is used.

The members required to run these jobs are provided with CICS Automation. However, some modifications are required, as described below:

A helpful hint

You might want to backup your CSDs before doing this step.

For each CSD, run the EVESJ015 sample job. This job defines transactions and programs for CICS Automation in three groups—EVEGRP1, EVEGRP2, and EVEGRP3. Before you run it, modify the job as follows:

```
//STEPLIB DD DISP=SHR,DSN=CICS xxy.LOADLIB           (See note 1.)
//        DD DISP=SHR,DSN=ING.SINGMODC             (See note 2.)
//DFHCSD  DD DISP=SHR,DSN=yourCSD.DFHCSD          (See note 3.)
//SYSUT1  DD UNIT=SYSALLDA,SPACE=(1024,(100,100))
//SYSPRINT DD SYSOUT=*
//SYSIN   DD *
          UPGRADE USING(EVECSDDxx)                 (See note 4.)
          ADD GROUP(EVEGRP1) LIST(yourlist)        (See note 5.)
          ADD GROUP(EVEGRP2) LIST(yourlist)
          ADD GROUP(EVEGRP3) LIST(yourlist)
/*
//
```

Notes:

1. Change the data set name of this file to match your CICS load library, where xx is 41, 51, 52, or 53, depending on your level of CICS. 51 refers to CICS TS for SA OS/390 R1, 52 refers to CICS TS for SA OS/390 R2, and 53 refers to CICS TS for SA OS/390 R3.

2. If required, change the data set name on this file to match your CICS Automation load library.
3. Change the data set name to match your CICS CSD for this CICS.
4. Change *xx* to 41, 51, 52, or 53, depending on your CICS level. 51 refers to CICS TS for SA OS/390 R1, 52 refers to CICS TS for SA OS/390 R2, and 53 refers to CICS TS for SA OS/390 R3.
5. Change *yourlist* for EVEGRP1, EVEGRP2, and EVEGRP3 to the name of the group lists for this CICS.

After running this job, you will need to INSTALL these three groups on each CICS.

Step 5: DFHRPL and the CICS Automation Library

Add the ING.SINGMODC data set library to the DFHRPL concatenation.

CICS Automation Definitions in NetView

This section describes the definitions that take place in NetView. For the necessary definitions in the policy database, a number of sample APPLICATION classes and instances for CICS have been supplied with the sample policy databases for SA OS/390. Use these samples as a guideline to assist you with defining your CICS Automation policies. The names of the classes have the format:

CLASS_CICS_XXXX,

the name format of the instances is:

CICSXXXX
CMASXXXX

XXXX stands for a version suffix, for example, TS11.

Step 1: Basic CICS Automation Common Policy Definitions

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

1. Verify that the system environment has been defined as described in the SA OS/390 base documentation.

An important note!

You should already have a working and tested SA OS/390 before beginning the customization of CICS Automation.

2. Ensure that the Automation Operators that will be used for CICS have been defined under the AOP entry type. The following entries are necessary:

Automated Function	Operator ID	Message Classes
CICSMSTR	AUTCICS	EVE*,DFH*,AND*
CICSCPPI	AUTCPPI	

3. Define sets of State/Action Tables under the CSA entry type.
4. Define any service periods, events and triggers under the SVP, EVT, and TRG entry types.
5. Define any link monitoring connections by first creating links (under the CCN entry type) and link monitoring service periods (under the CVP entry type),

and then associating service periods to connections under the MONITORING PERIOD policy item of the CCN entry type.

This step has now been completed.

Step 2: Basic CICS Application Definitions

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

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

Important!

Note that the subsystem name as specified in the **Subsystem Name** field of the **Define New Entry** panel must not exceed *eight* characters for applications of type CICS. This is in contrast with standard applications where 11 characters are allowed.

2. Examine the CICS CONTROL policy item and enter any required values. You *must* specify values for the **APPLid** and **Major node** fields.
3. Specify the automation flags under the AUTOMATION FLAGS policy item.
4. Specify the thresholds of the application under the THRESHOLDS policy item.
5. Specify the startup commands under the STARTUP policy item. Note that the possible startup types depend on the CICS version as follows:

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

6. Specify the shutdown commands under the SHUTDOWN policy item.
7. Link the application to the required CICS STATE ACTION policy object (CSA) under the STATE ACTION policy item.
8. If you will use service periods or triggers, link these to the application under the SERVICE PERIOD or TRIGGER policy item.
9. If you will use link monitoring, link the application to a CICS connection (CCN entry type) under the CICS CONNECTION policy item.
10. Specify the ACORESTART keyword under the MESSAGES/USER DATA item of the APPLICATION policy object. The command associated with this keyword must be CICSRSYC, see “CICSRSYC—CICS Resync” on page 76.
11. If you will use the health check function, specify the HEALTHCHK keyword under the MESSAGES/USER DATA item. For details, see “HEALTHCHK—Health Checking” on page 56.

12. Review the supplied CICS sample classes and instances for the the message keywords described in “CICS-Specific MESSAGES/USER DATA Keywords” on page 50. Customize these entries as required.
13. If you need thresholds for the RCVRAUTO, RCVRSOS, RCVRTRAN (RCVRTRAN.*tranid*), or RCVRVIOL message keywords (see “CICS-Specific MESSAGES/USER DATA Keywords” on page 50 for these keywords), define resource thresholds for them with the names of AUTO, SOS, TRAN (TRAN.*tranid*), or VIOL, respectively, under the CICS-specific RESOURCE THRESHOLDS policy item. For more details on recovery, see “Automating Recovery for Transactions and Problem Areas” on page 30.

This step has now been completed.

Step 3: The Program-to-Program Interface Initialization Member (EVENTASK)

This member only needs to be changed under the following circumstances:

1. The BUFFQL is set to 20. If you get error messages indicating that 20 is not enough, increase this number. Refer to the *buffer queue limit* description in the *NetView Application Programming: Program-to-Program Interface* documentation for specific error messages and buffer queue guidelines.

The following is the default entry:

BUFFQL=20

2. If you are running CICS Automation in more than one 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. The following is the default entry:

RECEIVERID=NETCPPI

3. If you are using something other than the default program-to-program interface automation operator ID, change that ID in the SERVER entries. The default is AUTCPPI, as shown in the following:

**SERVER=REQUEST, LMT, AUTCPPI, EVEEYPPS
 SERVER=RESPONSE, CEMT, AUTCPPI, EVESNRSP
 SERVER=RESPONSE, LMT, AUTCPPI, EVESNRSP
 SERVER=REQUEST, NACK, AUTCPPI, EVESNACK
 SERVER=RESPONSE, NACK, AUTCPPI, EVESNACK**

Step 4: Adding a New CICS Automation Status File Record

If you define a new CICS region to CICS Automation and attempt to start using the operator interface, the startup will fail because there will be no status record for CICS Automation to read. To overcome this problem, you must build a status file record for each new CICS region, or you must trap a message for that region prior to attempting the manual startup. Any message trapped by CICS Automation for a CICS subsystem (such as IEF403I or DFH1500), will result in a status file record being built for that subsystem if one does not already exist. CICS Automation will dynamically build a status file record for each CICS subsystem upon trapping the first message for it.

To create a new status file record, enter:

EVEEMIGR NAME=cics_subsystem_name NEWVERSION=cics_version

For more information on EVEEMIGR, see “EVEEMIGR—Migrate Subsystem to CICS/TS” on page 81.

Step 5: Installing CICSplex SM REXX API

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

Alternatively, module IRXFLOC can be customized according to the instructions in the section "Installing the REXX function package" in manual *CICSplex SM Setup*.

Extended CICS Definitions

This chapter describes the CICS Automation definitions that take place in CICS. These steps must be performed on each CICS region.

Step 1: Health Check Programs

Note: This step assumes that the health check routines are already written and that you know the transaction name, program name, and language of each. If you need more information, read "How to Set Up Health Checking" on page 36 before proceeding with this step.

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

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

Note: The program name must correspond to a program name defined in the **Data** field under the HEALTHCHK keyword in the MESSAGES/USER DATA policy item message ID for the respective subsystem. For more information on HEALTHCHK, see "HEALTHCHK—Health Checking" on page 56.

Step 2: Program-to-Program Interface Initialization Member (CICS)

Note: This step is only required if the initialization member requires modification. Modification is needed when:

- The RECEIVERID is changed. This value must be the same as the value defined in "Step 3: The Program-to-Program Interface Initialization Member (EVENTASK)" on page 22.
- You are using the program-to-program interface for your own transactions.

The purpose of this initialization member is to relate program-to-program interface function names to CICS transaction names. The member looks something like this:

```
EVEMPINM TYPE=INITIAL,      INITIAL ENTRY
      BUFFQL=4,              BUFFER QUEUE LIMIT
      RECEIVERID=NETVCPPI,   NPDS RECEIVER IDENTIFICATION
      USERID=YES             USE CICS 4.1 SECURITY

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

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

```

EVEMPINM TYPE=ENTRY,      DEFINE A FUNCTION
      FUNCTION=HEALTH,    FUNCTION NAME
      TRANSID=COHR        TRANSACTION NAME

EVEMPINM TYPE=FINAL      REQUIRED END

```

If this member requires changes, do the following:

1. Edit EVESPINM. USERID=YES means that CICS security checking is required, USERID=NO means that CICS security checking is not required.
2. Assemble the program-to-program interface initialization member with the EVESJ020 sample job.

Note: This JCL is used for assembling EVESCMT3 as well. Edit the JCL and make sure that the member name to be assembled is EVEMPINM.

3. Place the assembled member into one of the libraries in the CICSDFHRPL chain.

Step 3: Add or Change the CICS Transient Data Messages

If you want to add or change the CICS transient data messages:

1. Edit EVESCMT3 and make the required changes.
2. Assemble the table with the EVESJ020 sample job.

Note: This JCL is used for assembling EVEMPINM as well. Edit the JCL and make sure that the member name to be assembled is EVESCMT3.

3. Place the assembled member into one of the libraries in the CICSDFHRPL chain.

Step 4: Echoplex Back-End Programs

Echoplexing can be implemented for target subsystems of subtype IMS or CICS. For IMS target subsystems, no installation is required.

For CICS target subsystems, install the back-end echoplex program to the remote systems as follows:

```

DEFINE TRANSACTION(ECHO) PROGRAM(EVESYCB7)
DEFINE PROGRAM(EVESYCB7) LANGUAGE(ASSEMBLER)

```

Note: The transaction name ECHO may be changed for your installation. Whatever name is used, it must be specified in the **Echo** field of the CICS LINK policy object (CCN entry type) that is associated with the respective subsystem in the policy database. See *System Automation for OS/390 Defining Automation Policy*.

Step 5: Security Considerations

If you wish to use non-terminal transaction security, refer to the V4R1 *CICS Release Guide* for CICS-specific information about non-terminal transaction security and the security manager domain.

1. Define all NetView operators which will invoke CICS functions defined to RACF® (or your SAF-compliant security system). This will include:
 - Regular NetView operators

NetView autotasks which perform CICS-related actions. These autotasks will include those autotasks specifically defined for CICS Automation use, and may include the autotasks which process shutdown functions or resynchronization functions.

2. Define SAF surrogate authorization for CICS. Since CICS Automation is started from the PLTPI, surrogate authorization for the PLTPIUSR is recommended for all CICS Automation NetView users. Define surrogate profile types of DFHINSTL for PLTPI processing and DFHSTART for normal-started task processing.
3. Define TCICSTRN profiles for all the transactions which will use non-terminal transaction security. See “Step 2: Program List Table Definitions” on page 18 (in Define CICS Automation to CICS) for the list of supplied CICS Automation transactions.
4. 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).
5. Enable non-terminal transaction security in CICS by modifying the CICS SIT to specify XTRAN=YES and XUSER=YES.
6. Modify PLTPIUSR and PLTPISEC as required.

If you do not plan on exploiting the non-terminal transaction security in CICS, disable this function in CICS Automation by modifying the EVESPINM member and specifying USERID=N0 to disable extended support. Reassemble the member using sample job EVESJ020.

CICS Automation Definitions for CICSplex System Manager (CPSM)

This section is a guide to help you define CICSplex System Manager components to Automation.

Automating Coordinating Address Space (CAS) Startup and Shutdown

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

To help with the definition of the CAS, CICS Automation supplies NetView message automation table entries to let Automation know when the CAS is UP. These definitions are stored in member EVEMEYU1 in the ING.SINGNPRM.

To assist in the task of building the policy, a sample subsystem definition for a CAS has been provided. The name of the sample CAS region is EYUCAS1A.

Automating CICSplex SM Address Space (CMAS) Startup and Shutdown

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

The Automation required to manage the CMAS regions is less than a normal CICS region because there is no need to have PPI and other CICS-monitoring functions. However, CMAS CICS regions need to be shutdown via the CICSplex SM

SHUTDOWN command. To allow CICS Automation to shutdown the CMAS, a new command, CMASSHUT, has been created. Use it in place of the normal CICSSHUT command in the policy definition for CMAS regions.

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

CPSM recommends that CMAS regions be started prior to MAS regions they manage. This can be achieved via relationships as supported by System Automation for OS/390.

Migration and Coexistence

This section contains information on migrating from CICS Automation Version 1 Release 4 to CICS Automation Version 2 Release 1, and on compatibility between Version 1 Release 4 and Version 2 Release 1.

Migration

The general migration process is described in *System Automation for OS/390 Defining Automation Policy*.

The following table provides every ACF keyword of Version 1 Release 3 that is CICS specific, or has a CICS-specific application to the corresponding policy item of Version 2 Release 1. Comments are added as required.

Table 3. CICS-Specific Correspondences between ACF Keywords and Policy Objects/Items

ACF keyword	Policy object/item	Comments
ABCODESYSTEM	Keyword for MESSAGES/USER DATA policy item of APPLICATION objects	
ABCODETRAN	Keyword for MESSAGES/USER DATA policy item of APPLICATION objects	
AREA	STATE/ACTIONTABLES object (CSA entry type)	These objects must be linked to applications through the STATE ACTION TABLE policy item of CICS applications.
AUTOOPS	AUTO OPERATORS policy object (AOP entry type)	The CICSOP nn auto operators are obsolete.
CICSCNTL	CICS CONTROL policy item for APPLICATION objects of type CICS	
CONNECTION	CICS CONNECTION policy item for APPLICATION objects of type CICS	
CICSGROUP	—	Obsolete. Use standard application groups.
EXTCOND	—	Obsolete. Use standard triggers.
ENVIRON SETUP	—	The entry and the EVEEIIINT exit are obsolete.

Table 3. CICS-Specific Correspondences between ACF Keywords and Policy Objects/Items (continued)

ACF keyword	Policy object/item	Comments
ENVIRON TIMEOUT (CEMT keyword)	TIMEOUT SETTINGS object (TMO entry type)	This policy object must be linked to SYSTEM objects.
HEALTHCHK	Keyword for MESSAGES/USER DATA policy item of APPLICATION objects	
LISTSHUT	Keyword for MESSAGES/USER DATA policy item of APPLICATION objects	
INITSTART	AUTOMATION FLAGS item of APPLICATION object	The EVEEIEXT exit is obsolete.
PRODUCT	STATE/ACTIONTABLES object (CSA entry type)	These objects must be linked to applications through the STATE ACTION TABLE policy item of CICS applications.
RCVRAUTO	Keyword for MESSAGES/USER DATA policy item of APPLICATION objects	
RCVRSOS	Keyword for MESSAGES/USER DATA policy item of APPLICATION objects	
RCVRTRAN	Keyword for MESSAGES/USER DATA policy item of APPLICATION objects	
RCVRVIOL	Keyword for MESSAGES/USER DATA policy item of APPLICATION objects	
RECOVERY	MINOR RESOURCE FLAGS item of the APPLICATION object	For the names of the minor resources, see “Automating Recovery for Transactions and Problem Areas” on page 30.
RESTART	AUTOMATION FLAGS item of APPLICATION object	The EVEEIEXT exit is obsolete.
SERVICE	—	Obsolete. Use standard service periods.
THRESHOLDS (CICS specific thresholds in minor resource format)	CICS specific RESOURCE THRESHOLDS policy item for application of type CICS	For the minor resource names, see “Automating Recovery for Transactions and Problem Areas” on page 30.
TRIGGER	—	Obsolete. Use standard triggers.

Note that additional customization is necessary for the migrated policy database, as there are policy items that are required in CICS Automation V2R1 but have no counterpart in CICS Automation V1R4.

In particular, you must perform the following step:

Define the startup commands for the individual CICS startup types in the STARTUP policy item of the CICS applications. For the valid startup types, see “Startups” on page 113; for specifying startup commands in the STARTUP policy item, see *System Automation for OS/390 Defining Automation Policy*.

The CICS-specific trace function no longer exists. CICS Automation V2R1 uses the trace function of SA OS/390. Note, however, that most components of CICS Automation only support the global debug setting. For details on the trace function of SA OS/390, see the description of AOCTRACE in the *System Automation for OS/390 Operator’s Commands*.

The **Set Cold Start Indicator** panel is no longer available. Instead, the update panel of the SA OS/390 INGLIST command will be displayed when you select this option on the **Support Functions** panel. For details, see “Cold Start Indicator” on page 127.

Coexistence between V1R4 and V2R1

Generally, you can control CICS subsystems running on a CICS Automation V1R4 target system from CICS Automation V2R1 (downward compatibility), and conversely CICS subsystems running on a CICS Automation V2R1 target system from CICS Automation V1R4 (upward compatibility). There is, however, a restriction to this:

- Service periods and triggers.

You can access the triggers and service periods of a CICS subsystem running on a CICS Automation V1R4 target system from CICS Automation V2R1; in this case, the CICS-specific panels of CICS Automation V1R4 will be displayed. However, you cannot access a CICS subsystem running on a CICS Automation V2R1 target system from the **Service Periods Functions** or the **Triggers List** panels of CICS Automation V1R4. Thus service periods and triggers are only downward compatible.

Chapter 4. How to Set Up the Special Functions of CICS Automation

This chapter explains how to set up the special functions of CICS Automation for your specific needs. For the setup of base functions, like starting and stopping subsystems, see the SA OS/390 documentation.

Defining the SDF States for CICS Automation

The Status Display Facility uses color to represent the various subsystem resource statuses; such as error, warning, action, or informational states. Typically, a subsystem shown in green on a Status Display Facility status panel indicates that it is up, whereas red indicates a stopped or problem state. For more information, see *System Automation for OS/390 Programmer's Reference*.

Priority, highlight level, and color definitions of the states are defined in the customization dialogs under a STATUS DETAILS policy object (SCR entry type). For CICS Automation, the following states must be present:

CICSTRAN	CICS Transactions.
CICSHLTH	Health checking.
CICSLMT	Link monitoring.
CICSTIMR	CICS timers.
CICSVIOL	Storage violations (CICS storage).
CICSSOS	Short on storage conditions (CICS storage).
VTAMACB	VTAM ACB errors.
CRITMSG	The default critical messages definition (CICS Critical Message).
CRITMSGA	Messages ending in A (CICS Critical Message).
CRITMSGE	Messages ending in E (CICS Critical Message).
CRITMSGW	Messages ending in W (CICS Critical Message).
CRITMSGI	Messages ending in I (CICS Critical Message).

Each of these categories except the suffixed **CRITMSG_x** keywords corresponds to an item on the **CICS MONITOR PANEL** (see “Chapter 12. The Status Display Facility” on page 133). The color definitions indicate which color to use when a message is logged against a specific category. **CRITMSG_A** through **CRITMSG_I** are subcategories of **CRITMSG**. These subcategories are associated with different priorities, and the color of the **CICS Critical Messages** panel item, which corresponds to **CRITMSG**, is determined by the message that belongs to the subcategory with the highest priority.

The default specifications assign messages ending in I as having the lowest priority. Messages ending in W have the next highest, messages ending in E have the next highest, and messages ending in A have the highest priority. Thus, if a message ending in A is logged, the **CICS Critical Messages** item will turn to the color defined for those messages (probably red), overriding any other message color.

Automating Recovery for Transactions and Problem Areas

CICS Automation provides automated recovery for transactions and for the following problem areas:

- Short-on-storage conditions
- Storage violations
- Autoinstall problems
- VTAM ACB failures

You can control automated recovery for transactions and problem areas through the following three policy items of the APPLICATION object:

MINOR RESOURCE FLAGS

With these flags, you can switch automated recovery on and off for transactions or for a certain problem area. To do this, you must define a minor resource and set its **Recovery** flag as required; for the definition of minor resources, see *System Automation for OS/390 Defining Automation Policy*. The names of these minor resources must be as follows:

Table 4. Minor Resource Names for Problem Areas

Problem area	Minor resource name
Short-on-storage conditions	SOS
Storage violations	VIOL
Autoinstall problems	AUTO
VTAM ACB failures	VTAMACB
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. Thus you only need to define minor resources when the recovery setting for a lower level is to be different from the next higher level.

RESOURCE THRESHOLDS

With this CICS-specific policy item, you determine the threshold at which recovery should stop. This threshold is defined by the number of errors within a certain time interval. As with the recovery flags, you must associate the threshold definition with the transaction/problem area by giving it one of the names listed in Table 4; you can also specify thresholds for a single transaction.

MESSAGES/USER DATA

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

Problem area	Keywords
Short-on-storage conditions	RCVRSOS (see page 60)

Problem area	Keywords
Storage violations	RCVRVIOL (see page 63)
Autoinstall problems	RCVRAUTO (see page 59)
VTAM ACB failures	—
Transaction recovery	ABCODETRAN (see page 54), RCVRTRAN (see page 62)

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:

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

The recovery itself is typically triggered from the MAT 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

Suppose that recovery is enabled for the CICS1 application on the application level, and that you want it also enabled for transactions PAYR, DBTS, and BLNG, but not for any other transaction. Then you must define four minor resources for CICS1 in the customization dialogs as follows:

```

COMMANDS  HELP
-----
Minor Resource Selection                               Row 1 to 10 of 24
Command ==>                                           SCROLL==> PAGE

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

Major Resource: CICS1

Select minor resource option to be altered.

s  TRAN_____
_  TRAN.BLNG_____
_  TRAN.DBTS_____
_  TRAN.PAYR_____
_  _____
_  _____
_  _____
_  _____
_  _____
_  _____

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

```

Figure 5. Defining Minor Resources for Transactions

Set the recovery automation flag to NO for TRAN and to YES for the three second level minor resources. For example, to do this for TRAN, enter s in the **Select** column and press ENTER. The following panel is displayed:

```

COMMANDS  ACTIONS  HELP
-----
Flag Automation Specification
Command ==>

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

Resource: CICS1.TRAN
Enter level of automation desired.

Automation Flags: Y = Yes    N = No    E = Exits
Assist Flags:      D = Display L = Log    N = None

More:      +

Actions  Flag      Auto      Assist      Exits
Automation .
Recovery . . NO
Start. . . .
ShutDown . .
Initstart. .
Restart. . .

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

```

Figure 6. Automation Flag Panel

Here you specify which flags are set and which are not. For more information, see *System Automation for OS/390 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

under the CICS-specific RESOURCE THRESHOLDS 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

```

COMMANDS  HELP
-----
Command ==>                               Thresholds Definition           Policy saved

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

Resource:   CICS1.TRAN.PAYR
Description:

Specify the number of times an event must occur to define a particular level.

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

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

```

Figure 7. Thresholds Definitions Panel

For more details, see *System Automation for OS/390 Defining Automation Policy*.

Selecting the Abend Codes

The abend codes for which recovery is to take place are specified through the ABCODETRAN keyword in the MESSAGES/USER DATA policy item for CICS1. If you want to initiate recovery for transaction PAYR only when the abend code is AEI0 or AKC3, you must create the ABCODETRAN entry in the **Message Processing** panel and associate codes with this entry as displayed in Figure 8 on page 34:

```

COMMANDS  HELP
-----
                                Code Processing                      Row 1 to 6 of 21
Command ==>>>                                SCROLL==>> PAGE

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

Subsystem  : CICS1
Message ID : ABCODETRAN.PAYR

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

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

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

```

Figure 8. Code Processing Panel

For more details, see “ABCODETRAN—Transaction Abend Recovery” on page 54.

Specifying Recovery Actions

You specify the commands to be issued for recovery in the **CMD Processing** panel for the RCVRTRAN message keyword entry of CICS1. For example:

```

COMMANDS  HELP
-----
                                CMD Processing                      Row 1 to 2 of 20
Command ==>>>                                SCROLL==>> PAGE

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

Subsystem  : CICS1
Message ID : RCVRTRAN.PAYR

Enter commands to be executed when resource issues the selected message.

Pass/Selection Automated Function/'*'
Command Text
-----
MSG OP1, TRAN &EHKCFGV1 FAILED_____
-----

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

```

Figure 9. Command Processing Panel

For more details, see “RCVRTRAN—Transaction Recovery” on page 62.

How to Set Up the State/Action Tables

In CICS Automation, state/action tables are used for recovery within the following problem areas:

- Short-on-storage-conditions
- Storage violations
- Autoinstall problems
- VTAM ACB failures

State/action tables work independently of service periods and external triggers and are referenced when messages occur that are relevant to these entities. For an explanation on what state/action tables are and how they work, see “State/Action Tables” on page 11.

If you want to enable automated recovery for one of the problem areas listed above with respect to a CICS application, proceed as follows:

- If the recovery flag of the application (AUTOMATION FLAGS policy item) is set to NO, define a minor resource flag for the respective problem area in the customization dialogs under the MINOR RESOURCE FLAGS policy item of the CICS application and set its recovery flag to YES. For details, see “Automating Recovery for Transactions and Problem Areas” on page 30.
- Associate the respective CICS application with a set of state/action tables. To do this, you must perform two steps in the customization dialogs:
 1. Define a set of state/action tables as a CICS STATE/ACTION policy object for CICS (CSA entry type).
 2. Link the set to the subsystem under the STATE ACTION TABLE policy item of the APPLICATION object.

The state/action tables are read and the actions respectively state changes performed by the EVEEY00S routine (see “EVEEY00S—Common State Handler for State/Action Tables” on page 82). EVEEY00S is typically invoked from the MAT. It determines which set of state/action tables is associated with the subsystem that issued the message, and then consults the appropriate table. Besides EVEEY00S, the following components for support of state/action processing are shipped with CICS Automation:

- Default state/action tables for the four problem areas. These are:

Problem area	Name of state/action table
Short-on-storage condition	EVEESA01
Storage violations	EVEESA02
Autoinstall problems	EVEESA03
VTAM ACB failures	EVEESA04

- Common routines to be used by the action routines.
- Action routines for the four problem areas.

Note: Some of these routines make use of the information specified under the RCVRAUTO, RCVR SOS, and RCVRVIOL entries. For information on these entries, see “Chapter 5. MESSAGES/USER DATA Entries for CICS Automation” on page 45.

Every state/action table is associated with an *area* and a *product*. The area tag specifies for which of the four problem areas the table is intended, the product tag

says whether the table is to be used by CICS Automation or by IMS Automation. These tags must be specified in the first two rows of the table. The format for CICS Automation is:

```
PRODUCT=CICS  
AREA={AUTO|SOS|VIOL|VTAMACB}
```

In the third row of the table, you must specify the initial state of the table, that is, the state that is assumed when the table is consulted for the first time. Thus, the header of the sample table in Figure 4 on page 12 would have to look like this:

```
PRODUCT=CICS  
AREA=SOS  
STATE=0
```

How to Set Up Health Checking

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

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

Health checking is set up in the following manner:

Step 1: The health check program is written

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

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

The ACK or NACK are returned through the use of a DFHCOMMAREA. The user-written health check program is linked to by a CICS Automation health check program, which passes the 104-byte DFHCOMMAREA. The first 4 bytes are reserved for the characters ACK or NACK. The last 100 bytes can be used for a NACK message if the user-written program encounters an error. Refer to the samples for the format of the DFHCOMMAREA, and for an example of how to use the DFHCOMMAREA to return the response.

Step 2: The health check programs are defined to CICS

Use either the *CICS/MVS Resource Definition (Online)* or the *CICS/ESA Resource Definition (Online)* publication to do this.

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

The health check program is defined to CICS Automation under the MESSAGES/USER DATA policy item HEALTHCHK (see “HEALTHCHK—Health Checking” on page 56 on the **User Defined Data** panel. For example:

```

COMMANDS  HELP
-----
Command ==>          User Defined Data          Row 1 to 2 of 20
                   SCROLL==> PAGE

Entry Type : Application      PolicyDB Name   : SCENARIO
Entry Name  : CIGS1           Enterprise Name : TEST

Subsystem   : CIGS1
Message ID  : HEALTHCHCK

To change keyword-data pair, specify the following:

Keyword
Data
FUNCTION
(0,PAYCHECK,00:02:00,15,, 'Check payroll')
_____
_____
_____

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

```

Figure 10. Defining a Health Check Program to CICS Automation

Up to 10 health check programs can be defined for each CICS subsystem. Zero (0) is the first one defined. One (1) would be the second one, two (2) the third one, and so on. The program name, in this case PAYCHECK, is used to identify the specific health check program. You will also indicate how often you want the program executed (every two minutes) and how long you want to wait for a response before sending operator notification (fifteen seconds). The description (for example 'Check payroll') is used on the CICS Automation Health Checking panel, see "Health Checking" on page 121.

How to Set Up Link Monitoring

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

In order to activate link monitoring, perform the following steps:

1. Define the link under a CICS LINK policy object (CCN entry type).
2. Define the monitoring periods under a MONITORING PERIOD policy object (CVP entry type).
3. Connect the monitoring periods to the link under the MONITORING PERIOD item of the CICS LINK object.
4. Connect the link to a CICS application under the CICS CONNECTION item of the APPLICATION policy object.

For more details, see *System Automation for OS/390 Defining Automation Policy*.

Setting Up Echoplexing

Basic link monitoring ensures that VTAM connections are acquired and available but does not detect problems at the other end of the link. CICS Automation provides the ability to verify the other end of the link by sending data across it and waiting for a response. This is referred to as *echoplexing*. You can echoplex to any system or subsystem as long as the:

1. Link is defined on the primary CICS region.
2. Type of connection is either multiregion operation (MRO), LU6.1 or LU6.2.
Note: LU6.2 is not supported for links to IMS systems.
3. Target system is either CICS, or IMS.

SA OS/390 is not required on the target systems. With IMS target systems, no additional programming is required; CICS Automation uses the IMS /TEST function to get a response from IMS.

A CICS target subsystem requires access to EVESYCB7 (a CICS Automation program), and the definitions required for this program are made as shown:

```
DEFINE TRANSACTION(ECHO) PROGRAM(EVESYCB7)
DEFINE PROGRAM(EVESYCB7) LANGUAGE(ASSEMBLER)
```

Note: The default name for the transaction is ECHO. This can be changed as long as the transaction definition on the target system matches the transaction name on the primary system that is identified with the ECH0= keyword.

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 will 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 94.
- Enable non-terminal transaction security in CICS by modifying the CICS SIT to specify XTRAN=YES and XUSER=YES. Additional CICS definitions may require similar modification, such as PLTPIUSER.

Note: In order to perform any of the basic functions of CICS Automation, like displaying subsystem information, an operator must be authorized to use the ACF command. For this command, see *System Automation for OS/390 Operator's Commands*.

Adding Local Applications to the CICS Automation Operator Interface

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

To do this, write a module named EVVEU000 using the programming notes described below. This is the module that is called when option 99 is selected.

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

Programming notes:

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

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

and

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

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

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

and

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

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

```

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

```

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

```

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

```

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

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

```

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

```

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

```

'GLOBALV GETT EVESELNM'
MYNAME = EVESELNM

```

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

```

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

```

Using Linemode Functions

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

- Health checking
- SIT override
- Link monitoring
- Message options
- CICSPOST
- CEMTPPI

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

Health Checking

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

SIT Override

Linemode SIT overrides give user-written routines the capability of setting the SIT overrides, which can then be used by CICS Automation to control the startup of the CICS. A typical use of this linemode command will be to enable automation to perform cold startups on a given day of the week. For example, using SA OS/390 timer facilities, a user could set a timer to set the overrides to cold-start every Monday morning. Then, using service periods, the CICS system could be recycled, and a cold start would be performed.

Link Monitoring

Linemode link monitoring provides support for the link monitoring functions of CICS Automation. Support for most of the link monitoring capabilities are provided; excluded are system news update, service period update and recover-all-links functions.

Message options

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

CICSPOST

You can use the CICSPOST routine to set trigger conditions. Internally, the INGEVENT command of SA OS/390 is called. See *System Automation for OS/390 Operator's Commands* for more information on INGEVENT.

CEMTPPI

CEMTPPI allows you to code a CEMT command:

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

It accepts CEMT input as data, sends it across the program-to-program interface, and sends a response back to the originating task.

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

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

To invoke CICS RLS support for off-site recovery, you must start CICS systems with INGREQ using start type AUTO and specifying OFFSITE=YES in the **Appl Parns** field of the INGREQ input panel. See “Startups” on page 113.

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

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

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

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

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

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

Special Considerations for RACF-Protected Subsystems

When a CICS subsystem is RACF protected, the CICS autotask must first Signon to the subsystem to be able to issue commands. For this purpose, the user exits EVEEX080 and EVEEX081 are supplied as user-modifiable samples.

Tailoring:

Module EVEEX081 will need to be modified to your specific CICS names, operator sign-on IDs and passwords. For example, in EVEEX081 in the statement `evcesn.CICS10AA.opid = 'CICSn'` change CICS10AA to your CICS jobname and change CICSn to your operator sign-on ID. In `evcesn.CICS10AA.pswd = 'CICSn'` change CICS10AA to your CICS jobname and change CICSn to your password. You will have as many of these pairs of statements as you have CICS jobs that you want to RACF protect.

RACF Consideration:

You will need to define the operator sign-on ID to RACF with NOINTERVAL so the password will not have to be changed at certain intervals. The operator sign-on ID will need to have RACF authority for CEMT commands.

Changing Password:

In the event you need to change the password for the CICS operator sign-on, change the following statement in EVEEX081 from:

```
evcesn.CICS10AA.newpswd = ''
```

to:

```
evcesn.CICS10AA.newpswd = 'CICS9'
```

where CICS9 is the new password and CICS10AA is your CICS jobname.

Note: After EVEEX080 is executed, which will sign the operator on with the new password, be sure to change the password in EVEEX081 to the new password and erase the new password as follows:

from:

```
evecesn.CICS10AA.pswd = 'CICS1'  
evecesn.CICS10AA.newpswd = 'CICS9'
```

to:

```
evecesn.CICS10AA.pswd = 'CICS9'  
evecesn.CICS10AA.newpswd = ''
```

where CICS1 is the former password and CICS9 is the new/current password.

Chapter 5. MESSAGES/USER DATA Entries for CICS Automation

As CICS Automation is integrated into SA OS/390, you must enter any information for CICS Automation in the policy database via the customization dialogs. In most cases the customization dialogs precisely determine the format in which this information must be entered. There are, however, a number of CICS application-specific automation parameters that must or can only be specified as entries in the MESSAGES/USER DATA policy item of the respective application; for the MESSAGES/USER DATA policy item, see *System Automation for OS/390 Defining Automation Policy*. In these cases, the customization panels provide no information about the keywords and the format of their parameters.

The following chapter contains detailed descriptions of these automation entries. Note, however, that a general understanding of the MESSAGES/USER DATA policy item will be assumed.

Translating Format Descriptions into MESSAGES/USER DATA Entries

The following two examples show how to convert the formal descriptions of the keyword parameters into entries in the MESSAGES/USER DATA panels of the customization dialogs.

The first example is the RCVRSOS keyword. With this entry, you specify that the operator be notified and optionally one or more commands be issued when CICS is short on storage for a certain amount of time (see “RCVRSOS—Short-On-Storage Handling” on page 60 for more details). The format description of RCVRSOS is as follows:

Format
<pre>RCVRSOS TIMELIMIT=mm:ss [CMD=(,Command_Text)] . . . [CMD=(,Command_Text)]</pre>

The NAME=value pairs are called the *attributes* of the entry. RCVRSOS has two attributes, TIMELIMIT and CMD. TIMELIMIT must occur exactly once, CMD can occur zero or more times. For general information on the format descriptions, see “Notation for Format Descriptions” on page xi.

Now suppose you want the operator to be notified and a dump to be produced when the subsystem CICS1 is short on storage for more than 30 seconds. To specify a RCVRSOS entry for CICS1 in the customization dialogs, you must call the **Message Processing** panel for that subsystem:

```

COMMANDS  ACTIONS  HELP
-----
Message Processing                               Row 1 to 4 of 20
Command ==>>                                     SCROLL==>> PAGE

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

Subsystem : CICS1

Enter messages issued by this resource that will result in automated actions.
Actions: CMD = Command  REP = Reply  CODE = CODE  USER = User defined values

Action  Message ID                               Cmd  Rep  Code  User
Description
CMD_____RCVRSOS_____
How to react to storage shortage_____
_____
_____
_____
_____

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

```

Figure 11. Message Processing Panel of the Customization Dialogs 1

In this panel you specify the keyword of the entry (RCVRSOS in our example) in the **Message ID** field. The attributes are specified through the **Action** field. Here, two cases must be distinguished according to the following rule:

Rule

- The attribute names CMD, CODE, and REP must be entered in the **Action** field; the values for these attributes are specified in a follow-on panel.
- For all other attributes, you must enter USER in the **Action** field; in this case, both name and value are entered in a follow-on panel.

The fields on the right side of the panel specify how many actions of the respective type are associated with the message ID of the respective line.

To specify the value for the (optional) CMD attribute enter CMD in the **Action** field and press ENTER. This invokes the **CMD Processing** panel:

```

COMMANDS  HELP
-----
Command ==>          CMD Processing          Row 1 to 2 of 20
                               SCROLL==> PAGE

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

Subsystem   : CICS1
Message ID  : RCVRSOS

Enter commands to be executed when resource issues the selected message.

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

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

```

Figure 12. CMD Processing Panel of the Customization Dialogs

Every entry in this panel consists of three fields that correspond to the three items of the value list for the CMD attribute. Thus, the general format for the CMD attribute is

`CMD=([Pass/Selection],[Automated_Function],Command_Text)`

The format description of the CMD attribute for a certain keyword specifies what type of information you must enter in the three fields. For the RCVRSOS keyword, the omission of the first two values signifies that the fields **Pass/Selection** and **Automated Function** are to be left blank. A command text must be specified in the third field.

For the CMD, REP, and CODE attributes, the following notational conventions apply:

Notational Conventions for CMD, CODE, REP Attributes

- The '=' sign, the parentheses enclosing the value list, and the commas separating the individual values must not be entered in the panels. They just serve to make the format description more readable and to identify uniquely the panel field with which a value specification is associated.
- When the format of any value is specified in more detail, and this specification contains a comma, the value is enclosed in single quotes; these quotes must also not be entered in the respective panel field.

For the other attributes, the conventions are different; see "Notational Conventions for USER Attributes" on page 48.

For more information on the panel fields, see *System Automation for OS/390 Defining Automation Policy*. For the EVEERDMP command, see "EVEERDMP—CICS Dump" on page 80.

The second (required) attribute of RCVRSOS is TIMELIMIT. To enter the time limit, return to the **Message Processing** panel, where the number '1' will now be displayed in the **Cmd** field of the RCVRSOS entry. Specify USER in the **Action** field on the RCVRSOS line according to the rule stated on page 46, and press ENTER. The following panel is invoked:

```

COMMANDS  HELP
-----
Command ==>                                User Defined Data                Row 1 to 2 of 20
                                           SCROLL==> PAGE

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

Subsystem  : CICS1
Message ID : RCVRSOS

To change keyword-data pair, specify the following:

Keyword
Data
TIMELIMIT _____
00:30 _____
_____
_____

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

```

Figure 13. User-Defined Data Panel of the Customization Dialogs

As you can see from Figure 13, you must enter the attribute name in the **Keyword** field and the attribute value(s) in the **Data** field.

For attributes of the USER action type, the following conventions apply:

Notational Conventions for USER Attributes

- The '=' sign must not be entered in the panels.
- Everything to the right of the '=' sign, including parentheses, commas, and single quotes, *must* be entered in the **Data** field.

When you return to the **Message Processing** panel, the number '1' will be displayed in the **User** field.

The ABCODETRAN entry (see "ABCODETRAN—Transaction Abend Recovery" on page 54) supplies the second example. This entry specifies conditions for transaction recovery (for more details on transaction recovery, see "How to Define Transaction Recovery" on page 31). The format of the ABCODETRAN entry is as follows:


```

Format
ABCODETRAN[.tran]
  CODE=(tran,abend1,pgm,{INCLUDE|EXCLUDE})
  [CODE=(tran,abend1,pgm,{INCLUDE|EXCLUDE})]
  .
  .
  .
  [CODE=(tran,abend1,pgm,{INCLUDE|EXCLUDE})]

```

For ABCODETRAN, you must specify one or more instances of the CODE attribute.

If you want transaction recovery to be performed for transaction PAYR on CICS1 for all abend codes except AKC3, you must enter the ABCODETRAN entry for CICS1 as follows:

First specify the keyword in the **Message Processing** panel:

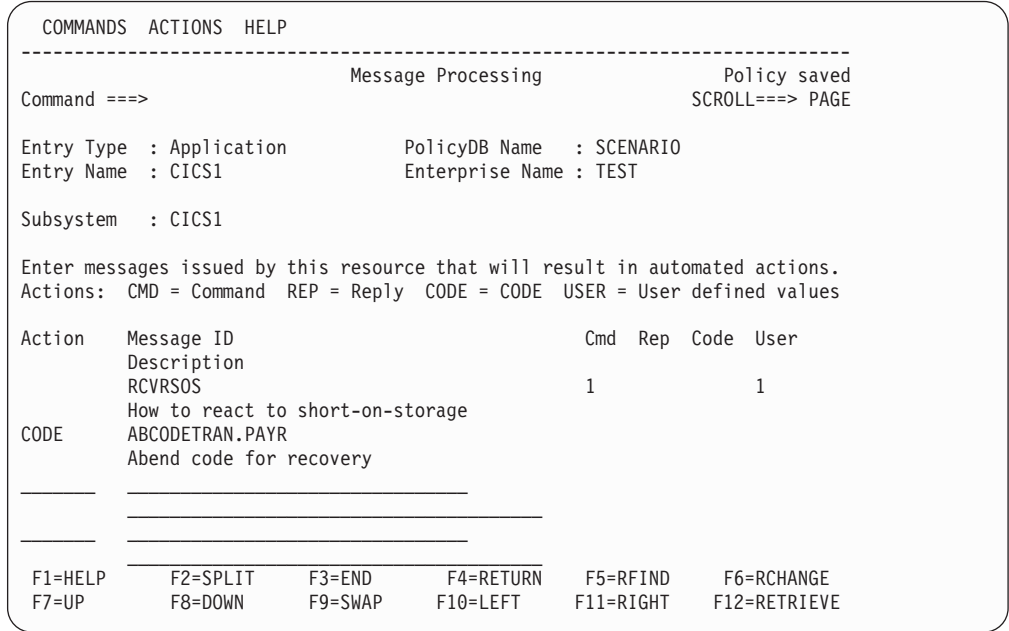


Figure 14. Message Processing Panel of the Customization Dialogs 2

Note that the keyword is expanded by the transaction name in order to restrict the application of the specified codes to transaction PAYR.

Now you must enter CODE in the **Action** column according to the rule stated on page 46. When you press ENTER, the **Code Processing** panel is called:

```

COMMANDS  HELP
-----
Code Processing                               Row 1 to 6 of 20
Command ==>                                SCROLL==> PAGE

Entry Type : Application                      PolicyDB Name : SCENARIO
Entry Name : CIGS1                          Enterprise Name : TEST

Subsystem : CIGS1
Message ID : ABCODETRAN.PAYR

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

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

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

```

Figure 15. Code Processing Panel of the Customization Dialogs

Here you must specify the values of the CODE attribute as displayed in Figure 15. The general format of the CODE attribute is:

CODE=([Code_1],[Code_2],[Code_3],Value_Returned)

For the ABCODETRAN keyword, **Code 1** must be a transaction name, **Code 2** an abend code, and **Code 3** the name of a program that abended. **Value Returned** must specify whether to include in, or to exclude from, recovery the combination specified by **Code 1** to **Code 3**. For other keywords, however, the fields can have a quite different function.

An asterisk (*) is admitted as a *trailing* wildcard character for the three **Code** fields; that is, you can specify simply * and ABC*, but not *ABC.

CICS-Specific MESSAGES/USER DATA Keywords

The following keywords are specific for CICS Automation.

Entry	Description
"ABCODESYSTEM—System abend recovery" on page 52.	Use this ID to define actions to be taken for specific abend codes.
"ABCODETRAN—Transaction Abend Recovery" on page 54.	Use this ID to define actions to be taken for transaction abend codes.
"HEALTHCHK—Health Checking" on page 56.	This ID is used to define the health check routines.
"LISTSHUT—Transaction Purging During Shutdown" on page 58.	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 60.	Use this ID if you want CICS Automation to take action for short on storage conditions.

Entry	Description
"RCVRVIOL—Storage Violation Handling" on page 63.	Include this ID to define commands to be executed as a result of recursive storage violations.
"RCVRAUTO—Autoinstall Time Limit" on page 59.	Use this ID to send alerts to the operator if a CICS autoinstall takes longer than the specified time.
"RCVRTRAN—Transaction Recovery" on page 62.	Use this ID to define actions to be taken when this specific transaction has abended.

ABCODESYSTEM—System Abend Recovery

ABCODESYSTEM—System abend recovery

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

Format

```
ABCODESYSTEM CODE=(msg,abend1,abend2,{RESTART|NORESTART})  
[CODE=(msg,abend1,abend2,{RESTART|NORESTART})]  
.  
.  
[CODE=(msg,abend1,abend2,{RESTART|NORESTART})]
```

Keyword and Parameter Definitions

CODE

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

msg

The abend message ID.

abend1 and *abend2*

The specific abend codes or qualifiers.

RESTART|NORESTART

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

Comments and Usage Notes

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

DFHLG0736	DFHLG0738	DFHLG0740	DFHRM0134	DFHRM0136
DFHRM0144	DFHRM0401	DFHDM0106	DFHTM0400	DFHSI1542

a restart of CICS requires INITIAL to be specified as the startup type. In these cases CICS Automation prevents a restart by ARM. Rather, the restart policy must be defined with the ABCODESYSTEM entry.

Examples of Usage

```

COMMANDS  HELP
-----
Code Processing                               Row 1 to 6 of 20
Command ==>                                SCROLL==> PAGE

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

Subsystem : CICS1
Message ID : ABCODESYSTEM

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

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

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

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

ABCODETRAN—Transaction Abend Recovery

ABCODETRAN—Transaction Abend Recovery

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

Format

```
ABCODETRAN[.tran]
  CODE=(tran,abend1,pgm,{INCLUDE|EXCLUDE})
  [CODE=(tran,abend1,pgm,{INCLUDE|EXCLUDE})]
  .
  .
  .
  [CODE=(tran,abend1,pgm,{INCLUDE|EXCLUDE})]
```

Keyword and Parameter Definitions

ABCODETRAN[.tran]

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

CODE

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

tran

The transaction ID.

abend1

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

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.

Examples of Usage

```

COMMANDS  HELP
-----
Code Processing                               Row 1 to 6 of 20
Command ==>                                SCROLL==> PAGE

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

Subsystem : CICS1
Message ID : ABCODETRAN

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

Code 1      Code 2      Code 3      Value Returned
CSFE_____ ATNI_____ *_____ EXCLUDE_____
CSFE_____ AKC3_____ *_____ EXCLUDE_____
*_____ *_____ *_____ INCLUDE_____
_____
_____
_____
F1=HELP    F2=SPLIT  F3=END     F4=RETURN  F5=RFIND   F6=RCHANGE
F7=UP      F8=DOWN   F9=SWAP    F10=LEFT   F11=RIGHT  F12=RETRIEVE
    
```

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

HEALTHCHK—Health checking

HEALTHCHK—Health Checking

Use this entry to define the health check routines.

Format

```
HEALTHCHK FUNCTION=(0,pgm,int,resp,{AUTO|NOAUTO},'desc')
          .
          .
          .
          [FUNCTION=(9,pgm,int,resp,{AUTO|NOAUTO},'desc')]
```

Keyword and Parameter Definitions

0 through 9

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

pgm

The program name as defined to the CICS region.

int

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

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

resp

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

AUTO|NOAUTO

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

desc

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

Comments and Usage Notes

1. It is recommended that the AUTO default be used with the HEALTHCHK parameter.
2. Refer to “How to Set Up Health Checking” on page 36 for more information about health checking.

Examples of Usage

```

COMMANDS  HELP
-----
User Defined Data                                Row 1 to 2 of 20
Command ==>                                     SCROLL==> PAGE

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

Subsystem : CICS1
Message ID : HEALTHCHK

To change keyword-data pair, specify the following:
Keyword
Data
FUNCTION
(0,HCPAY1,00:02:00,15,, 'Check payroll database')_____

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

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

```

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

LISTSHUT—Transaction purging during shutdown

LISTSHUT—Transaction Purging During Shutdown

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

Format

```
LISTSHUT {EXCLUDE|INCLUDE}=transid  
        [{EXCLUDE|INCLUDE}=transid]  
        .  
        .  
        .  
        [{EXCLUDE|INCLUDE}=transid]
```

Keyword and Parameter Definitions

EXCLUDE

Do not purge this transaction during a shutdown.

INCLUDE

Purge this transaction during a shutdown.

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 75.

Examples of Usage

```
COMMANDS  HELP  
-----  
User Defined Data                                Row 1 to 2 of 20  
Command ==>                                     SCROLL==> PAGE  
  
Entry Type : Application                          PolicyDB Name : SCENARIO  
Entry Name : CICS1                               Enterprise Name : TEST  
  
Subsystem : CICS1  
Message ID : LISTSHUT  
  
To change keyword-data pair, specify the following:  
  
Keyword  
Data  
EXCLUDE _____  
PAYR _____  
_____  
  
EXCLUDE _____  
BAT1 _____  
_____  
  
F1=HELP    F2=SPLIT    F3=END      F4=RETURN   F5=RFIND    F6=RCHANGE  
F7=UP      F8=DOWN     F9=SWAP    F10=LEFT   F11=RIGHT   F12=RETRIEVE
```

This example specifies that the PAYR and BAT1 transactions are specifically not purged during a shutdown of CICS1.

RCVRAUTO—Autoinstall Time Limit

Use this entry to send notifications to the operator if a CICS autoinstall takes longer than the specified time.

Format

```
RCVRAUTO TIMELIMIT=hh:mm:ss
```

Keyword and Parameter Definitions

TIMELIMIT

Use this attribute to determine if and when the Status Display Facility is updated with the error condition. The counter starts when the failure event occurs and stops at the completion of a successful install. If the install completes within the time specified, the Status Display Facility is not updated. If the install does not complete within the time specified, the Status Display Facility is updated.

Comments and Usage Notes

1. The **Critical** value of the AUTO thresholds of the subsystem is used to determine how often the update of the Status Display Facility is allowed to occur within a specific time frame. The AUTO thresholds must be defined in the customization dialogs under the CICS-specific RESOURCE THRESHOLDS policy item.
2. The RCVRAUTO entry is used by the EVEES302 action routine that is shipped with CICS Automation. This routine is called from the sample AUTO state/action table. On state/action tables, see “How to Set Up the State/Action Tables” on page 35 and “State/Action Tables” on page 11.

Examples of Usage

```

COMMANDS  HELP
-----
User Defined Data          Row 1 to 2 of 20
Command ==>                SCROLL==> PAGE

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

Subsystem   : CICS1
Message ID  : RCVRAUTO

To change keyword-data pair, specify the following:

Keyword
Data
TIMELIMIT _____
00:05:00 _____

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

```

The Status Display Facility is updated if an autoinstall stays suspended for longer than 5 minutes.

RCVRSOS—Short-On-Storage Handling

RCVRSOS—Short-On-Storage Handling

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

Format

```
RCVRSOS TIMELIMIT=mm:ss
        [CMD=(,cmd)]
        .
        .
        [CMD=(,cmd)]
```

Keyword and Parameter Definitions

TIMELIMIT

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

CMD

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

Comments and Usage Notes

1. There is a sample command list, EVEERDMP, that can be specified for the CMD attribute when a dump is to be produced.
2. The **Critical** value of the SOS thresholds of the subsystem is used to determine how often the specified commands are allowed to be issued within a specific time frame. The SOS thresholds must be defined in the customization dialogs under the CICS-specific RESOURCE THRESHOLDS policy item.
3. The RCVRSOS entry is used by the EVEES104 action routine that is shipped with CICS Automation. This routine is called from the sample SOS state/action table. On state/action tables, see “How to Set Up the State/Action Tables” on page 35 and “State/Action Tables” on page 11.

Examples of Usage

This following panel shows how the TIMELIMIT attribute can be specified:

RCVRSOS—Short-On-Storage Handling

```

COMMANDS  HELP
-----
Command ==>          User Defined Data          Row 1 to 2 of 20
                   SCROLL==> PAGE

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

Subsystem   : CICS1
Message ID  : RCVRSOS

To change keyword-data pair, specify the following:

Keyword
Data
TIMELIMIT _____
00:30_____
_____
_____

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

```

The following panel shows how the CMD attribute can be specified:

```

COMMANDS  HELP
-----
Command ==>          CMD Processing            Row 1 to 2 of 20
                   SCROLL==> PAGE

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

Subsystem   : CICS1
Message ID  : RCVRSOS

Enter commands to be executed when resource issues the selected message.

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

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

```

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

RCVRTRAN—Transaction Recovery

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

Format

```
RCVRTRAN[.tran]
      CMD=(,cmd)
      [CMD=(,cmd)]
      .
      .
      .
      [CMD=(,cmd)]
```

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, respectively 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 dialogs under the CICS-specific RESOURCE THRESHOLDS policy item.
2. The SA OS/390 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.
3. The RCVRTRAN entry is used by the EVEERTRN routine shipped with CICS Automation. EVEERTRN is typically called from the MAT.

Examples of Usage

```
COMMANDS  HELP
-----
                                CMD Processing                Row 1 to 2 of 20
Command ==>                                SCROLL==> PAGE

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

Subsystem   : CICS1
Message ID  : RCVRTRAN

Enter commands to be executed when resource issues the selected message.

Pass/Selection Automated Function/'*'
Command Text

MSG OP1,TRAN &EHKVAR1 FAILED _____
_____
_____

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

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

RCVRVIOL—Storage Violation Handling

Use this entry to set a trace as a result of recursive storage violations or to execute other commands.

Format

```
RCVRVIOL CMD=([{ON|OFF}], ,cmd)
          [CMD=([{ON|OFF}], ,cmd)]
          .
          .
          [CMD=([{ON|OFF}], ,cmd)]
```

Keyword and Parameter Definitions

CMD

The command or commands to be issued in the event of recursive storage violations. ON or OFF is used to set and unset a trace. Refer to the example.

Comments and Usage Notes

1. The **Critical** value of the VIOL threshold for the subsystem is used to determine how often the specified commands are allowed to be issued within a specific time frame. The VIOL threshold must be defined in the customization dialogs under the CICS specific RESOURCE THRESHOLDS policy item.
2. The RCVRVIOL entry is used by the EVEES202 and EVEES203 action routines that are shipped with CICS Automation. These routines are called from the sample VIOL state/action table. On state/action tables, see “How to Set Up the State/Action Tables” on page 35 and “State/Action Tables” on page 11.

Examples of Usage

```
COMMANDS  HELP
-----
Command ===>          CMD Processing          Row 1 to 2 of 20
                               SCROLL===> PAGE

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

Subsystem   : CICS1
Message ID  : RCVRVIOL

Enter commands to be executed when resource issues the selected message.

Pass/Selection Automated Function/'*'
Command Text
ON _____
MVS F CICS1 CSFE DEBUG,FAQE=ON_____

OFF _____
MVS F CICS1 CSFE DEBUG,FAQE=OFF_____

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

When the entry is coded this way, a trace is set on the first occurrence, then set off at the second occurrence. The third occurrence sets the trace back on and the fourth sets it off. This scenario continues until the **Critical** value of the VIOL threshold is exceeded.

RCVRVIOL—Storage Violation Handling

Chapter 6. CICS Automation Routines, Commands, and Definition Members

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

Subroutines

These subroutines provide generic functions for expanding automation capabilities beyond those supplied and supported by CICS Automation. They are typically invoked from other programs.

Commands and common routines

These commands and common routines can be called from the control file or the message table to invoke CICS Automation routines.

Definition members

These are modified during customization to provide additional function.

For general information on the format descriptions of the commands, see "Notation for Format Descriptions" on page xi.

Subroutines

The CICS Automation subroutines are:

“CICSQRY—Name Lookup” on page 67

Retrieves CICS subsystem information.

“CICSRCMD—Request a CICS Function” on page 70

Provides the ability to perform commands on any CICS region, local or remote.

CICSQRY—Name Lookup

Use this routine to retrieve CICS subsystem information.

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

Syntax

```
CICSQRY {REQ=VALIDATE,
        NAME={subsystem|resource_name|jobname},
        [TYPE={CICS|GROUP|DOMAIN|ANY|JOBNAME}] |
        REQ=GET,
        NAME={subsystem|resource_name|jobname}}
```

Keyword and Parameter Definitions

REQ=

The request type. The request types are:

VALIDATE

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

GET

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

NAME= *subsystem* | *resource_name* | *jobname*

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

subsystem

The name by which a CICS subsystem is known to SA OS/390.

resource_name

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

jobname

The jobname by which a CICS subsystem is known to SA OS/390.

TYPE=

Used to provide a specific type. The types are:

CICS (default)

Search for a specific CICS subsystem name, as it is known to SA OS/390.

ANY

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

DOMAIN

The NetView domain name coded in DSIDMN with the NCCFID parameter.

GROUP

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

JOBNAME

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

CICSQRY—Name lookup

Comments and Usage Notes

1. The return codes are:

Table 5. CICSQRY Return Codes

RC	Meaning
0	Good.
4	An internal error occurred.
8	A timeout occurred on a request forwarded to a remote system.
12	An internal error occurred.
20	A subsystem, group, or domain was not found for the search criteria specified.
24	The parameters for this request are invalid.
28	An internal error occurred.

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

EVELOOKUP_NAME

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

EVELOOKUP_TYPE

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

EVELOOKUP_JOBNAME

The jobname associated with the subsystem.

EVELOOKUP_DOMAIN

The NetView domain on which SA OS/390, managing this subsystem, is running.

EVELOOKUP_AUTOOPS

The NetView automated operator that handles automation for this subsystem.

EVELOOKUP_USERVAR

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

EVELOOKUP_APPLID

The specific VTAM application ID associated with this subsystem.

EVELOOKUP_RESHOME

The location of the resource in the following format:

sysplex.domain.system\VxRyMz

EVELOOKUP_RESLIST

The resource name in the following format

name/type/system

EVELOOKUP_AGENTDATA

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

agent_name sysplex_name system domain agent_version [netview_version]

EVELOOKUP_GROUP

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

CICSRCMD—Request a CICS Function

CICSRCMD—Request a CICS Function

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

Syntax

```
CICSRCMD NAME=name, [RESP=YES|ACK,] [OPER=operator,] CMD=cmd
```

Keyword and Parameter Definitions

NAME=

The name by which the target CICS subsystem is known to SA OS/390.

RESP=

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

OPER=

The operator, on the target domain, that will execute this command. If this is omitted, BASEOPER is used.

CMD=

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

Comments and Usage Notes

Table 6. CICSRCMD Return Codes

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

Commands and Common Routines

The CICS Automation subroutines are:

“CEMTPPI—CEMT PPI Short Syntax” on page 72

Allows you to code a CEMT command:

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

“CICSDLY—Change the Shutdown Delay Time” on page 73

This routine is used to change the default shutdown delay time specified in the **Shut Delay** field of the AUTOMATION INFO item of the

APPLICATION policy. The shutdown delay time is used during a shutdown process where user-defined shutdown commands are used and multiple passes are specified.

“CICSPOST—Post An External Event” on page 74

Use this routine to set trigger conditions in the status file.

“CICSPURG—Purge Transactions” on page 75

If you are using the MESSAGES/USER DATA keyword described in “LISTSHUT—Transaction Purging During Shutdown” on page 58, then you need to code this command so that CICS Automation checks for transactions that should or should not be purged during a shutdown.

“CICRSYU—CICS Resync” on page 76

The purpose of this routine is to resynchronize CICS information with what is currently operational in the system (such as the VTAM ACB status). When NetView is initialized, the routine updates the status file.

“CICSSHUT—Shutdown Processor” on page 77

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

“EVEED003—Critical Message Handler for the Status Display Facility” on page 79

This routine is used to identify those messages that are to be displayed on the Status Display Facility critical messages panels.

“EVEERDMP—CICS Dump” on page 80

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

“EVEEMIGR—Migrate Subsystem to CICS/TS” on page 81

Use this routine to migrate a subsystem from versions of CICS prior to CICS/TS.

“EVEEY00S—Common State Handler for State/Action Tables” on page 82

This routine is used to drive actions defined in the state/action tables.

“CICSHLTH—Linemode Health Checking” on page 83

This command allows you to perform linemode health checking.

“CICSOVRD—Linemode SIT Override” on page 85

This command allows you to perform linemode SIT overrides.

“CICSLM—Linemode Link Monitor” on page 87

This command allows you to perform linemode link monitoring.

“CICSMHDR—Linemode Message Options” on page 90

This command allows you to set message header options.

“CMASSHUT—CICSplex SM Address Space (CMAS) Shutdown” on page 92

This routine invokes CICSplex SM (CPSM) Application Programming Interface calls to shut the selected CMAS down.

CEMTPPI—CEMT PPI Short Syntax

CEMTPPI—CEMT PPI Short Syntax

CEMTPPI allows you to code a CEMT command:

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

It accepts CEMT input as data, sends it across the program-to-program interface, and sends a response back to the originating task.

Syntax

```
CEMTPPI subsys cemt-command-stream
```

Keyword and Parameter Definitions

subsys

The symbolic name by which this CICS subsystem is known to SA OS/390.

cemt-command-stream

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

Comments and Usage Notes

1. This command can route across domains.
2. The CEMT PERFORM SHUTDOWN option is not allowed across the program-to-program interface. Therefore, it cannot be issued with CEMTPPI.

CICSDLY—Change the Shutdown Delay Time

This routine is used to change the default shutdown delay time specified in the **Shut Delay** field of the AUTOMATION INFO item of the APPLICATION policy object. The shutdown delay time is used during a shutdown process where user-defined shutdown commands are used and multiple passes are specified. It indicates how long to wait for the shutdown before executing the next command in the sequence. Multiple passes are only used when the shutdown does not occur within the specified time frame.

Syntax

CICSDLY *hh:mm:ss*

Keyword and Parameter Definitions

hh:mm:ss

How long to wait for the previous command request to complete before executing the command specified in the next pass.

Comments and Usage Notes

If a delay time is not specified on a particular pass, the time defaults back to the entry in the policy database. This common routine needs to be used on each pass that you want the default delay time changed.

Examples of Usage

```

COMMANDS  HELP
-----
Command ==>>                                CMD Processing                                Row 1 to 2 of 20
                                                SCROLL==>> PAGE

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

Subsystem : CICS1
Message ID : SHUTFORCE

Enter commands to be executed when resource issues the selected message.

Pass/Selection Automated Function/ '*'
Command Text
PASS1 _____
CICSDLY 00:00:30 _____

_____

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

CICSPOST — Post An External Event

CICSPOST—Post An External Event

Use this routine to set trigger conditions. Internally, the INGEVENT command of SA OS/390 is called. Therefore, the TYPE parameter is ignored. For information on INGEVENT, see *System Automation for OS/390 Operator's Commands*.

Syntax

```
CICSPOST NAME=subsys,FUNCTION={SET|UNSET},EVENT=event  
[,TYPE={STARTUP|SHUTDOWN}]
```

Keyword and Parameter Definitions

NAME=

Is used to define the symbolic name by which this CICS subsystem is known to SA OS/390.

FUNCTION=

Specified whether the trigger is to be SET or UNSET.

EVENT=

The name of the external condition that this trigger represents.

TYPE=

Specifies whether this is a STARTUP or SHUTDOWN trigger. This keyword is ignored by INGEVENT.

Comments and Usage Notes

As CICSPOST calls INGEVENT internally, the TYPE parameter is ignored.

CICSPURG—Purge Transactions

If you are using the LISTSHUT keyword under the MESSAGES policy item (see “LISTSHUT—Transaction Purging During Shutdown” on page 58), then you need to code this command in the SHUTDOWN policy so that CICS Automation checks for transactions that should, or should not, be purged during a shutdown.

Syntax

```
CICSPURG [subsys]
```

Keyword and Parameter Definitions

subsys

The symbolic name by which this CICS subsystem is known to SA OS/390.

Comments and Usage Notes

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

Examples of Usage

```

COMMANDS  HELP
-----
Shutdown Command Processing          Row 1 to 2 of 20
Command ==>>>                        SCROLL==>>> PAGE

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

Subsystem      : CICS1
Shutdown Phase: NORM

Enter commands to be executed when the selected shutdown phase is invoked
for this subsystem.

Pass/Selection Automated Function/'*'
Command Text
PASS1_____
CICSSHUT NORMAL_____

PASS2_____
CICSPURG_____

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

```

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

CICSRSYC—CICS Resync

CICSRSYC—CICS Resync

The purpose of this routine is to resynchronize CICS information with what is currently operational in the system (such as the VTAM ACB status). If a CICS subsystem should be active, and health checking and link monitoring are defined for this subsystem, CICSRSYC will activate these monitoring functions. When you define the ACORESTART keyword under the MESSAGES policy item for a CICS application, you must specify CICSRSYC as the command. For ACORESTART, see *System Automation for OS/390 Defining Automation Policy*.

The format is:

Syntax

```
CICSRSYC subsys
```

Keyword and Parameter Definitions

subsys

The symbolic name by which this CICS subsystem is known to SA OS/390.

Comments and Usage Notes

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

Examples of Usage

```
COMMANDS  HELP
-----
                                CMD Processing                Row 1 to 2 of 20
Command ==>                                SCROLL==> PAGE

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

Subsystem   : CICS1
Message ID  : ACORESTART

Enter commands to be executed when resource issues the selected message.

Pass/Selection Automated Function/ '*'
Command Text
CICSRSYC CICS1 _____
_____
_____

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

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

CICSSHUT—Shutdown Processor

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

Syntax

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

Keyword and Parameter Definitions

cicsname

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

SDTRAN=*tranid* | NONE

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

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

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

Comments and Usage Notes

1. The CEMT PERFORM SHUTDOWN types are passed as parameters to this command.
2. CICSSHUT is recommended for all shutdown policies for subsystems automated by CICS Automation.
3. These commands are issued across the console. You may have to sign the console on first before issuing the command.

CICSSHUT—Shutdown Processor

Examples of Usage

```
COMMANDS  HELP
-----
Shutdown Command Processing          Row 1 to 2 of 20
Command ==>                          SCROLL==> PAGE

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

Subsystem      : CICS1
Shutdown Phase: NORM

Enter commands to be executed when the selected shutdown phase is invoked
for this subsystem.

Pass/Selection Automated Function/ '*'
Command Text
PASS1_____
CICSSHUT NORMAL_____

PASS2_____
CICSPURG_____

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

EVEED003—Critical Message Handler for the Status Display Facility

EVEED003—Critical Message Handler for the Status Display Facility

This routine is used to list messages in the critical message Status Display Facility panel.

Syntax

```
EVEED003 msgtext
```

Keyword and Parameter Definitions

msgtext

The message text and message identifier passed to the critical message panel.

Comments and Usage Notes

This must be called from the MAT because it uses the jobname associated with the message.

Examples of Usage

Example 1

```
* CICS STARTUP NUCLEUS MODULE NOT FOUND - NEEDS REPLY
IF MSGID='DFH1596' & TEXT = MESSAGE
  & HDRMTYPE = '>'
  THEN EXEC(CMD('EVEED003 'MESSAGE) ROUTE(ALL *))
      EXEC(CMD('ISSUEREP AUTOTYP=START') ROUTE(ALL *))
      DISPLAY(N) BEEP(N) HOLD(N) NETLOG(Y) SYSLOG(Y);
```

Example 2

```
IF MSGID='DFH0964' & TEXT = MESSAGE
  THEN EXEC(CMD('EVEED003 'MESSAGE) ROUTE(ALL *))
      DISPLAY(N) BEEP(N) HOLD(N) NETLOG(Y) SYSLOG(Y);
```

EVEERDMP—CICS Dump

EVEERDMP—CICS Dump

EVEERDMP will create a dump for specific CICS problems. It will dump 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.

Syntax

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

Keyword and Parameter Definitions

jobname

The jobname for this CICS.

subsys

The symbolic name by which this CICS subsystem is known to SA OS/390.

Examples of Usage

```
COMMANDS  HELP
-----
Command ==>          CMD Processing          Row 1 to 2 of 20
                                SCROLL==> PAGE

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

Subsystem   : CICS1
Message ID  : RCVRSOS

Enter commands to be executed when resource issues the selected message.

Pass/Selection Automated Function/'*'
Command Text

EVEERDMP CICS1 _____
_____
_____

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


EVEEMIGR—Migrate Subsystem to CICS/TS

Use this routine to migrate a subsystem from versions of CICS prior to CICS/TS.

Syntax

```
EVEEMIGR NAME=subsys,NEWVERSION=version
```

Keyword and Parameter Definitions

NAME=

The name by which the CICS subsystem is known to SA OS/390.

NEWVERSION=

Is used to define the new version of CICS/TS that is to be migrated to. Valid versions are:

- V1R1
- V1R2
- V1R3

Comments and Usage Notes

1. The return codes are:

Table 7. EVEEMIGR Return Codes

RC	Meaning
0	Function performed (normal exit)
1	Invalid keyword entered
2	Invalid CICS system NAME entered
3	Invalid NEWVERSION entered
4	The specified CICS system name does not exist
> 4	An unexpected error occurred and processing has not completed.

2. This command is only applicable when migrating to CICS/TS V1R1, V1R2, and V1R3.
3. The version entries in both the status file and the currently active Cglobal for the selected system will be modified.

Examples of Usage

Example

The following command must be issued after migrating CICSA from Version 4 to CICS/TS V1R1.

```
EVEEMIGR NAME=CICSA,NEWVERSION=V1R1
```

EVEEY00S—Common State Handler for State/Action Tables

EVEEY00S—Common State Handler for State/Action Tables

This routine is used to drive actions defined in the state/action tables. It is typically invoked from the MAT (see the examples).

Syntax

```
EVEEY00S [MSGID=msgid]  
          [,JOB=jobname]  
          [,MSGSTR=msgstring]
```

Keyword and Parameter Definitions

MSGID=

The message passed to the state/action table as an event.

JOB=

The jobname associated with the event.

MSGSTR=

The message string associated with the message. MSGSTR= cannot be coded unless MSGID= is coded.

Comments and Usage Notes

1. If MSGID= is not specified, the message that invoked the routine is used as the event.
2. If JOB= is not specified, the jobname associated with the message is used.
3. If the routine is **not** invoked from the automation table, MSGID= and JOB= **must** be coded.
4. Refer to “How to Set Up the State/Action Tables” on page 35.

Examples of Usage

Example 1

```
* VTAMACB STATE ACTION - VTAM DISCOVERED DOWN BY CICS  
IF (MSGID = 'DFH3463I'  
   | MSGID = 'DFHSI1572'  
   | MSGID = 'DFH1572' )  
& (TEXT = . 'X'52' . | TEXT= . 'X'5C' .  
   | TEXT = . '=52.' . | TEXT= . '=5C.' .)  
THEN EXEC(CMD('EVEEY00S MSGID=VTAMDN') ROUTE(ALL *));
```

Example 2

```
* VTAMACB STATE ACTION - ACB CLOSED BY CICS  
IF MSGID = 'DFH2316I'  
THEN EXEC(CMD('EVEEY00S MSGID=ACBCLS') ROUTE(ALL *));
```

CICSHLTH—Linemode Health Checking

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

Syntax

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

Keyword and Parameter Definitions

subsys

The name of the CICS subsystem.

programe

The name of the user-written health check program.

ACTION=START

Used to initiate health check processing.

ACTION=STATUS

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

ACTION=RESUME

Used to continue a process that was temporarily suspended.

ACTION=SUSPEND

Used to temporarily stop a process.

ACTION=STOP

Used to stop health check processing.

ACTION=CHECK

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

Comments and Usage Notes

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

Examples of Usage

Example 1 — Status on a Health Check Program

Command:

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

Message Response:

```
C AOF01   EVE441I  HEALTH CHECK PROGRAM EVECHLTH STATUS INACTIVE
C AOF01   EVE445I  ABNORMAL RESPONSE ON 05/25/95 09:51:35 -
```

CICSHLTH—Linemode Health Checking

Example 2 - Start of a Health Check Program

Command

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

Message Response:

```
C A0F01    EVE436I  CICS01A HEALTH START FOR EVECHLTH SUCCESSFUL.
```

CICSOVRD—Linemode SIT Override

CICSOVRD allows you to set CICS SIT override conditions prior to CICS startup. (Otherwise, CICS Automation only allows you to set override conditions through the INGREQ input panel; see “Chapter 8. Starting and Stopping Resources” on page 113.)

Syntax

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

          ACTION=STATUS
```

Keyword and Parameter Definitions

type

The type of startup.

subsys

The name of the CICS subsystem.

ACTION=SET

Used to change the SIT override for a CICS subsystem.

ACTION=STATUS

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

delim

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

data

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

KEYPOINT=[REQuired|OPTional]

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

Comments and Usage Notes

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

```
C AOF01   EVE556I  CICSOVRD Completed successfully
```

To clear an override, enter:

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

CICSOVRD—Linemode SIT Override

Examples of Usage

Example—Add an override for the PLTPI for CICS01A

Command:

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

Message Response:

```
C A0F01 EVE556I CICSOVRD Completed successfully
```

Example—Change the Keypoint option for CICS01A

Command:

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

Message Response:

```
C A0F01 EVE556I CICSOVRD Completed successfully
```

CICSLM—Linemode Link Monitor

CICSLM provides a linemode interface for the link monitoring functions of CICS Automation.

Syntax

```
CICSLM NAME=subsys, {ACTION=STARTLMT |
                        ACTION=STATUSLMT |
                        ACTION=CONNINFO |
                        ACTION=STOPLMT |
                        ACTION=STATUS, CONNECTION=conname |
                        ACTION=RECOVER, CONNECTION=conname |
                        ACTION=SUSPEND, CONNECTION=conname
                        [, FUNCTION={ECHO|MONITOR}] |
                        ACTION=RESUME, CONNECTION=conname
                        [, FUNCTION={ECHO|MONITOR}]}
```

Keyword and Parameter Definitions

subsys

The name of the CICS subsystem.

ACTION=STARTLMT

Used to start link monitoring processing.

ACTION=STATUSLMT

Used to determine if link monitoring is started or stopped.

ACTION=CONNINFO

Used to acquire information on connection information.

ACTION=STOPLMT

Used to stop link monitoring processing.

ACTION=STATUS

Used to acquire status on a named connection.

ACTION=RECOVER

Used to start a series of repair actions for a named connection.

ACTION=SUSPEND

Used to temporarily stop link monitoring processing.

ACTION=RESUME

Used to restart link monitoring processing after it was suspended.

conname

This is the 4-character CICS name for a connection

FUNCTION=ECHO

Used to run the echoplexing transaction over the named connection.

FUNCTION=MONITOR

Used to run monitoring for the named connection.

Comments and Usage Notes

ACTION=STATUS, **RECOVER**, **SUSPEND**, and **RESUME** require the use of the **CONNECTION** keyword.

Examples of Usage

Example—Get connection information for CICS01A

Command:

```
CICSLM NAME=CICS01A,ACTION=CONNINFO
```

Message Response:

```
C AOF01 EVE793I AUTOMATION DISPLAY - CONNINFO
C AOF01 EVE794I CURRENT ITEM - CONNID=C10A
C AOF01 EVE795I DATA IS APPLID=CICS10AA
C AOF01 EVE795I DATA IS DESCRIPTION=FROM CICS01A TO
C AOF01 EVE795I DATA IS DESIRED=DOWN
C AOF01 EVE795I DATA IS ACTUAL=UNKNOWN
C AOF01 EVE795I DATA IS MONITOR=ON
C AOF01 EVE795I DATA IS LASTCHK=
C AOF01 EVE795I DATA IS ECHOPLEX=
C AOF01 EVE796I END OF CONNINFO DISPLAY
```

Example—Get status information for specific connection

Command:

```
CICSLM NAME=CICS01A,ACTION=STATUS,CONNECTION=C10A
```

Message Response:

```
C AOF01 EVE793I AUTOMATION DISPLAY - STATUS
C AOF01 EVE794I CURRENT ITEM - CONN=C10A
C AOF01 EVE795I DATA IS LOCAL=CICS01A
C AOF01 EVE795I DATA IS REMOTE=CICS10AA
C AOF01 EVE795I DATA IS DESCRIPTION=FROM CICS01A TO
C AOF01 EVE795I DATA IS CONNTYPE=LU62
C AOF01 EVE795I DATA IS CRITICAL=NO
C AOF01 EVE795I DATA IS TIMEZONE=00:00 EAST
C AOF01 EVE795I DATA IS MONSTATUS=ON
C AOF01 EVE795I DATA IS ECHOSTATUS=
C AOF01 EVE795I DATA IS LASTCHK=
C AOF01 EVE795I DATA IS RESPONSE=
C AOF01 EVE795I DATA IS DESTLINK=DOWN
C AOF01 EVE795I DATA IS ACTIVELINK=UNKNOWN
C AOF01 EVE795I DATA IS SERVICE=UNKNOWN
C AOF01 EVE795I DATA IS ACQUIRE=UNKNOWN
C AOF01 EVE795I DATA IS INTERVAL=27:00
C AOF01 EVE795I DATA IS REPAIR=3
C AOF01 EVE795I DATA IS RD=05
C AOF01 EVE795I DATA IS AD=05
C AOF01 EVE795I DATA IS ED=
C AOF01 EVE795I DATA IS SYSTEM=CICS
C AOF01 EVE795I DATA IS ECHOPROC=
C AOF01 EVE796I END OF STATUS DISPLAY
```


Example—Resume monitoring for a specific connection

Command:

```
CICSLM NAME=CICS01A,ACTION=RESUME,CONNECTION=C10A,FUNCTION=MONITOR
```

Message Response:

```
C A0F01    EVE968I  LINK MONITORING REQUEST RESUME(MONITOR) FOR  
CONNECTION C10A WAS SUCCESSFUL
```

CICSMHDR—Linemode Message Options

CICSMHDR—Linemode Message Options

CICSMHDR allows you to change the message header options which display on the operator panel.

Syntax

```
CICSMHDR NAME=[domain],ACTION=SET
                OPID=opname
                DOMAIN=[ON|OFF]
                OPERATOR=[ON|OFF|DEFAULT]

                ACTION=STATUS
                OPID=opname
```

Keyword and Parameter Definitions

[domain]

The name of the NetView domain.

ACTION=SET

Used to change the message options for a domain or operator.

ACTION=STATUS

Used to inquire about the message options for a domain or operator.

opname

The NetView operator ID.

DOMAIN=[ON|OFF]

For ACTION=SET, this keyword turns message ID on or off for the domain and/or operator. Both can be set in a single command.

OPERATOR=[ON|OFF|DEFAULT]

For ACTION=SET, this keyword turns message ID on or off for the operator specified on the OPID keyword. A value of default allows the message id options to be controlled by the domain setting.

Comments and Usage Notes

1. If the NAME keyword is not supplied, a default of the current domain is used.
2. For ACTION=SET, both the OPID and OPERATOR keywords must be used if either is specified.

For ACTION=STATUS, no additional keywords are required.

3. This linemode command returns the following message to the invoking routine:

```
C AOF01    EVE449I  CICSMHDR NAME=AOF01 DOMAIN=OFF OPID= OPERATOR=
```

Examples of Usage

Example—Turn message ID on for an operator

Command:

```
CICSMHDR ACTION=SET,OPID=GULLA,OPERATOR=ON
```

Message Response:

```
C AOF01    EVE449I  CICSMHDR NAME=AOF01 DOMAIN=OFF OPID=GULLA
OPERATOR=ON
```

Example—Turn message id on for a domain

Command:

```
CICSMHDR ACTION=SET,DOMAIN=ON
```

Message Response:

```
C AOF01 EVE449I CICSMHDR NAME=AOF01 DOMAIN=ON OPID= OPERATOR=
```

Example—Reset message options for a domain and operator

Command:

```
CICSMHDR ACTION=SET,DOMAIN=OFF,OPID=GULLA,OPERATOR=DEFAULT
```

Message Response:

```
C AOF01 EVE449I CICSMHDR NAME=AOF01 DOMAIN=OFF OPID=GULLA  
OPERATOR=DEFAULT
```

CMASSHUT—CICSplex SM Address Space (CMAS) Shutdown

CMASSHUT—CICSplex SM Address Space (CMAS) Shutdown

This is a command list that determines whether or not this CICS subsystem is running a CICSplex SM Address Space (CMAS). It then uses the CPSM REXX API to shut down the CMAS.

Syntax

```
CMASSHUT [cmasname]
```

Keyword and Parameter Definitions

cmasname

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

This routine invokes the CICSplex SM (CPSM) Application Programming Interface calls to shut the selected CMAS down.

Comments and Usage Notes

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

Examples of Usage

```
COMMANDS  HELP
-----
Command ==>                                CMD Processing                Row 1 to 2 of 20
                                           SCROLL==> PAGE

Entry Type : Application                    PolicyDB Name  : SCENARIO
Entry Name  : CMAS1                        Enterprise Name : TEST

Subsystem   : CMAS1
Message ID  : SHUTNORM

Enter commands to be executed when resource issues the selected message.

Pass/Selection Automated Function/'*'
Command Text
PASS1_____
CMASSHUT_____

PASS2_____
MVS C CMAS1_____

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

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

Definition Members

The definition members are:

“EVESPINM—CICS PPI Initialization Member” on page 94.

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

CMASSHUT—CICSplex SM Address Space (CMAS) Shutdown

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 96

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.

“EVESCMT3—Message Exit Table for CICS” on page 98.

Identifies which transient data queue messages require automation for CICS Version 3 and higher.

EVESPINM—CICS PPI Initialization Member

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 96.

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

```
EVEMPINM TYPE=INITIAL,    INITIAL ENTRY
      BUFFQL=4,          BUFFER QUEUE LIMIT
      RECEIVERID=NETVCPPI, NPDS RECEIVER IDENTIFICATION
      USERID=[YES|NO],   INVOKE TRANSACTION WITH REAL USERID
      CONSOLE=           TERMID OF CONSOLE FOR COPC

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

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

EVEMPINM TYPE=ENTRY,     DEFINE A FUNCTION
      FUNCTION=HEALTH,  FUNCTION NAME
      TRANSID=COHR      TRANSACTION NAME

EVEMPINM TYPE=ENTRY,     DEFINE A FUNCTION
      FUNCTION=TEST,    FUNCTION NAME
      TRANSID=TSTL      TRANSACTION NAME

EVEMPINM TYPE=FINAL     REQUIRED END
```

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 associated 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.

EVEPINM—CICS PPI Initialization Member

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. EVEPINM must be link-edited into one of the CICS DFHRPL libraries.
3. At least one valid TYPE=ENTRY must be specified.
4. There must be a TYPE=ENTRY definition for each function that uses the CICS Automation program-to-program interface. The corresponding NetView side initialization member entry looks like this:

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

where CEMT is the function.

5. If you are running CICS Automation in more than one NetView domain on the same MVS system, then you need to provide unique RECEIVERIDs in this member. This must also be changed in the corresponding NetView program-to-program interface initialization member. See "EVENTASK—NetView PPI Initialization Member" on page 96, which explains where the matching RECEIVERID is changed for that member.

EVENTASK—NetView PPI initialization 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 94.

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

Syntax

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

Keyword and Parameter Definitions

BUFFQL

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

SERVER=

These entries define:

1. Whether this function is a REQUEST or a RESPONSE. A REQUEST is used to identify a receiver program to be invoked if NetView gets a CONVERSE or SEND from CICS. A RESPONSE is used to identify a sender program to be invoked if CICS sends a RESPONSE. See “EVESCCCI - CICS to NetView Communication Interface” on page 152.
2. The function, such as LMT (link monitor) or CEMT.
3. The operator ID under which the program runs, for example, AUTCPPI.
4. The command list or command processor used for this function, such as EVEEYPPS (the receiver program for LMT functions from CICS) and EVESNRSP (the common response handler).

RECEIVERID=

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

Comments and Usage Notes

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

EVENTASK—NetView PPI initialization member

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

4. If you are running CICS Automation in more than one NetView domain on the same MVS system, then you need to provide unique RECEIVERIDs in this member. This must also be changed in the corresponding CICS program-to-program interface initialization member EVESPINM. See “EVESPINM—CICS PPI Initialization Member” on page 94.

EVESCMT3—Message exit table for CICS

EVESCMT3—Message Exit Table for CICS

Message automation works with messages sent to an MVS console. Some CICS messages are not sent to an MVS console, but to transient data queues instead. The purpose of EVESCMT3 is to identify which of these messages require automation so that they can be sent to an MVS console for automation to take place.

A macro, EVEMXMET, is used to do this. To define the messages to be automated, edit EVESCMT3 and use the following EVEMXMET macro format:

Syntax

```
EVEMXMET TYPE=ENTRY  
        ,QUEUE=queue  
        ,MSGID=msgid
```

Keyword and Parameter Definitions

QUEUE=

The actual queue name to which this message is sent.

MSGID=

The message ID (without the A, I, or W suffix).

Comments and Usage Notes

1. The queue specifications for the EVEMXMET macros must contain the actual queue names.
2. EVESCMT3 must be link edited together with the EVESCME3 exit program in one of the CICS DFHRPL libraries.
3. A sample EVESCMT3 member is provided. You will see in this member that EVEMXMET has several TYPEs:

```
EVEMXMET TYPE=INITIAL  
EVEMXMET TYPE=ENTRY  
EVEMXMET TYPE=FINAL  
EVEMXMET TYPE=DSECT
```

The only TYPE that you should work with is ENTRY. Do not disturb the other TYPEs.

Examples of Usage

Example 1

The following is used for VTAM ACB messages:

```
EVEMXMET TYPE=ENTRY,  
        QUEUE=CSNE,  
        MSGID=DFHZC3463
```

Example 2

The following is used for program and transaction abend messages:

```

EVMXMET TYPE=ENTRY,
        QUEUE=CSMT,
        MSGID=DFHAC2230

EVMXMET TYPE=ENTRY,
        QUEUE=CSMT,
        MSGID=DFHAC2236

EVMXMET TYPE=ENTRY,
        QUEUE=CSMT,
        MSGID=DFHAC2237

EVMXMET TYPE=ENTRY,
        QUEUE=CSMT,
        MSGID=DFHAC2238

EVMXMET TYPE=ENTRY,
        QUEUE=CSMT,
        MSGID=DFHAC2240

EVMXMET TYPE=ENTRY,
        QUEUE=CSMT,
        MSGID=DFHAC2241

EVMXMET TYPE=ENTRY,
        QUEUE=CSMT,
        MSGID=DFHAC2242

EVMXMET TYPE=ENTRY,
        QUEUE=CSMT,
        MSGID=DFHAC2243
    
```

Example 3

The following is used for autoinstall error messages:

```

EVMXMET TYPE=ENTRY,
        QUEUE=CSNE,
        MSGID=DFHZC3482

EVMXMET TYPE=ENTRY,
        QUEUE=CSNE,
        MSGID=DFHZC3483

EVMXMET TYPE=ENTRY,
        QUEUE=CADL,
        MSGID=DFHZC5966
    
```

EVESCMT3—Message exit table for CICS

Part 3. Using CICS Automation

This part describes the tasks of the operator who manages CICS subsystems through CICS Automation.

Chapter 7. Using Panels and Working with Subsystems

This chapter explains how to use the CICS Automation panels and to work with subsystems. We assume that you have used and are familiar with the SA OS/390 operator interface. This chapter is devoted to describing those characteristics unique to CICS Automation. We try to provide plentiful details to guide you. But to thoroughly understand your role as the CICS Automation operator, you will need some hands-on experience with SA OS/390.

Using CICS Automation Panels

This section explains:

- Panel characteristics
- How to select tasks from panels
- How to use panel fields and function (F) keys
- Options available from the Main Menu.

To start a CICS Automation operator session and display the CICS Automation Main Menu, enter **CICS** on a NetView command line.

Panel Characteristics

All CICS Automation panels have similar characteristics, as illustrated in Figure 16.

```
EVEK0000          SA OS/390 - Command Dialogs
Domain ID  = IPSFM  ----- CICS -----      Date = 04/20/00
Operator ID = SCHR                               Time = 09:19:00

Resource 1      => CICSK8G                      Format: name/type/system
System 2       =>                               System name, domain ID or sysplex name

   1. Inquire           Display detailed status of CICS subsystem
   2. Start             Start a CICS subsystem, group or domain
   3. Shutdown         Shutdown a CICS subsystem, group or domain
   4. Triggers         Display start and shutdown trigger conditions
   5. Service Periods  Perform service periods functions
   6. Master Terminal  Perform master terminal functions
   7. Monitoring       Perform monitoring functions
   8. Broadcast        Send message to specific CICS subsystem(s)
   9. Support          Provide support functions
  99. Local Functions  Provide access to user defined local functions

ING006I THERE IS NO RESOURCE CICSK8G/APL 3
Command ==> 4
PF1=Help      PF2=End      PF3=Return
PF6=Roll 5
PF12=Retrieve
```

Figure 16. Common Characteristics of CICS Automation Panels

The list below identifies common characteristics of CICS Automation panels:

- 1** In this field, you must specify the resource you will work with. Most of the CICS Automation panels have this field, allowing you to change resources

without having to return to the main menu. On the CICS main panel, you can specify wildcards ('*' for a string and '%' for a single character) anywhere in the **Resource** field.

By entering a question mark ('?') at position one of the **Resource** field and pressing ENTER, you can call up a list of all CICS subsystems from the current sysplex that are defined to SA. You can then select a subsystem from this list. To see this list for another sysplex other than the current one, specify the respective sysplex in the **System** field, before you press ENTER.

Note however that many of the menu options are not valid for remote sysplexes.

- 2** In this field, you can specify the system, domain, or sysplex to which the command that you call by selecting an option is to be routed. If you specify a sysplex name and then enter a question mark in the **Resource** field you call up a list of all the CICS subsystems in the specified sysplex.
- 3** CICS Automation messages display here.
- 4** From the command line, you can enter CICS Automation operator commands, as well as any other command that can be entered from a NetView panel. You also select from the current menu by entering the option number in the command line.
- 5** You use the function keys primarily to navigate through the panels. Each function key has a self-explanatory label. However, if you see an unfamiliar function key, refer to the panel help information by pressing PF1.

Using the Main Menu

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

```

EVEK0000          SA OS/390 - Command Dialogs
Domain ID   = IPSFM   ----- CICS -----      Date = 04/20/00
Operator ID = SCHR                    Time = 09:28:09

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

  1. Inquire           Display detailed status of CICS subsystem
  2. Start             Start a CICS subsystem, group or domain
  3. Shutdown         Shutdown a CICS subsystem, group or domain
  4. Triggers         Display start and shutdown trigger conditions
  5. Service Periods  Perform service periods functions
  6. Master Terminal  Perform master terminal functions
  7. Monitoring       Perform monitoring functions
  8. Broadcast        Send message to specific CICS subsystem(s)
  9. Support          Provide support functions
 99. Local Functions  Provide access to user defined local functions

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

```

Figure 17. CICS Automation Main Menu

The following list briefly describes the options you can select from the main menu:

1. Inquire

Use this option to retrieve detailed information for a CICS resource. Specify the resource either in the **Resource** field of the main menu or subsequently on the **Subsystem Information** panel.

2. Start

Use this option to initiate the startup process of a resource. By choosing this option you call the INGREQ command of SA OS/390. See "Startups" on page 113.

3. Shutdown

Use this option to initiate the shutdown process of a resource. By choosing this option you call the INGREQ command of SA OS/390. See "Chapter 8. Starting and Stopping Resources" on page 113.

4. Triggers

Use this option if you want to display the triggers associated with a resource. By choosing this option you call the DISPTRG command of SA OS/390. See *System Automation for OS/390 Operator's Commands*

5. Service Periods

Use this option if you want to display or override the schedule associated with a resource. By choosing this option you call the INGSCHED command of SA OS/390. See *System Automation for OS/390 Operator's Commands*.

6. Master Terminal

This option provides an interface to the CEMT transaction where you can enter free-format CEMT commands. Refer to the CICS documentation for CEMT transaction information.

7. Monitoring

Use this option to work with link monitoring and health checking. See "Chapter 9. Monitoring Your CICS Subsystems" on page 117.

8. Broadcast

If you need to broadcast a message to a specific subsystem, group, or domain, select this option. See "Chapter 10. Broadcasting Messages" on page 125.

9. Support

Select this option to:

- Set a cold start indicator
- Work with the program-to-program interface
- Set message and trace defaults for your user ID
- Set the interface panels display options.

See "Chapter 11. Support Functions" on page 127.

99. Local Functions

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

Important!

The options 6 through 99 are only valid for the local sysplex. You cannot access a remote sysplex with any of these functions.

Using Fast Path

CICS Automation offers a *fast path* option that lets you go directly to any panel in its interface from NetView. Panels have identifying numbers based on their option number from the main menu. You *fast path* by entering a panel's identifying number on the command line. Several of the second-level panels have numbered menu options on them which take you to third-level panels. To get to the second-level panels, add a period and the number of the selection. The format to access third-level panels is *num.num.num*.

A fast path to the main menu is defined to a function key. You can return to the main menu panel from other panels by pressing PF4.

To fast path to a CICS Automation panel from NetView, enter:

CICS *nn*

or

CICS *subsys*

or

CICS *nn,subsys*

on the command line, where *nn* is the number corresponding to the option panel and *subsys* is the name of the CICS subsystem you want to work with.

Example

Entering CICS 9.2,CICS10AA

will take you from NetView into CICS Automation to the **SA/CICS – Program-to-Program Interface** panel, with the subsystem CICS10AA specified.

Selecting and Viewing Subsystems

This section explains 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 CICS-specific panels you can call a list of the available CICS subsystems by entering a question mark at position one of the **Subsystem** or **Resource** field. On the panels that belong to tSA OS/390 (INGREQ, INGSCHED, DISPTRG commands), you can use an asterisk (*) as a wildcard.

Figure 18 on page 107 shows a list of the CICS subsystems generated with the question mark function.

```

INGKYSTS          SA OS/390 - Command Dialogs          Line 1 of 16
Domain ID = IPSFM ----- Selection Panel ----- Date = 04/25/00
Operator ID = SCHR          Sysplex = KEY1PLEX          Time = 09:26:29

CMD: S Select / scroll
CMD Name      Type System      Compound      Desired      Observed      Nature
-----
s  CICSK1G     APL KEY1      SATISFACTORY UNAVAILABLE  SOFTDOWN
   CICSK1H     APL KEY1      SATISFACTORY AVAILABLE    AVAILABLE
   CICSK3A     APL KEY1      PROBLEM      AVAILABLE    PROBLEM
   CICSK3B     APL KEY1      PROBLEM      AVAILABLE    HARDDOWN
   CICSK3E     APL KEY1      PROBLEM      AVAILABLE    HARDDOWN
   CICSK4C     APL KEY2      SATISFACTORY AVAILABLE    AVAILABLE
   CICSK4D     APL KEY2      PROBLEM      AVAILABLE    HARDDOWN
   CICSK4F     APL KEY2      PROBLEM      AVAILABLE    HARDDOWN
   EYUCMS1A    APL KEY1      SATISFACTORY AVAILABLE    AVAILABLE
   EYUCMS1B    APL KEY2      SATISFACTORY AVAILABLE    AVAILABLE
   EYUCMS2A    APL KEY1      PROBLEM      AVAILABLE    HARDDOWN
   EYUCMS2B    APL KEY2      INHIBITED    AVAILABLE    SOFTDOWN
   EYUMAS1A    APL KEY1      PROBLEM      AVAILABLE    PROBLEM

Command ==>
PF1=Help      PF2=End      PF3=Return
PF8=Forward   PF9=Refresh  PF10=Previous PF11=Next
PF6=Roll      PF12=Retrieve

```

Figure 18. Selection Panel for CICS Resources

The list contains all subsystems of the KEY1PLEX sysplex (see line 3 of the panel) that are defined to SA OS/390. 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 3 through 5, for example, contain status information for the subsystems; for more details on the different status types, see *System Automation for OS/390 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

When you select option 1, Inquire, from the main menu panel, the **Subsystem Information** panel displays information for the subsystem specified in the **Subsystem** field. Figure 19 on page 108 displays subsystem information for CICSK1G/APL/KEY1. It is presented as one panel, but on the real screen you will have to scroll to view all fields.

```

EVEKQ400          CICS Automation: Subsystem Information Line 1 of 31
Domain ID = IPSFM                               Date = 04/25/00
Operator ID = SCHR                               Time = 13:08:41

Subsystem ==> CICS1G/APL/KEY1          (? for list) System ==>

Subsystem Status:
Status..... AUTODOWN
Since..... 04/20/00 11:46
Version..... V5R3
Jobname..... CICS1G
Netview Domain.. IPSFM

CICS Startup and Shutdown Information:
                Last Start      Last Shutdown
Initiated..... --/--/-- --:--:-- --/--/-- --:--:--
Completed..... 04/07/00 09:43:25 --/--/-- --:--:--
Start Type..... EMERGENCY
Indoubt Unit of Work..... Yes
Abend Message ID.....
Abend Code.....

CICS Schedule Information:
Next Start..... NONE NONE          Type..... AUTO START REQUIRED
Next Stop..... NONE NONE

CICS VTAM Communications Information:
Specific APPLID... IPSAMCIG          ACB Status.. OPEN
Generic APPLID... *****
Active Sessions...                  Pending Sessions...

CICS TCP/IP Communications Information:
TCP/IP Status..... N/A

CICS XRF Information:
XRF Status.....

Command ==>
PF1=Help      PF2=End      PF3=Return    PF4=CICS Menu      PF6=Ro11
              PF8=Forward  PF9=Refresh

```

Figure 19. Subsystem Information Panel

The panel fields contain the following information (not all of these fields are present in Figure 19):

- Subsystem ==>**
This field shows the name of the subsystem being displayed. It is an input field that can be used to specify another subsystem to display. This can be achieved by typing the name of the new subsystem and pressing the ENTER key.
- Subsystem Status:**
This section contains current subsystem status information.
- Status** This field displays the status of the subsystem.
- Since** This field displays the timestamp for the current automation status.
- Version**
This field displays the version of CICS that the subsystem is using.

Note: CICS Transaction Server V1.1 is shown as V5R1. Subsequent CICS TS releases will be displayed as V5Rx.
- Job name**
This field displays the JES job name for the address space.

NetView Domain

This field displays the name of the NetView domain that is automating the subsystem.

CICS Startup and Shutdown Information:

This section shows how the CICS subsystem was last started and last shutdown.

Initiated

This line details when the last startup and shutdown events were initiated.

Completed

This line details when the last startup and shutdown events were completed.

Start Type

This field displays the type of CICS start that occurred at the startup event time shown above. For example, AUTO, COLD, INITIAL.

Keypoint Taken

This field only appears for pre-CICS Transaction Server V1R1 regions. It is YES if a keypoint was taken at the last shutdown event. Otherwise it is NO.

Indoubt Unit of Work

This field only appears for CICS Transaction Server subsystems. If there were indoubt units of work at the last shutdown this field is set to YES, otherwise it is set to NO.

Abend Message ID

This field displays the CICS message ID that was detected as being the cause of the CICS system abending, otherwise it is blank.

Abend Code

This field may be set by an abend message and contains the abend code (User or System) that was detected as being the cause of the CICS system abending.

CICS Schedule Information:

This section details the timing of the next shutdown and startup of the subsystem.

Next Start

This field shows the date and time of the next scheduled start of this subsystem or the word "none" if no schedule exists.

Type This field displays the type of startup that CICS Automation will use when starting this subsystem. The following values may be displayed:

INITIAL START REQUIRED

Automation has determined that CICS will not start unless an INITIAL start is specified. Typically the message that has caused this problem will be in the Abend Message ID.... field. Further problem determination will require the investigation of the message in the *CICS Messages and Codes* manual.

INITIAL START SELECTED

An operator or system programmer has determined that CICS must be started with an INITIAL start. An override has been entered via the panels or via the CICSOVRD command to force the next startup to be INITIAL.

AUTO START REQUIRED

Automation has detected that either a keypoint was not taken, there are outstanding units of work or CICS has abended. In these cases an AUTO start is required to enable CICS to perform recovery actions.

COLD START SELECTED

An operator or system programmer has determined that CICS must be started with a COLD start. An override has been entered via the panels or via the CICSOVRD command to force the next startup to be COLD.

NONE

No start type has been set. The SIT start value will be used.

Next Stop

This field displays the next scheduled shutdown date and time or the word "none" if no schedule exists.

CICS VTAM Communications Information:

This section details the status of the SNA communications protocol.

Specific APPLID

This field shows the VTAM application ID that CICS is using to communicate to VTAM terminals.

ACB Status

This field shows the status of the VTAM ACB. A value of CLOSED means that no SNA terminals can log onto this subsystem.

Generic APPLID

This field shows the name of the GENERIC application ID if this CICS is running in XRF mode.

Active Sessions

This field shows the number of terminals that are currently engaged in communication with this subsystem. This count also includes the number of CICS to CICS (ISC links) and CICS to IMS links.

Pending Sessions

This field shows the number of terminals that want to connect to this subsystem but cannot.

CICS TCP/IP Communications Information:

This section will only appear for CICS Transaction Server V1.3 and above. It details the status of TCP/IP as defined to CICS.

TCP/IP Status

This field shows the status of the TCP/IP protocol as determined by CICS. The status is determined via a CEMT INQUIRE TCPIP command and the value of the status is the same as in the response to that command.

Port xxxx

This line details the status of the Ports that CICS is listening to. xxxx is the name of the CICS TCPIPService. The Status, port number and number of currently active connections are shown. This information is determined by issuing a CEMT INQUIRE TCPIPService command. All status values are the same as in the response to that command.

CICS XRF Information

This section details the status of XRF for this subsystem. This section will only appear if the CICS region has been defined to XRF.

XRF Status

This field shows the XRF status of this subsystem.

CICS Sysplex Information

This section details the status of sysplex functions for this subsystem. This section will only appear if VSAM RLS has been enabled for this subsystem.

RLS Status

This field will show either RLSACTIVE or RLSINACTIVE if CICS is participating in a VSAM Record Level Sharing environment. RLSACTIVE means that CICS can access VSAM data sets defined as RLS. RLSINACTIVE means that there is an error in the RLS environment and CICS cannot access the VSAM data sets defined as RLS.

You can view detailed status for another subsystem by changing the currently specified subsystem name in the **Subsystem** field and pressing ENTER.

An important note

The “Last Start” and “Last Shutdown” information is only accurate when:

1. The startup or shutdown did not occur while NetView was down.
2. This CICS subsystem was not moved to another processor since the last CICS shutdown.

Chapter 8. Starting and Stopping Resources

CICS Automation uses the `INGREQ` command of SA OS/390 for starting and stopping resources. For information on `INGREQ`, see *System Automation for OS/390 Operator's Commands*. In this section, only the special points you have to observe when starting or stopping a CICS resource will be described in detail.

To select a resource, refer to "Selecting a Subsystem" on page 106.

Startups

When you select option 2, Startup, from the main menu panel, you call the `INGREQ` command of SA OS/390. The input panel for `INGREQ` looks as follows:

```

INGKYRU0          SA OS/390 - Command Dialogs
Domain ID = IPSFM  ----- INGREQ -----      Date = 05/03/00
Operator ID = SCHR                               Time = 12:34:09

Resource => CICS1H/APL/KEY1          format: name/type/system
System   =>                          System name, domain ID or sysplex name

Request  => START                    Request type (START, UP or STOP, DOWN)
Type     => NORM                     Type of processing (NORM/IMMED/FORCE/user) or ?
Scope    => ONLY                    Request scope (ONLY/CHILDREN/ALL)
Priority  => LOW                     Priority of request (FORCE/HIGH/LOW)
Expire   =>                          , Expiration date(yyyy-mm-dd), time(hh:mm)
Timeout  => 0 / MSG                 Interval in minutes / Option (MSG/CANCEL)
AutoRemove =>                       Remove when (SYSGONE, UNKNOWN)
Restart  => NO                      Restart resource after shutdown (YES/NO)
Override => NO                      (ALL/NO/TRG/FLG/DPY/STS/UOW/INIT)
Verify   => YES                     Check affected resources (YES/NO/WTOR)
Precheck => YES                     Precheck for flags and passes (YES/NO)
Appl Parm =>

AOF710A VERIFY/REVISE INPUT AND THEN PRESS ENTER
Command ==>>
PF1=Help   PF2=End   PF3=Return           PF6=Roll
PF12=Retrieve
  
```

Figure 20. Input Panel for the `INGREQ` Command

The CICS-specific features concern the **Type**, the **Override**, and the **Appl Parm** fields:

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

Start Type	Explanation	Valid for
AUTO	Uses the restart dataset to determine the startup type.	All releases
COLD	Initiates a cold start.	All releases
INITIAL	Initiates a cold start with additional processing for CICS TS resources.	CICS TS V1R1 and above
LOGTERM	Initiates a startup, and then a shutdown as soon as the startup is complete.	Pre CICS TS V1R1

Start Type	Explanation	Valid for
NORM	This is the same as AUTO.	
STANDBY	Initiates a start for an XRF backup.	All releases

INGREQ accepts all of these types, regardless of the CICS version to which the resource belongs; thus, for example, LOGTERM is accepted for a CICS TS subsystem. It is up to the automation programmer to map the INITIAL type to an accepted start type for CICS4, and similarly for LOGTERM and CICS TS. This is done in the STARTUP policy item for the application within the policy database.

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

```

AOFKSEL3          SA OS/390 - Command Dialogs      Line 1 of 4
Domain ID  = IPSFM  ----- INGREQ -----      Date = 05/03/00
Operator ID = SCHR                                     Time = 12:34:12

The following start types are defined for CICS4/APL/KEY1
Select one item to be processed, then press ENTER.

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

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

```

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

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

Override

The following values are CICS specific:

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

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

Appl Parm

Here you can enter overrides to the system initialization table (SIT) in the format

KEYWORD=value

For details see the CICS documentation.

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

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

If you have not changed the default value of YES for the **Verify** field, CICS Automation will display a verification panel (see Figure 21) after you have pressed ENTER. This panel displays the target resource and in addition all the resources which SA OS/390 will try to start because the startability of the selected resource directly or indirectly depends on them.

```

AOFKVFY1          SA OS/390 - Command Dialogs          Line 1 of 3
Domain ID = IPSFM  ----- INGREQ -----             Date = 05/03/00
Operator ID = SCHR                                     Time = 12:34:11

Verify list of affected resources for request START

CMD: S show overrides  T show trigger details  V show votes
Cmd Name      Type System  TRG SVP  W Action Type  Observed Stat
-----
CICSK4C      APL  KEY2                Y      AUTO  UNAVAILABLE
JES2         APL  KEY2                AUTO  UNAVAILABLE
VTAM         APL  KEY2                AUTO  UNAVAILABLE

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

Figure 21. Verification Panel for INGREQ

For more information on the verification panel of INGREQ, see *System Automation for OS/390 Operator's Commands*.

Shutdowns

When you select option 3, Shutdown, from the main menu panel, the INGREQ panel displays as follows.

```
INGKYRU0          SA OS/390 - Command Dialogs
Domain ID = IPSFM  ----- INGREQ -----      Date = 05/03/00
Operator ID = SCHR                               Time = 16:43:22

Resource  => CICS4C/APL/KEY2                    format: name/type/system
System    =>                                     System name, domain ID or sysplex name

Request   => STOP                               Request type (START, UP or STOP, DOWN)
Type      => NORM                               Type of processing (NORM/IMMED/FORCE/user) or ?
Scope     => ONLY                               Request scope (ONLY/CHILDREN/ALL)
Priority   => LOW                               Priority of request (HIGH/LOW)
Expire    =>                                     , Expiration date(yyyy-mm-dd), time(hh:mm)
Timeout   => 0 / MSG                           Interval in minutes / Option (MSG/CANCEL)
AutoRemove =>                                     Remove when (SYSGONE, UNKNOWN)
Restart   => NO                                Restart resource after shutdown (YES/NO)
Override  => NO                                (ALL/NO/TRG/FLG/DPY/STS/UOW/INIT)
Verify    => YES                               Check affected resources (YES/NO/WTOR)
Precheck  => YES                               Precheck for flags and passes (YES/NO)
Appl Parm =>

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

Figure 22. Input Panel for INGREQ Command

Shutdown of CICS resources conforms to standard practice.

Chapter 9. Monitoring Your CICS Subsystems

CICS Automation provides two functions for you to monitor your CICS subsystems:

- Link monitoring
- Health checking

Select option 7, Monitoring, from the main menu to display the CICS Automation Monitoring panel, as shown:

```
EVEKM000          SA/CICS - Monitoring          Date: 05/25/00
Subsystem . . . . . _____ (? for list)      Time: 14:04
Select an option . . . . . _
1. Link monitoring
2. Health checking
```

Figure 23. CICS Automation Monitoring Panel

The following sections, “Link Monitoring” and “Health Checking” on page 121, describe the two functions accessed from this panel.

Link Monitoring

Link monitoring verifies that the interregion and intersystem connections (IRC and ISC) are active. This verification occurs at fixed intervals. When a link failure is detected, link monitoring will perform automatic recovery actions.

Basic link monitoring only ensures that the VTAM connections are available. To verify the other end of the link, the so-called echo facility, also referred to as *echoplexing*, must be activated. This is supported for links to CICS and IMS.

You can access the Link Monitoring interface in two ways, from NetView and from CICS. To access Link Monitoring from NetView, use option 7 from the CICS Automation main menu. To access Link Monitoring from CICS, log on to CICS and issue C0L0 as a transaction. This action will invoke the Link Monitoring interface running under CICS.

From the Monitoring panel of the CICS Automation operator interface, select option 1 to display the following:

```
EVEKM100          SA/CICS - Link Monitoring
Subsystem . . . . . CICS10AB (? for list)      Date: 01/16/00
Current Monitor Status . : ON                  Time: 11:26
Select an option . . . . . _
  *. Start          Start link monitor
  2. Stop           Stop link monitor
  3. Display        Display links
  4. News           Update system news

F1=Help   F2=End   F3=Return   F5=Refresh   F6=Roll
```

Figure 24. Monitoring Links Panel

The **Current Monitor Status** field shows the status of the link monitor for this subsystem. You can choose among the options listed to start or stop the link monitor, display the link, or update system news.

Note: If an option is preceded by an asterisk instead of a number, the option is not valid with the current status. For example, START would not be a valid option if the monitor status is ON.

Starting or restarting link monitoring for a subsystem is useful after a system configuration change because all definitions controlling link monitoring are re-loaded. When you display the links, option 3, the system news, is also displayed.

Displaying Links

Select the “Display” option from the Link Monitoring panel to view the following panel:

```

EVEKM130          SA/CICS - Display Links
Subsystem . . . . . CICS2      (? for list)      Date: 01/03/00
Time: 11:34
More:
Select a command: 1. mon on   3. echo on   5. recover
                  2. mon off  4. echo off  6. details  7. periods

Cmd  Conn  Applid  Description  Desired  Actual  Mon  Last  Echoplex
      status  status  check
-    C01A  CICS1   CICS2 TO CICS1  UP      UP      ON   11:32  ON
-    C01B  CICS3   CICS2 TO CICS2  UP      DOWN    ON   10:00  ON
-    C01C  IMS01   CICS2 TO IMS01  UP      UP      ON   11:31  ON
-    C01D  AS400   CICS2 TO AS400  UP      UP      ON   11:33  ON

.....
SYSTEM NEWS: CICS3 will be unavailable 01/05/00 from 0800-1200 for
system maintenance.
.....

F1=Help      F2=End      F3=Return   F4=CICS menu  F5=Refresh   F6=Roll

```

Figure 25. Display Links Panel

From this panel, you can:

- Reactivate monitoring for a link.
- Deactivate monitoring for a link.
- Reactivate the echo facility for a link. You use this function when the echo facility has been turned off or when it has been disabled by link monitoring.
- Deactivate the echo facility for a link.
- Initiate a recovery action for a link.
- View details about a specific link.
- View a day’s schedule.
- View the schedule for seven days.
- Override the schedule. This is similar to service period overrides.

The Display Links panel displays the following information:

Subsystem	The symbolic name by which this CICS subsystem is known to SA OS/390.
Conn	The four-character symbolic name by which this link is known to this CICS subsystem. This is defined on the CICS subsystem itself (SYSID) and in the Connection id field of the CICS CONNECTION policy object.
Applid	The symbolic name by which the remote subsystem is identified to VTAM.
Description	A short description of the link.
Desired status	What the status of this link should be: UP Current time is within a link monitoring period. DOWN Current time is outside a link monitoring period.
Actual status	The last status obtained. Statuses are: UP Link is available. DOWN Link is not available. TROUBLE Link is being recovered after a link failure. UNKNOWN Connection has not yet been monitored, or the current time is outside the monitoring period of the link.
Mon	Specifies the status of link monitoring for the link: ON Link monitoring is active. OFF Link monitoring has been deactivated by the operator.
Last check	Specifies the time in hours and minutes of the last check for this link.
Echoplex	Specifies the status of the echo facility for the link: blanks The echo facility is not being used. nnnnnnnn Number of messages sent and received to and from the remote system. PROBLEMS The echo facility did not receive a response from the remote system within the echo delay time specified in the Echo field of the CICS CONNECTION policy object linked to the subsystem. FAILED The echo facility detected an error. DISABLED The echo facility could not recover from a failure and is inoperative. OFF The echo facility has been deactivated by operator. ON Echoplexing is done when the link status changes to UP.
System News	Specifies up to 210 characters of installation-specific information that can be entered by the operator with the System News option.

If you select option 6, Details, from the Display links panel you will see the following information:

Local application ID	The symbolic name by which this CICS subsystem
-----------------------------	--

	is identified to VTAM and is defined in the APPLid field of the CICS CONTROL policy item.
Remote application ID	The symbolic name by which the remote subsystem (at the other end of the link) is identified to VTAM.
Connection type	Specifies the type of communication: MRO, LU6.1, or LU6.2.
Time zone location	Specifies the difference in hours and minutes between the remote link subsystem and Greenwich time. EAST and WEST indicate where the subsystem is situated compared to Greenwich.
Last monitoring check	Specifies the time in hours and minutes of the last check for this link.
Echoplex status	Specifies the status of the echo facility for the link.
Average response time	Average response time of the echo facility over the last five minutes.
Service status	Specifies whether the link is in service.
Acquire status	Specifies whether the link is acquired.
Check interval	Specifies the interval after which the status of the link is checked regularly.
Max. recovery actions	Specifies the maximum number of automatic recovery actions after detection of a link failure, as defined in the Max repair tries field of the CICS CONNECTION policy object.
Release delay time	Specifies the time delay between a CICS request to release the link and the next CICS request for that link, as defined in the Release delay field of the CICS CONNECTION policy object.
Acquire delay time	Specifies the time delay between a CICS request to acquire the link and the next CICS request for that link, as defined in the Acquire delay field of the CICS CONNECTION policy object.
Remote system type	Specifies the type of the remote system.
Remote echo process	Specifies the name of the remote echo process as defined in the Echo field of the CICS CONNECTION policy object.
Echo delay time	Specifies the time delay after which an echo response must have been received from the remote system.
Monitoring periods	Specifies the intervals (in the remote system's local time) during which the link is monitored.

Health Checking

Through health checking, you execute programs that check the health of an application running under CICS. CICS Automation initiates a health check program by sending a request across the program-to-program interface to CICS. CICS invokes the program, and sends back an Acknowledgement (ACK),

indicating that the program executed as expected, or it sends a Negative Acknowledgement (NACK), indicating that the program did not execute as expected. A NACK includes data describing the error.

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

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

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

```

EVEKM200          SA/CICS - Health Checking          Page: 1 of 1
                                                         Date: 01/09/00
Subsystem . . . . . CICS2      (? for list)         Time: 11:34

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

                                Last Status Check
CMD  Program  Description          Status    Date    Time    Response
-   HCPAY1    CHECK PAYROLL DATA  ACTIVE   01/09/00 10:32:00 ABNORMAL
-   HCEDI5    ACCESS TO EDITOR     INACTIV  01/06/00 09:00:00 NORMAL
-   HCFULL    95% FULL CONDITION  ACTIVE   01/09/00 11:30:05 NORMAL
-   HCACTV    95% ACTIVE CONDITION ACTIVE   01/09/00 09:09:59 NORMAL
-   HCAREC    ACCOUNTS RECEIVABLE ACTIVE   01/09/00 11:33:00 ABNORMAL
-   HCTERMA   95% TERMINALS ACTIVE INACTIV  01/06/00 09:00:00 NORMAL
-   HCOU1     ACCESS TO OUTMAIL FILE ACTIVE   01/09/00 11:33:10 ABNORMAL
-   HGIN1     ACCESS TO INMAIL FILE INACTIV  01/06/00 09:10:00 ABNORMAL
-   HCPAY2    PAYROLL SUBMIT       ACTIVE   01/09/00 08:32:55 NORMAL
-   HCTERM2   REMOTE TERMINAL PROGRAM INACTIV  01/03/00 08:00:00 NORMAL

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

```

Figure 26. Health Checking Panel

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

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

From this panel, you can:

- | | |
|-------------------|--|
| 1. Start | Initiate automation of a routine |
| 2. Stop | Stop automation of a routine |
| 3. Suspend | Temporarily stop automation execution of a routine |

4. Resume

Continue with a process that has been temporarily stopped

5. Detail

Expand upon the given information to show the details of a health check routine

6. Immed check

Immediately submit a status check regardless of status and scheduled time interval

Chapter 10. Broadcasting Messages

The Broadcast function allows you to send a message throughout the system, as needed. For instance, if you need to shut down a specific subsystem, you could warn users of the planned shutdown. Figure 27 displays a sample Broadcast Messages panel:

EVEKB100	SA/CICS - Broadcast Messages	Page: 1 of 1
		Date: 03/26/00
Subsystem, Group, or Domain	CIC010 (? for list)	Time: 08:00
Message	Text of message to be broadcast	
	CICS10AA will be shut down at 14:00 today for approximately 30 minutes. Please plan accordingly.	
Destination	Terminal (name, list, or ALL), or Operator name	
	ALL	
Opclass	Operator Classes to receive message	
Optional		
Time	1000_	hhmm, +hhmm, +mm, +m
Date		yy.ddd, mm/dd, mm/dd/yy, +d
Errterm		Terminal id or ORIG
Id (Title)		
Heading	YES	YES or NO
Protect	YES	YES, NO, or Prefix
Command ==>>>		
F1=Help	F2=End	F3=Return F4=CICS Menu F6=Roll

Figure 27. Broadcast Messages Panel

In the example, the message will be sent throughout the domain CIC010 at 10:00 o'clock, informing users of a planned shutdown of the subsystem CICS10AA. Protect is set to YES, so if the system goes down before this message is broadcast, the message will be saved.

Not all of the Broadcast panel fields are required. You **must** specify

- Message text
- Subsystem, group, or domain
- Destination and/or an operator class.

Optional fields include:

- | | |
|----------------|---|
| Time | When the message is to be delivered. If no time is entered, the message will be processed immediately, assuming the current time. |
| Date | When the message is to be delivered. The default is the current date. |
| Errterm | The terminal where an error message should be delivered if the broadcast message is undeliverable. The original or sending terminal is the default. |
| ID | Allows you to give the message title. |
| Heading | Includes time, date, and terminal ID to precede the message text, if YES is entered here. |

Protect Saves the message for display if CICS abends, if YES is specified. When the message is saved, it is stored in CICS temporary storage. You may specify a “prefix” for the message.

For further information on Broadcast options, refer to the CMSG documentation in *CICS Supplied Transactions*.

After you have issued a broadcast, CICS Automation will display a notification panel, showing you the success of the broadcast. If a broadcast is successful, the Broadcast Notification panel will display each subsystem for which the broadcast was successful.

If a broadcast fails, the **Broadcast Notification** panel will display why the broadcast failed. Some possible reasons for failure include a subsystem is inactive or the terminal specified is invalid for that subsystem. Figure 28 shows a sample Broadcast Notification panel.

```
EVEKB120          SA/CICS - Broadcast Notification          Page: 01 of 1
                                                           Date: 04/17/00
Subsystem. . . . . : CICS10AA                               Time: 15:53

Broadcast command failed for the following subsystem(s):
Subsystem  Fail Reason
CICS10AA  BROADCAST CANCELLED BECAUSE CICS10AA NOT ACTIVE
```

Figure 28. Broadcast Notification—Cancelled Example

This sample panel shows that CICS10AA did not receive the broadcast because it was not active.

Chapter 11. Support Functions

With CICS Automation Support Functions, you can:

- Set a cold start indicator
- Operate the program-to-program interface (PPI)
- Set message options

Select option 9 from the CICS Automation Main Menu to invoke the Support Functions panel shown in Figure 29.

```
EVEKJ000          SA/CICS - Support Functions
Subsystem or Domain . . . CICS4F_ (? for list)
Date: 08/22/00
Time: 13:43
Select an option . . . . . _
    1. Set cold start indicator
    2. Program-to-Program Interface
    3. Set Message Id Options
```

Figure 29. Support Functions

Cold Start Indicator

With this option, you can either set the startup type of a CICS subsystem to COLD or (for CICS TS VIR1 and above) to INITIAL, or reset it to the default. How this is done depends on the level of SA OS/390 and CICS Automation on the target system. If the target system has V2R1, the INGLIST command will be invoked; if not, CICS Automation-specific panels will be displayed.

V2R1 Target System

When the system on which the CICS subsystem runs has a V2R1 SA OS/390, you will invoke the update panel of the INGLIST command by selecting the cold start indicator option:

```

INGKYST1          SA OS/390 - Command Dialogs
Domain ID = IPSFN ----- INGLIST ----- Date = 08/23/00
Operator ID = SCHR          Sysplex = KEY1PLEX Time = 09:24:52

Resource . . : CICS1G/APL/KEY1
Description. : CICS TS V1.3 Key1

Specify the action to be performed:

Action ==> 1_      1. Set START Type COLD____
                  2. Set STOP  Type  _____

                  3. Set Automation Flag NO
                  4. Set Hold Flag YES
                  5. Set Group failed flag  __ (Yes or No)

                  6. Set Observed Status _____
                  7. Set Automation Status  _____

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

```

Figure 30. Update Panel of INGLIST

Here you must specify the desired startup type in the **Set START Type** field; enter 1 in the **Action** field, and then press ENTER.

For more information on INGLIST, see *System Automation for OS/390 Programmer's Reference*

V1R3 Target System

For target systems with V1R3, CICS Automation provides its own panels for setting the startup type. These differ depending on the release of the CICS subsystem.

Pre-CICS TS V1R1 Subsystem

Select option 1 from the Support Functions panel to invoke the **Set Cold Start Indicator** panel shown in Figure 31 on page 129. This panel shows the current setting of the cold start indicator for a pre-version CICS TS V1R1 system. You can toggle this setting between YES and NO. If there is no current setting, the **Current cold start indicator** field will be blank.


```

EVEKJ100          SA/CICS - Set Cold Start Indicator
Subsystem . . . . . CICS10AA
Current cold start indicator . . NO
Select an option . . . . . _
1. NO              Use default start options on next CICS startup
2. YES             Cold start CICS on next startup

```

Figure 31. Set Cold Start Indicator Panel (Pre-CICS TS V1R1 Subsystem)

Selecting a Cold Start Option: Option 1, NO, tells CICS Automation to use the default start options the next time CICS is started. If you select YES, CICS Automation will use a cold start on the next startup.

CICS TS V1R1 Subsystem

Select option 1 from the Support Functions panel to invoke the Set Cold Start Indicator panel shown in Figure 32. This panel shows the current setting of the cold start indicator for a CICS TS V1R1 system and above. You can toggle this setting between NONE, COLD and INITIAL. If there is no current setting, the **Current cold start indicator** field will be blank.

```

EVEKJ100          SA/CICS - Set Cold Start Indicator
Subsystem . . . . . CICS10AA
Current cold start indicator . . NONE
Select an option . . . . . _
1. NONE           Use default start options on next CICS startup
2. COLD           Cold start CICS on next startup
3. INITIAL        Initial start CICS on next startup

```

Figure 32. Set Cold Start Indicator Panel (CICS TS V1R1 Subsystem and Above)

Selecting a Cold Start Option: Option 1, NONE, tells CICS Automation to use the default start options the next time CICS is started. If you select COLD, CICS Automation will use a cold start and if you select INITIAL, CICS Automation will use an INITIAL start on the next startup.

Program-to-Program Interface

Select option 2 from the Support Functions panel to access the program-to-program interface panel, shown in Figure 33 on page 130.

The program-to-program interface panel displays the status of the:

- Specified subsystem
- Program-to-program interface on the CICS side
- Program-to-program interface on the NetView side (EVENTASK)

- NetView Subsystem Interface (SSI), which passes the buffers for the program-to-program interface.

From this panel, you can start or stop the PPI for the CICS subsystem or for the NetView side (EVENTASK).

```

EVEKJ300          SA/CICS - Program-to-Program Interface
Subsystem . . . . . CICS10AA (? for list)          Date: 03/26/00
                                                    Time: 13:17
Subsystem status . . . . : STARTED
Subsystem PPI status . . : INACTIVE

Select an option . . . . _

    *. Start          Start subsystem PPI
    *. Stop           Stop subsystem PPI

EVENTASK status . . . . : ACTIVE
NetView subsystem status : ACTIVE

Select an option . . . . _

    *. Start          Start EVENTASK
    2. Stop           Stop EVENTASK

Command ==>
F1=Help      F2=End      F3=Return  F4=CICS Menu  F5=Refresh  F6=Roll

```

Figure 33. Program-to-Program Interface

Note: If the subsystem status is active, starting it will not be an option. CICS shows this by putting **ACTIVE** or **STARTED** in the status field and by inserting an asterisk (*) in place of the Start option number. If the subsystem is **not** active, you cannot start the subsystem interface.

Message Options

Select option 3 from the Support Functions panel to access the Set Message Id Display panel, shown in Figure 34 on page 131. From this screen you can set message options for the operator or for the domain.

```

EVEKJ400          SA/CICS - Set Message Id Display
                                                    Date: 03/20/00
Domain or Subsystem. . . . . CICS10AA (? for list)   Time: 13:16
Operator id . . . . . MYUSERID

Message option for operator: DEFAULT
Select an option . . . . . _

    1. ON              Display message id with panel messages
    2. OFF             Suppress message id from panel messages
    3. DEFAULT         Use display option for domain(domainwide default)
    Note: Affects messages for operator in all CICS subsystems of domain.

Domain affected . . . . . : AOF10
Message option for domain. : OFF
Select an option . . . . . _

    1. ON              Display message id with panel messages
    2. OFF             Suppress message id from panel messages
    Note: Affects messages in all CICS subsystems of domain.

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

```

Figure 34. Set Message Id Display Panel

Setting the option to **ON** will cause message numbers to display; setting the option to **OFF** will suppress message numbers. This option can be set for all CICS subsystems within a domain (system-wide default), or the operator can override the default for a specific user in the domain.

Note: If on the previous screen a subsystem was specified, CICS Automation will automatically insert the correct domain name in the “Domain Affected” field when you enter the Set Message Id panel. If you want to work with a different domain, simply enter a valid domain name in the field, or type ? and press the ENTER key for a list.

Chapter 12. The Status Display Facility

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

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

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

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

CICS-critical messages

Several messages are defined in the automation table as critical. These are routed to the Status Display Facility for display on the CICS Automation Critical Messages panel.

Health checking

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

VTAM ACBs

Messages associated with VTAM ACBs are routed to the Status Display Facility VTAM ACB panels.

Autoinstall

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

Link monitoring

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

CICS storage

Short-on-storage and storage violation messages are routed to the Status Display Facility storage panels.

CICS timers

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

CICS transactions

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

To use the CICS Automation Status Display Facility panels, enter **SDF** on a NetView panel command line. A panel similar to the following will display:

SYSTEM						SA OS/390 - SUPPORT SYSTEMS					
System	Subsystems	WTORs	Gateways	Products	System						
KEY1	IMS712M1	IM631C4	IPUFMI	C I D O	P V M B T U						
KEY2	CICSK4D	NETATST2	IPSFNO	C I D O	P V M B T U						
KEY3			IPSF00	C I D O	P V M B T U						
KEY4				C I D O	P V M B T U						

08/16/00 07:37

===>

1=HELP 2=DETAIL 3=RETURN 6=ROLL 8=NEXT SCR 10=LEFT 11=RIGHT 12=TOP

Figure 35. Status Display Facility Main Panel

Note: Sample Status Display Facility panels are provided with CICS Automation. The programmer customizes the panels for your specific environment, so the panels shown here will not look exactly like your panels.

This could be your primary panel that lists the systems and their status. The color of KEY1 through KEY4 will reflect the most critical status of any resource in that system.

If you place the cursor under the letter **C** on the panel displayed in Figure 35 and press PF8, the following panel displays (assuming you are using the default sample panels):

```

KEY2C                CICS MONITOR PANEL

                    CICS Critical Messages

                    Health

                    VTAM ACB

                    Autoinstall

                    Link Monitor

                    CICS Storage

                    CICS Timers

                    CICS Transactions

                                                    08/16/00 07:38

===>
PF1=HELP 2=DETAIL 3=END 4=HWD 6=ROLL 7=UP 8=DN                12=TOP

```

Figure 36. The CICS Monitor Panel

This shows several categories in which CICS status is important. If the letter C shown on the previous panel was red, then at least one of the items on the CICS Monitor panel will be red. Tab down to the red item and press PF8. This displays the messages logged against that item, as shown in the following:

```

EVEDTIM1            CICS: Timers

                    System      Message text
04/16/95 08:00:32 CICSA      'START DENIED BY OTHER EXIT'
04/16/95 08:00:29 CICS06AB   'SUBSYSTEM STATUS IS STOPPED.'
04/15/95 19:00:20 CICSTST    'START DENIED BY OTHER EXIT'
04/15/95 18:00:49 CICSTST1   'TIMED SHUTDOWN PENDING, TRIGGERS NOT SET'

                                                    04/16/95 13:07

===>
PF1=HELP 2=DETAIL 3=RET          6=ROLL 7=UP 8=DN          10=LF 11=RT 12=TOP

```

Note: If the full message is not displayed on the screen, press PF11 to shift to the right.

To see the detail of a message, tab down to that message and press PF2. This displays the following:

```
----- DETAIL STATUS DISPLAY -----
                                         1 OF 4

COMPONENT: CICSA                      SYSTEM : KEY2
COLOR   : RED                          PRIORITY : 200
DATE    : 04/16/95                     TIME     : 08:00:32
REPORTER : GATAOF06                    NODE     : AOF06
REFERENCE VALUE: CICSA_TI_
'START DENIED BY OTHER EXIT'

===>
PF3=RET 4=FPI 6=ROLL 7=UP 8=DN 9=AST 10=DEL 11=BOT 12=TOP
```

To delete a message, press PF10 from this screen.

Note: If any of the panels have 1 of X in the upper-right corner of the screen, where X is a number greater than 1, subsequent panels contain additional data. Press PF8 to scroll forward to view the information. Press PF7 to scroll back.

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

Warning!

CICS Automation uses the program-to-program interface for issuing CEMT transactions from the operator interface, for link monitoring, and 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.

Program-to-Program Interface Components in NetView and CICS

Figure 37 on page 138 illustrates CICS Automation program-to-program interface components in NetView and in CICS.

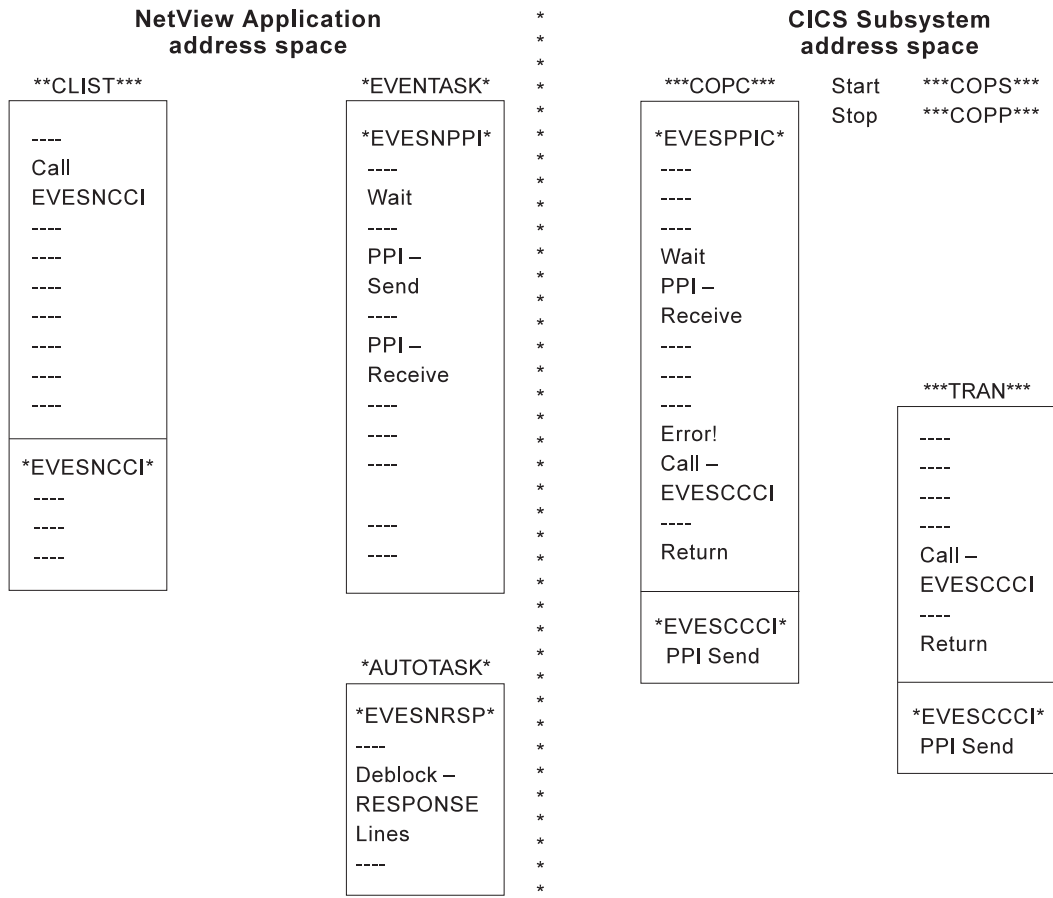


Figure 37. Program-to-Program Interface Components in NetView and CICS. Two of the programs shown in this figure have not been previously mentioned: EVESNPPI and EVESPPIC. EVESNPPI is the NetView subtask program. EVESPPIC is the CICS long-running receiver program. These are internal programs and are not described in this manual.

NetView Requests Using the Program-to-Program Interface

The following requests from NetView are described in this section:

- CONVERSE
- SEND
- CANCEL

EVESNCCI is used for these requests. This section only provides an overview of how EVESNCCI works. The programming details, such as command syntax, return codes, and segment support, are provided in "EVESNCCI—NetView to CICS Communication Interface" on page 146.

CONVERSE from NetView

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

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

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

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

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

A for ACK

The following message is sent to the requestor:

```
EVE128I POSITIVE ACKNOWLEDGEMENT
```

N for NACK

The following message is sent to the requestor:

```
EVE129E response-data
```

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

R for RESPONSE

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

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

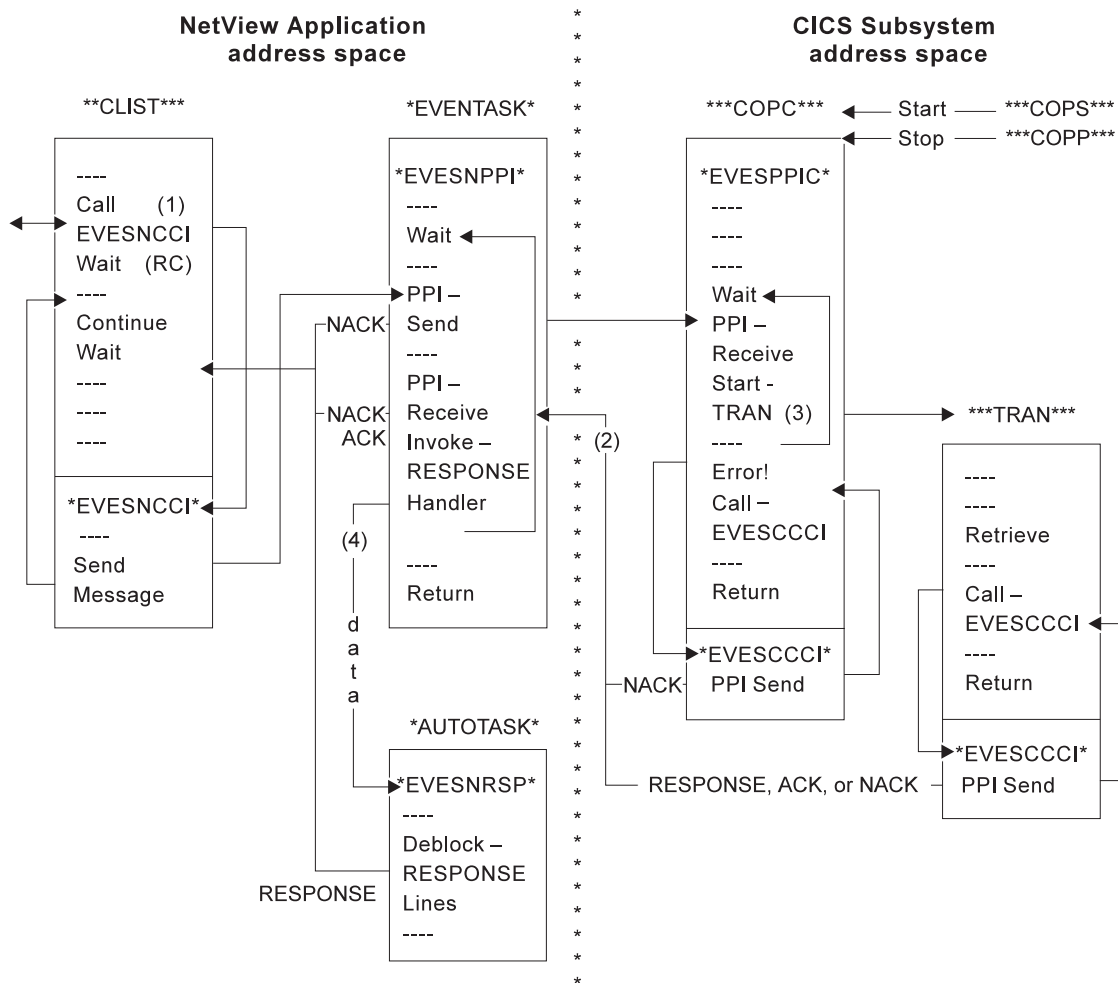


Figure 38. An EVESNCCI CONVERSE Request

Notes:

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

SEND from NetView

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

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

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

Refer to the following figure:

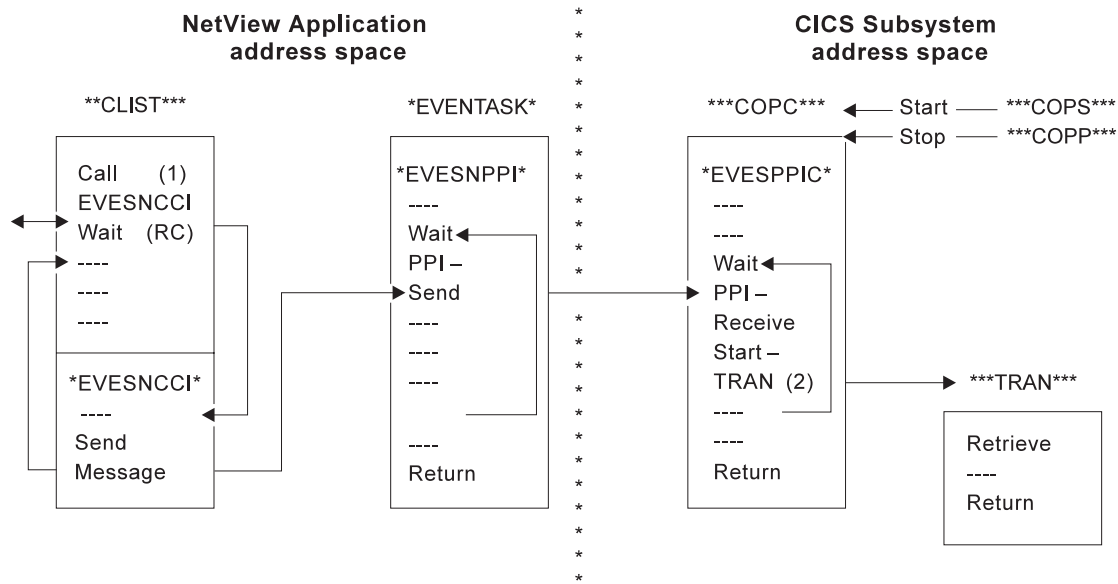


Figure 39. An EVESNCCI SEND Request

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

CICS Requests Using the Program-to-Program Interface

The following requests from CICS are described in this section:

- CONVERSE
- SEND

EVESCCCI is used for these requests. This section only provides an overview of EVESCCCI. The programming details, such as command syntax and return codes, are provided in "EVESCCCI - CICS to NetView Communication Interface" on page 152.

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 40 on page 143.

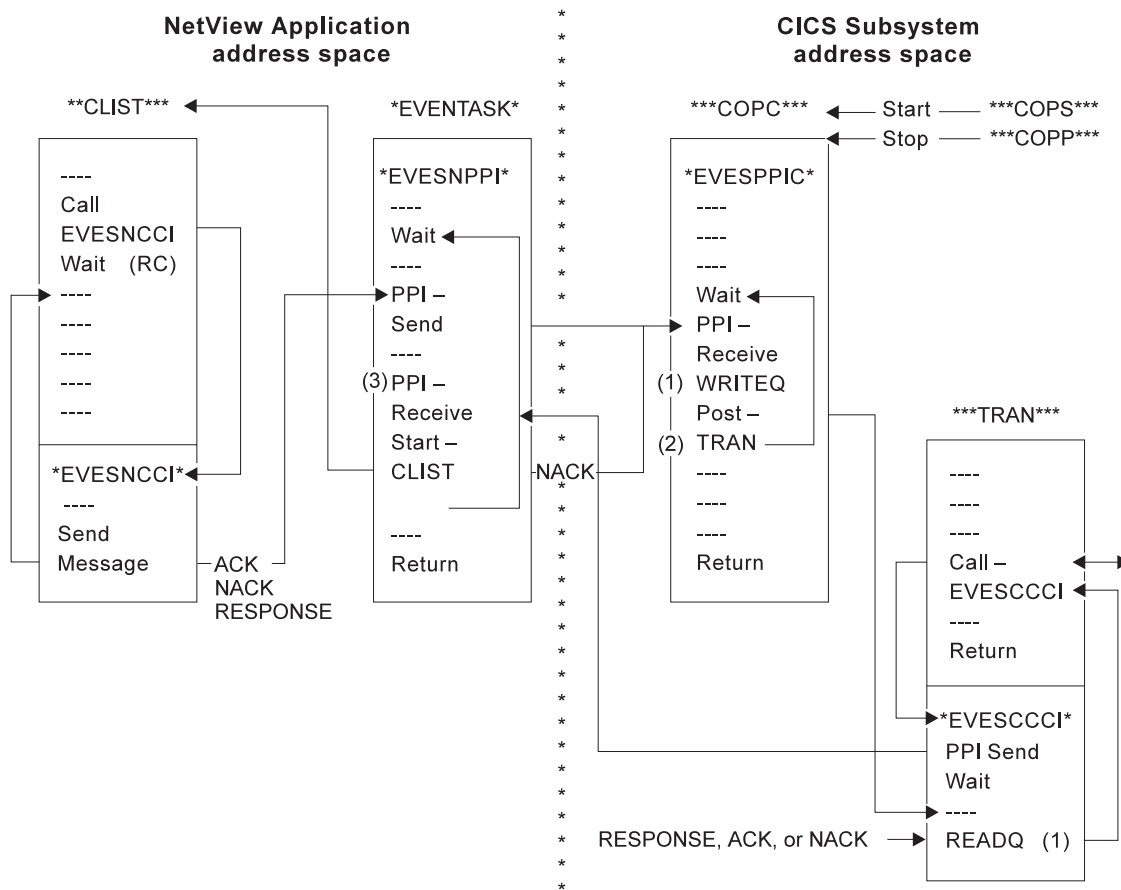


Figure 40. An EVESCCCI CONVERSE Request

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 the following figure:

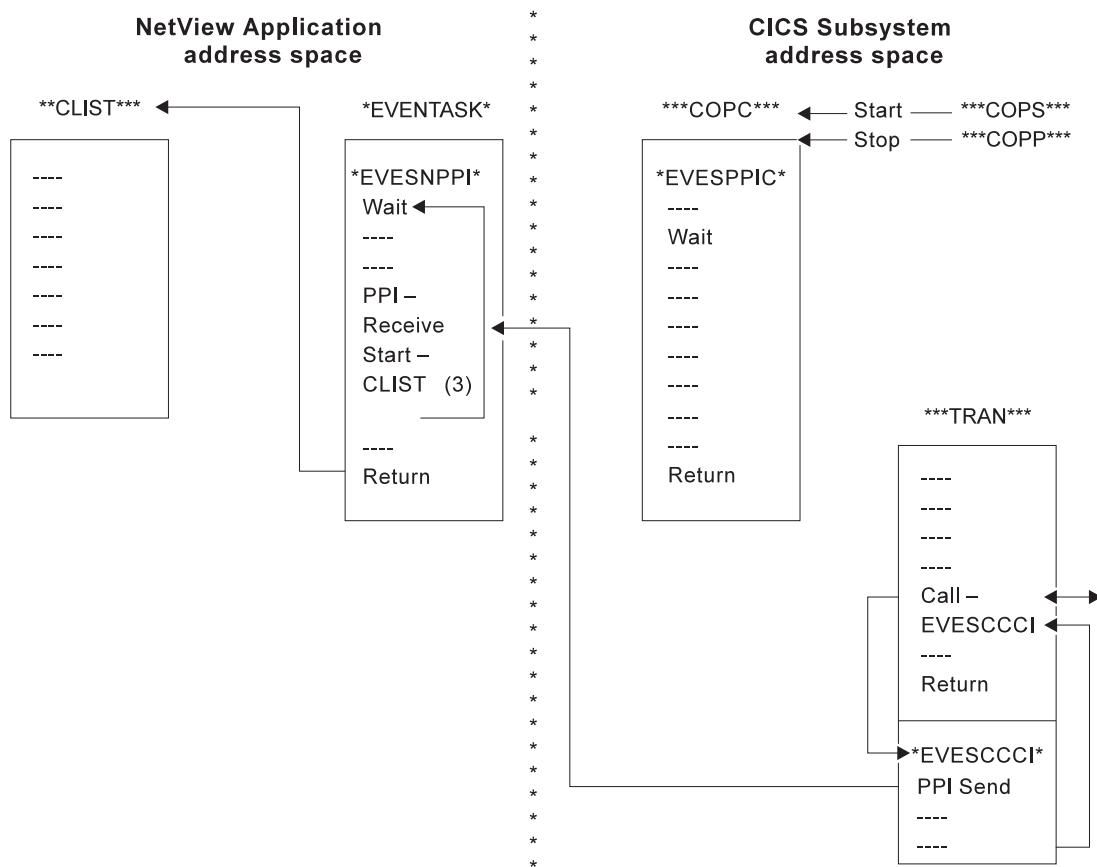


Figure 41. An EVESCCCI SEND Request

Programming Interface

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

“EVESNCCI—NetView to CICS Communication Interface” on page 146

Allows you to:

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

“EVESNRSP - Common Response Handler from CICS” on page 151

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

“EVESCCCI - CICS to NetView Communication Interface” on page 152.

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

“EVEMPINT—EVESCCI Parameter List Copy Book” on page 155.
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

EVESNCCI - NetView to CICS Communication Interface

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

SEGMENT=

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

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

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

ID=

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

Comments and Usage Notes

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

In NetView (EVENTASK)

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

In CICS (EVEMPINM)

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

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

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

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

EVESNCCI - NetView to CICS Communication Interface

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

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

Notes:

- a. SEGMENT=F only can be used with these TYPEs.
 - b. The length of the data must be less than 101 bytes.
 - c. SEGMENT=M or SEGMENT=L can only be used when 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_{xxx} REQUEST IN EVESNCCI, RC=*ccc*

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

RC=8 EVE122E EVENTASK TASK NOT ACTIVE

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

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

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

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

EVESNCCI - NetView to CICS Communication Interface

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

RC=16

EVE124E SEGMENT ERROR, CODE = *ccc*

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

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

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

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

RC=20

EVE125E NO STORAGE AVAILABLE ON DSI*xxx* REQUEST IN EVESNCCI

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

RC=24

ERROR ON EVESX*mmn* CALL IN EVESNCCI, RC = *ccc*

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

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

The following NACK responses can be expected:

EVE129E text

EVE122E *task* TASK NOT ACTIVE

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

EVE137E NETVIEW SUBSYSTEM NOT AVAILABLE

EVE141E INCORRECT MQS BUFFER RECEIVED IN *progrname*

EVE142E FUNCTION *funcname* NOT FOUND IN *membrname*

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

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

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

EVESNCCI - NetView to CICS Communication Interface

Examples of Usage

Example 1

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

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

Example 2

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

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

EVESNRSP - Common Response Handler from CICS

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

Comments and Usage Notes

1. The request data is scanned for the presence of NL (X'15') characters within the maximum response line length. If an NL character is found, it delimits the current response line. If no NL character is found, a maximum length response line is assumed, or the end of the response data delimits the current response line.
2. The following EVESX002 (CICSBMSG) commands are issued by EVESNRSP:
EVESX002 START,"*domainid,opid*
EVESX002 DATA,C,EVE790I PPI RESPONSE FROM *applid* FOR FUNCTION *function maxln totln*
EVESX002 DATA,D,*response line text*
EVESX002 DATA,E,EVE792I END"

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

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

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

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

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

EVESCCCI - CICS to NetView Communication Interface

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 10 on page 154 for a SEND request parameter list example):

Table 8. EVESCCCI CONVERSE Request Parameter List

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

Note: Sample copy books for this parameter list are included in the CICS Automation sample library. Refer to "EVEMPINT—EVESCCCI Parameter List Copy Book" on page 155 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 *nm* seconds (see OUTWAITC) for the response to arrive. If the OUTWAITC value is zero, the default wait time is 30 seconds. Valid specifications in the OUTWAITC field range from 1 up to and including 999 (seconds).

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

EVESCCCI - CICS to NetView Communication Interface

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

Table 9. EVESCCCI Fields Returned to Caller from CONVERSE Request

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

Note: Refer to “EVEMPINT—EVESCCCI Parameter List Copy Book” on page 155 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

EVESCCCI - CICS to NetView Communication Interface

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:

Table 10. EVESCCCI SEND Request Parameter List

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

Note: Refer to "EVEMPINT—EVESCCCI Parameter List Copy Book" on page 155 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 146.

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

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

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

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

EVEMPINT—EVESCCCI Parameter List Copy Book

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

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

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

EVEMPINT—EVESCCI Parameter List Copy Book

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

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

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

EVEMPINT—EVESCCCI Parameter List Copy Book

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

Examples of Usage

Example 1

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

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

Example 2

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

```

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

```

EVEMPINT—EVESCCCI Parameter List Copy Book

Glossary of CICS and Other Terms

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

abend. Abnormal end of task.

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

ACK. Acknowledgement.

APAR. Authorized program analysis report.

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

ASCII. American National Standard Code for Information Interchange.

batch. An accumulation of data to be processed.

CEC. Central Electronic Complex.

CEMT. The CICS master terminal transaction.

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

CICS. Customer Information Control System.

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

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

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

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

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

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

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

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

initialization. (1) Actions performed by CICS to construct the environment in the CICS region to enable CICS applications to be run. (2) A process started by SA OS/390 and CICS Automation to construct the environment in which automation is to occur.

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

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

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

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

IPL. Initial Program Load.

IRC. Interregion communication.

ISC. Intersystem communication.

keyword. (1) A symbol that identifies a parameter. (2) A part of a command operand that consists of a specific

character string. (3) An operand in a CEDDA definition. Key-sequenced data set—a VSAM database organization.

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

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

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

NACK. Negative Acknowledgement.

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

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

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

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

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

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

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

PTF. Program Temporary Fix.

PUT. Program update tape.

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

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

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

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

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

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

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

SIT. System Initialization Table.

SNA. Systems Network Architecture.

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

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

subsystem. (1) A secondary or subordinate system. (2) A resource defined to SA OS/390 and CICS Automation.

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

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

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

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

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

terminal operator. The user of a terminal.

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

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

update. To modify a file with current information.

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

Index

A

- ABCODEPROG 52
- ABCODETRAN 33, 54
- abend codes 33
- ACF 4
- ACK response 139
- actual state 4
- application groups 8
 - BASIC 9
 - MOVE 9
 - nesting of 9
 - SERVER 9
- application samples 20
- applications 3
 - policy items
 - CICS CONNECTION 37
 - CICS CONTROL 13, 21, 115
 - MESSAGES/USER DATA 10, 30, 45
 - MINOR RESOURCE FLAGS 30
 - RESOURCE THRESHOLDS 30, 33
 - STARTUP 18, 21, 114
 - STATE ACTION TABLE 35
- automation
 - control file 4
 - goal-driven 4
 - operators 20

B

- broadcasting messages 125

C

- CANCEL
 - from NetView 141
- CDRM recovery 17
- CEMTTPI 41
- CEMTTPI—CEMT PPI short syntax 72
- CICS Automation library 20
- CICS Automation panels
 - Broadcast Messages 125
 - CICS Monitor 135
 - Display Links 118
 - Health Checking 122
 - Link Monitoring 117
 - main menu 104
 - Messages 136
 - Monitoring 117
 - Program-to-Program Interface 130
 - search function 104
 - Set Cold Start 128
 - Set Message Id 130
 - Support Functions 127
- CICS receiver program 138
- CICS DLY—Change the shutdown delay time 73
- CICSHLTH—Linemode health checking 83
- CICSLM—Linemode link monitor 87

- CICSMHDR—Linemode message options 90
- CICSOVRD—Linemode SIT override 85
- CICSplex SM address space (CMAS) 25
- CICSplex SM REXX API
 - installing 23
- CICSPOST—Post an external event 74
- CICSPURG—Purge transactions 75
- CICSQRY—Name lookup 67
- CICSRCMD—Request a CICS Function 70
- CICSRSYC—CICS resync 76
- CICSSHUT—Shutdown processor 77
- CMASSHUT—CICSplex SM Address Space (CMAS) shutdown 92
- coexistence 28
- cold start indicator 28, 127
- commands
 - CEMTTPI 72
 - CICSDLY 73
 - CICSHLTH 83
 - CICSLM 87
 - CICSMHDR 90
 - CICSOVRD 85
 - CICSPOST 74
 - CICSPURG 75
 - CICSQRY 67
 - CICSRCMD 70
 - CICSRSYC 76
 - CICSSHUT 77
 - CMASSHUT 92
 - EVEED003 79
 - EVEEMIGR 81
 - EVEERDMP 80
 - EVEEY00S 35, 82
- consoles 18
- CONVERSE
 - from CICS 142
 - from NetView 138
- coordinating address space (CAS) 25
- COPC 13
- critical messages 29
- customization dialogs 3

D

- dependencies
 - start 4
 - stop 4
- desired state 4
- detailed status 107
- DFHRPL 20
- DISPTRG 105
- dump 80

E

- echoplexing 38
 - back-end programs 24
 - CICS target system 38
 - IMS target system 38

- entry 3
- entry type 3
- EVEED003—Critical message handler for the Status Display Facility 79
- EVEEMIGR—Migrate subsystem to CICS/TS 81
- EVEERDMP—CICS dump 80
- EVEEY00S—Common State Handler for State/Action Tables 35, 82
- EVEMPINT—EVESCCCI parameter list copy book 155
- EVENTASK 13, 22, 96
- events 7
- EVESCCCI - CICS to NetView communication interface 152
- EVESCMT3—Message exit table for CICS 24, 98
- EVESNCCI—NetView to CICS Communication Interface 146
- EVESNPPI 138
- EVESNRSP - Common response handler from CICS 151
- EVEPINM 23, 94
- EVESPPIC 138

F

- fast path 106

G

- generic routines
 - ISSUECMD 10
- goal 4
- goal-driven automation 4

H

- health checking 11, 36, 121
 - programs 23
- HEALTHCHK 56

I

- INGEVENT 41, 74
- INGLIST 128
- INGREQ 42, 113
 - Appl Params field 115
 - Override field 114
 - Type field 113
- INGSCHED 105
- ISSUECMD 10

L

- link monitoring 11, 37, 117
 - access from CICS 117
 - access from CICS Automation 117
 - display links 118
 - echoplexing 38

link monitoring 11, 37, 117 (*continued*)
 CICS target system 38
 IMS target system 38
LISTSHUT 58
local functions 39

M

main menu 104
MAT 9
messages
 broadcasting 125
 setting options 130
messages, critical 29
MESSAGES/USER DATA keywords
 ABCODEPROG 52
 ABCODETRAN 33, 54
 ACORESTART 21
 format descriptions 45
 notational conventions 47, 48
 translation rule 46
 HEALTHCHK 36, 56
 LISTSHUT 58
 RCVAUTO 59
 RCVRSOS 60
 RCVRTRAN 34, 62
 RCVRVIOL 63
MESSAGES/USER DATA policy item 45
 attributes 45
 CMD 46
 CODE 49
 others 48
migration 26
minor resources
 definitions for component
 recovery 30
MPFLSTxx 17

N

NACK response 139
NetView
 automation NetView 17
 EVENTASK 129
 network NetView 17
 Subsystem Interface (SSI) 130
NetView message automation table 9
NetView subtask program 138
notational conventions
 for CMD, REP, CODE attributes 47
 for USER type attributes 48
 general xi

P

persistence of requests 6
policy database 3
policy item 3
policy object 3
policy objects
 CICS LINK 37
 CICS STATE/ACTION 35
 MONITORING PERIOD 37
 STATUS DETAILS 29
PPI (*see* program-to-program
 interface) 13
priority of requests 4

program list table 18
program-to-program interface 13, 129
 CICS components 13
 CICS requests 142
 CONVERSE 142
 SEND 143
 communication components 13
 EVENTASK 13, 22, 96
 EVESPINM 94
 customization 23
 NetView requests 138
 CANCEL 141
 CONVERSE 138
 SEND 140
programs 19

R

RACF-protected subsystems 42
RCVAUTO 59
RCVRSOS 60
RCVRTRAN 34, 62
RCVRVIOL 63
recovery 13
 abend codes 33
 autoinstall problems 30
 minor resources 31
 RCVRTRAN 34
 short-on-storage condition 30
 storage violations 30
 thresholds 32
 transactions 31
 VTAM ACB failures 30
remote site recovery
 for VSAM RLS 41
requests 4
 conflicting 6
 persistence 6
 priority 4, 7, 8
 propagation 4
resources 3
 names, format of 3
RESPONSE response 139

S

SA OS/390 definitions for CICS 20
SA OS/390 operator commands
 DISPTRG 105
 INGEVENT 41, 74
 INGLIST 128
 INGREQ 42, 113
 INGSCHED 105
SDF (*see* status display facility) 133
SDF states 29
security checking 24, 38
segment support in NetView 147
SEND
 from CICS 143
 from NetView 140
sending messages 125
service periods 8
 service windows 8
service windows 8
setting
 cold start indicator 127
 message options 130

short-on-storage 29
SIT (*see* system initialization table) 18
start dependencies 4
startup
 types 28, 113
state
 actual 4
 desired 4
state/action tables 11, 35
 area 35
 default tables 35
 product 35
status display facility 133
 states 29
status file record
 creation for new subsystem 22
stop dependencies 4
storage violations 29
subsystems 3
 creating status file record 22
 monitoring 117
 name length 21
 RACF protection 42
 selecting 106
 shutting down 116
 starting 113
support functions
 cold start indicator 127
 program-to-program interface 129
 setting message options 130
system initialization table 25, 38
 overrides 18

T

thresholds for recovery 30
timers 29
trace function 28
transactions 19
 recovery 31
transient data messages
 EVESCMT3 24
triggers 7
 events 7
 shutdown conditions 7
 startup conditions 7

V

votes 5
VSAM RLS 41
VTAM ACM errors 29

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

System Automation for OS/390
CICS Automation
Programmer's Reference
and Operator's Guide
Version 2 Release 1

Publication No. SC33-7044-00

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Complete	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Easy to find	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Easy to understand	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Well organized	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Applicable to your tasks	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? Yes No

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

Name

Address

Company or Organization

Phone No.



Fold and Tape

Please do not staple

Fold and Tape



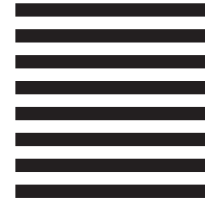
NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

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

POSTAGE WILL BE PAID BY ADDRESSEE

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



Fold and Tape

Please do not staple

Fold and Tape



Program Number: 5685-151



Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

SC33-7044-00

