CICS Transaction Server for z/OS



Troubleshooting CICSPlex SM

Version 5 Release 4

CICS Transaction Server for z/OS



Troubleshooting CICSPlex SM

Version 5 Release 4

Note
Before using this information and the product it supports, read the information in "Notices" on page 119.

This edition applies to the IBM CICS Transaction Server for z/OS Version 5 Release 4 (product number 5655-Y04) and to all subsequent releases and modifications until otherwise indicated in new editions.

© Copyright IBM Corporation 1997, 2017.
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

About this PDF. v

Chapter 1. Introduction to CICSPlex SM problem determination 1
What is problem determination? 1
How to solve CICSPlex SM problems 1
Where to look first 1

Chapter 2. CICSPlex SM system overview 5
The structure of CICSPlex SM 5
The Web User Interface 6
CMAS networks and registration 7
The structure of the CMAS 8
The agents in a MAS 9
The ESSS and data spaces 10
Common components 10
Kernel Linkage 11
Trace Services. 11
Message Services 11
Common Services 12
Data Cache Manager 12
Queue Manager 12
Data Repository 12
Communications 13

Chapter 3. Identifying a problem. . . . 15
Has CICSPlex SM run successfully before? 15
Have any changes been made since the last successful run? 15
Are there any messages that could explain the problem? 16
Does the problem occur at specific times? 16
Does the problem affect specific parts of the environment?. 16
Common types of problem 16

Chapter 4. Sources of information . . . 19
Your own documentation. 19
Change log 19
Manuals 19
Online diagnostic aids 19
Messages 20
Symptom strings 20
LOGREC records 20
Traces 21

Chapter 5. Tracing CMAS components 23

Chapter 6. Tracing MAS components 25

Chapter 7. Tools for problem determination 27
Using trace in CICSPlex SM 27
Tracing in a CMAS 27
Tracing in a MAS 27
Tracing in a WUI 28
Types and levels of tracing 28
Controlling the amount of tracing in a CMAS or MAS. 29
Formatting CMAS and MAS trace entries . . . 31
Web User Interface trace services 35
Using dumps. 37
Unexpected dumps. 37
CICSPlex SM-requested dumps. 39
User-requested dumps. 42
Using dumps with the Web User Interface . . . 42
Displaying and formatting dumps with IPCS . . . 43
Using the CICSPlex SM dump formatting routine 43
Using the ESSS utility (EYU9XEUT) 46
The EYU9XEUT options 46
The EYU9XEUT JCL 48
Using the ESSS Information Display Utility (EYU9XENF) 48
Using the online utility transaction (COLU) . . . 49
The COLU transaction. 50
Using the interactive debugging transactions (COD0 and CODB) 54
Running the debugging transactions 54
Method-level debugging with COD0 55
System-level debugging with CODB 83

Chapter 8. Investigating and documenting a problem 91
Investigating output and system management problems 91
Investigating abends 91
Investigating stalls 92
Investigating bottlenecks 93
Incomplete operations data returned 94
Missing monitor data 97
Unexpected real-time analysis results. 98
Unexpected workload management routing decision 100
Application programming interface problems 103
Investigating Web User Interface problems . . . 104
Server and web browser messages 104
COVC status panel 105
COVC debugging commands 105
Typical end-user problems 106

© Copyright IBM Corp. 1997, 2017

Chapter 9. CICSPlex SM naming standards 109
The format of names 109
Element type identifiers 109
Module prefixes 110

Chapter 10. Major components of CICSPlex SM. 113

Chapter 11. System parameters for problem determination 115
Specifying system parameters 115
The problem determination parameters 116

Notices 119

Index 125

About this PDF

This PDF describes the facilities and methods for determining the cause of problems in a system that uses CICSPlex SM. Before CICS TS V5.4, this PDF was called *CICSPlex SM Problem Determination Guide*.

For details of the terms and notation used in this book, see Conventions and terminology used in the CICS documentation in IBM Knowledge Center.

Date of this PDF

This PDF was created on June 6th 2017.

© Copyright IBM Corp. 1997, 2017

Chapter 1. Introduction to CICSPlex SM problem determination

Keep in mind that CICSPlex SM is a tool for managing the CICS® systems at your enterprise. As you investigate a potential problem in your CICS environment, be sure to distinguish between problems in managing your CICS systems and problems with the CICS systems themselves.

What is problem determination?

Usually, when you are investigating a problem, you start with a symptom, or set of symptoms, and try to trace them back to their cause. This process is called *problem determination*, and it is important to realize that it is not the same as problem solving.

Often, the process of problem determination enables you to solve the problem. For example:

- If you find that the cause of a problem is conflicting CICSPlex SM topology definitions, you can solve the problem by correcting the definitions.
- If you find that the cause of a problem is within CICS, you can solve the problem by modifying CICS. For example, if the CICSPlex SM Workload Manager will not route to a target region because there is no CICS connection between the routing region and the target region, you can create the links between the systems.

However, you might not always be able to solve a problem yourself after determining its cause. For example:

- An unexpected message might be caused by an unexpected response from another product.
- If you think the cause of a problem is in the CICSPlex SM code, you need to contact your IBM® Support Center for assistance.

How to solve CICSPlex SM problems

Start with the symptoms of a problem, then use those symptoms to classify it. For each type of problem, there are techniques you can use to determine the actual cause.

You should always assume first that the problem has a simple cause, such as a definition error. If, as a result of investigation, you find that the cause of the problem is not straightforward, then consider possible causes that may be more difficult to identify. If further investigation still does not provide an answer, it is possible that the cause of the problem is in the CICSPlex SM code itself. If this appears to be the case, you need to contact your IBM Support Center.

"Where to look first" will help you decide where to look first.

Where to look first

To find help in solving your CICSPlex® System Manager problem, determine the answers to a number of questions.

© Copyright IBM Corp. 1997, 2017

Question 1

Are you familiar with the structure of CICSPlex System Manager?

- **YES**: Go to question 2.
- **NO**: See Chapter 2, "CICSPlex SM system overview," on page 5.

Question 2

Do you think you have identified the problem?

- **YES**: Go to question 3.
- **NO**: See Chapter 3, "Identifying a problem," on page 15.

Question 3

Do you need help resolving a specific problem?

- **YES**: Go to question 7.
- **NO**: Go to question 4.

Question 4

Do you want to know more about sources of CICSPlex System Manager information?

- **YES**: See Chapter 4, "Sources of information," on page 19.
- **NO**: Go to question 5.

Question 5

Do you want to know more about tools for problem determination?

- **YES**: See:
  - "Using trace in CICSPlex SM" on page 27
  - "Using dumps" on page 37
  - "Displaying and formatting dumps with IPCS" on page 43
  - "Using the ESSS utility (EYU9XEUT)" on page 46
  - "Using the online utility transaction (COLU)" on page 49
  - "Using the interactive debugging transactions (COD0 and CODB)" on page 54
- **NO**: Go to question 6.

Question 6

Do you want to know more about contacting the IBM Support Center?

- **YES**: See IBM problem support.
- **NO**: Go to question 3.

Question 7

Is it an ABEND, STALL, or BOTTLENECK?

- **YES**: See "Investigating output and system management problems" on page 91.
- **NO**: Go to question 8.

Question 8

Is it an OUTPUT or SYSTEM MANAGEMENT problem?

- **YES**: See "Investigating output and system management problems" on page 91.
- **NO**: See Chapter 3, "Identifying a problem," on page 15.

Chapter 2. CICSPlex SM system overview

The components of CICSPlex SM work together to provide effective management of your CICS systems.

The structure of CICSPlex SM

CICSPlex SM makes use of a distributed system management architecture that is based on a manager-and-agent model.

In CICSPlex SM, the agent runs in a managed application system (MAS). The agent is in constant communication with a manager, called a CICSPlex SM address space (CMAS). This communication allows the manager to monitor and control the CICS system. The manager consolidates data from, and distributes actions to, the individual agents. The manager is also responsible for basic management applications, such as resource monitoring and workload management.

A typical CICSplex configuration would consist of many agents under the control of a single manager. In a more complex environment, there might be multiple managers, each controlling multiple agents. In order to achieve the distributed system management goal of a single-system image, these managers are normally connected to each other.

Another important aspect of distributed system management that is provided by CICSPlex SM is operation from a single point of control. In CICSPlex SM, the single point of control is the Web User Interface (WUI). The WUI runs on a standard Web browser using TCP/IP to contact a Web User Interface server running on a dedicated CICSPlex SM local MAS connected to a CMAS.

In addition to the visible parts of the CICSPlex SM system that run within existing address spaces (such as the agent code for a managed CICS system, which runs in the CICS address space), there is one largely invisible part of the system that is also an address space: Environment Services System Services (ESSS). An ESSS address space resides in each MVS™ image where a CMAS is run. The ESSS is automatically created when the first CMAS is started in a given MVS image and it remains for the life of the IPL. The ESSS provides the cross-memory services used for communication between a manager and agents when they reside on the same MVS image. It also serves as the owner of all data spaces used by the product, which enables data spaces that are shared between a CMAS and a MAS to survive the shutdown of either.

Figure 1 on page 6 illustrates the basic structure of the CICSPlex SM system.

© Copyright IBM Corp. 1997, 2017

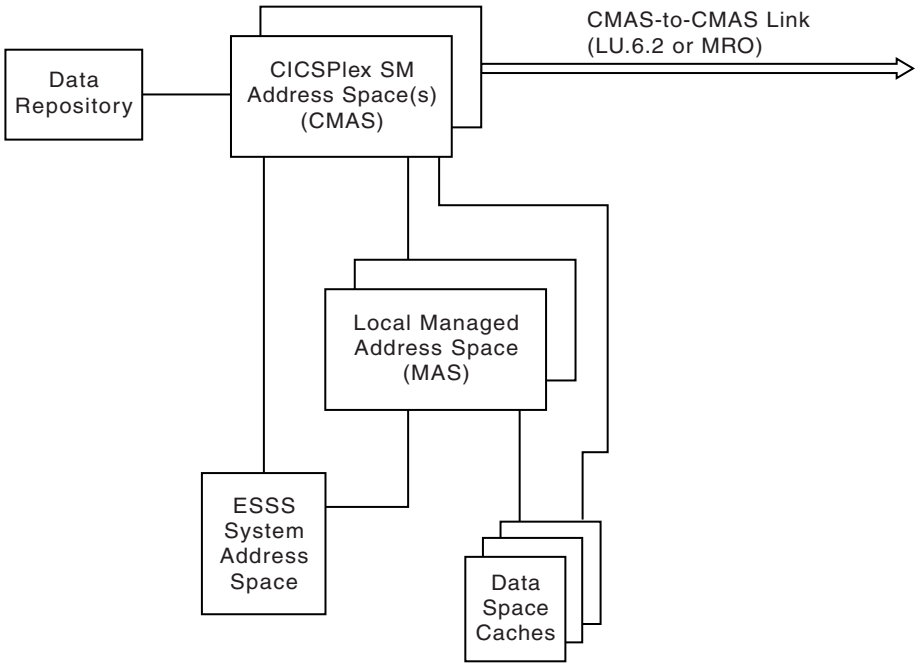


Figure 1. The CICSPlex SM system

The Web User Interface

The CICSPlex SM Web User Interface (WUI) is an easy-to-use interface that you can use to carry out the operational and administrative tasks necessary to monitor and control CICS resources. You can link to the WUI from any location that can launch a Web browser.

The WUI is accessed using standard Web browser software in contact with a dedicated CICS region acting as a WUI server. You can have more than one WUI server active; for example, you may have a requirement for different languages to be used or different systems available to different servers. The Web browser client contacts the web server by an HTTP request via the CICS Web Interface.

The WUI server runs as a CICSPlex SM local MAS and communicates with the managed resources via the CMAS to which it is connected. This CMAS needs to manage all CICSplexes that the WUI server needs to access. This is because the WUI server acts as a CICSPlex SM API application. However, it is not necessary for the CMAS, to which the WUI connects, to be managing any of the MASs in these CICSplexes.

All the menu and view definitions are stored on a server repository. There is one repository for each WUI server. The menu and view definitions can be exported for backup purposes, for distributing definitions to other servers, and for transferring menus and views when you upgrade to a new product release.

The WUI provides an administration transaction COVC that allows you to:

- Start and shutdown the WUI server
- View the server status
- View current user activity and terminate active user sessions
- Import and export repository definitions
- Set trace flags

Because a WUI server is defined as a MAS, it can be monitored using standard CICSPlex SM monitoring.

CMAS networks and registration

If more than one CMAS is involved in managing a CICSplex they must all be able to communicate with each other in order to implement single-system image.

This communication is also required to allow proper distribution of CICSPlex SM definitions from the maintenance point CMAS to other CMASs and to maintain the dynamic CICSplex topology. The maintenance point CMAS is responsible for maintaining the CICSplex definitions in the data repository as well as distributing them to other CMASs.

However, CMASs need not be fully interconnected. The CICSPlex SM communications component can deliver a request for remote processing even if the target is not directly connected to the CMAS or MAS where the request originates. The minimum requirement is that you can get from every CMAS to every other CMAS in the network via some route of CMAS-to-CMAS links, no matter how complex. Of course, performance may suffer if excessive transit nodes (those CMASs through which a request must pass on its way to the intended destination) are involved in a request. As a result, more than the minimum number of required communication links are often installed.

The CMAS provides information about the CICSplexes for which it can process requests to the ESSS address space that is running in its MVS image. This is necessary because ESSS establishes the connection between a CMAS and its local MASs. This connection is normally established when the MAS provides its name and the name of the CICSplex of which it is a member. So ESSS must be able to find a CMAS that manages the CICSplex named by a MAS.

Figure 2 on page 8 shows a sample CMAS network and the service points that result.

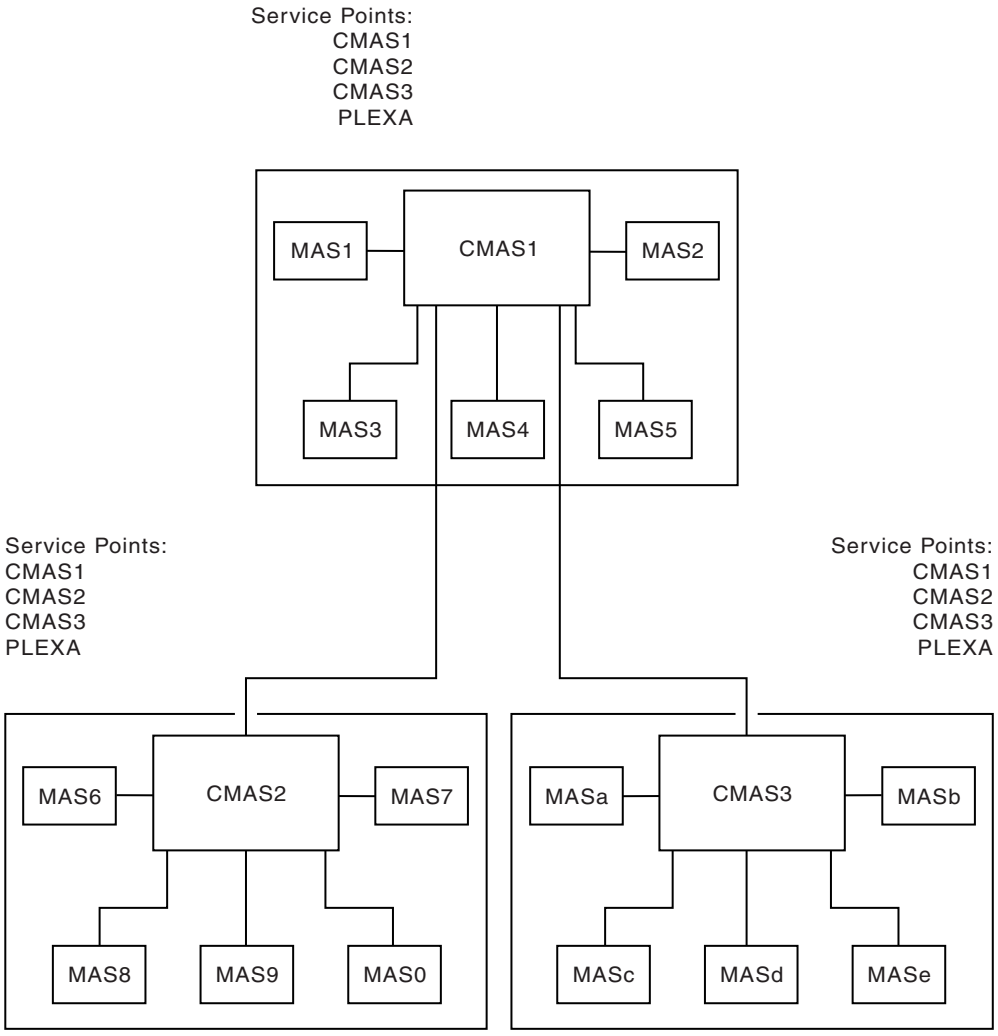


Figure 2. A sample CMAS network

The structure of the CMAS

The CMAS is a special type of CICS system.

To all the tasks that normally run in CICS, the CMAS adds a timing services task. A CMAS is started by running program EYU9XECS. This program is responsible for locating the CICSPlex SM subsystem and identifying the address space as a starting CMAS. If this is the first CMAS to start after an MVS IPL, EYU9XECS starts a CICSPlex SM subsystem. The program then transfers control to program EYU9XSTC, which is the timing services control program. After initializing, EYU9XSTC attaches DFHSIP, which is the CICS system initialization program. This starts the CICS system that runs within a CMAS.

It is possible to invoke CICSPlex SM code automatically during CICS initialization to initialize the region as a CMAS. The preferred method of initializing the region as a CMAS is to use the CPSMCONN=CMAS SIT parameter (see CPSMCONN system initialization parameter).

An alternative method is to specify programs to be run in the initialization program list table (PLTPI). For a CMAS, the PLTPI specifies program EYU9XLCS, which issues a CICS START command to start transaction XLEV. This transaction is queued to start when CICS initialization is complete.

The XLEV transaction runs program EYU9XLEV, which is responsible for creating the run-time environment for a CMAS. The component called Kernel Linkage is responsible for building data structures and controlling the interfaces between other CICSPlex SM components. Program EYU9XLEV starts the kernel linkage processing. Note that this program is used to create not only the CMAS run-time environment, but also the environment for agent code in local MASs.

The environment that Kernel Linkage creates is called the method call environment. Each program in the CICSPlex SM system is called a method. The methods in a single component are grouped and referred to as a major object, which is just another name for a component. When one method calls another method, it uses the Kernel Linkage method call services and passes parameters using a data structure called a message argument list (MAL). For each major object, there exist two primary control structures. The first, the major object descriptor block (MODB), is built by Kernel Linkage during initialization and, among other things, contains a directory of all the methods (or programs) that make up the component. The second, the major object environment block (MOEB) is pointed to from the MODB. The MOEB is created during the initialization of each component. While the format of all MODBs is the same, the MOEB for each component is unique and serves to store critical information and to anchor data used by the component.

Once the method call environment has been built, each component that requires initialization is given control to do so. Some components are merely callable services, however others are active parts of the system. Those that are active components (such as Communications, Monitor Services, real-time analysis, and Workload Manager) make calls to Kernel Linkage during initialization to start one or more tasks in the CMAS. These calls identify the method to be run. Kernel Linkage uses the method name and the MODB to determine the proper transaction ID to be used on the EXEC CICS START command. All such transactions have EYU9XLOP defined as their first program. The tasks that are started run as CICS tasks under control of the CICS system that runs within the CMAS. Each task must establish a unique run-time environment to support method processing. This environment, which runs separate from and in parallel to the other tasks, is created by a program called EYU9XLOP. This program establishes a unique copy of the environment, called an object process, and then calls the first method to be run.

After EYU9XLEV has completed the process of sequencing CMAS initialization, it enters a wait state. This wait state is broken only when EYU9XLEV must perform service functions (such as start additional CICS service tasks for the single-system image interface) or when CMAS termination is requested.

The agents in a MAS

For a CICS system to be managed by CICSPlex SM, agent code must exist and be in communication with a CMAS.

The agent code in a MAS is started in much the same way that CICSPlex SM code is started in a CMAS. A program is added to the CICS PLTPI that does a CICS START of a transaction; that transaction invokes EYU9XLEV, the same program that is used in a CMAS.

Once the MAS environment is initialized, a long running task is started that waits for requests from the controlling CMAS. Depending on the type of request received, a method call is made to process the request either synchronously or asynchronously. The long running task is also responsible for starting and stopping the other tasks involved in agent processing, such as monitoring tasks.

Another agent task is responsible for sending a heartbeat to the controlling CMAS. The heartbeat is used to let a CMAS know that the MAS is still able to communicate and to send required data on a regular basis. This data includes a current task count and the health status of the MAS.

The agent code in a managed CICS system is part of the CICSPlex SM component called the Managed Application System. This component has an identifier of MAS and its module names have the character N in the fifth position (for example, EYU0NLRT). So MAS is the identifier for both a Managed Application System (a CICS system in which CICSPlex SM agent code resides), and for the component that implements the bulk of that agent code.

The ESSS and data spaces

The Environment Services System Services (ESSS) address space is created when the CICSPlex SM subsystem is created by the first CMAS started after an MVS IPL.

The ESSS is, in MVS terms, a limited function system address space. Once it is started, it never terminates, but neither does it run. The ESSS serves as an anchor point for the data required to establish the connection between a CMAS and its local MASs. It also serves as the owner of all CICSPlex SM data spaces and cross-memory services resources. The data in the ESSS private area is updated by program call routines provided by ESSS itself. Since the ESSS does not run after initialization, it is very reliable. This reliability helps to ensure that the cross-memory resources and data spaces remain available until CICSPlex SM explicitly deletes them.

CICSPlex SM uses MVS data spaces to store some of its data structures because of the potentially large amount of data involved in managing a CICSplex environment. The size of some data structures is directly related to the number of managed systems, while the size of others is related to the number of interconnected CMASs or the system management options in use (such as real-time analysis or monitoring).

Individual data caches are used by each component that has significant storage requirements. Each logical cache can span more than one data space, but no two caches ever share a single data space. So even a simple CICSplex configuration can cause the allocation of many data spaces. While many data spaces may be created, however, CICSPlex SM uses only as much storage as is required for a given configuration.

Common components

In a system as complex as CICSPlex SM, it makes sense to have a foundation of common components on which to build so that common functions can be provided by a single component.

CICSPlex SM has many such building blocks that are used not only by all who require the service within a CMAS, but also within a MAS when the same services are required.

For a complete list of the major components of CICSPlex SM, see Chapter 10, "Major components of CICSPlex SM," on page 113.

Kernel Linkage

Kernel Linkage also has several subcomponents that provide services related to maintenance of the basic CICSPlex SM environment.

- Status Services

  Controls the synchronization between components and provides a common means for identifying the status of components.
- Notification Services

  Provides a flexible way for components to notify interested parties of events, such as the starting of a MAS. It also provides the means for components to register their interest in specific events.
- Single System Image

  Supports the distribution of requests to multiple CMASs and MASs and the consolidation of results.

Kernel Linkage also controls the interface between code running under the MVS TCBs (selectors) and code running under the CICS TCB (methods).

This component has an identifier of KNL and its module names have the characters XL in the fifth and sixth positions (for example, EYU0XLNE).

The role that Kernel Linkage plays in the transfer of control between methods was described in "The structure of the CMAS" on page 8.

Trace Services

Trace Services provides other CICSPlex SM components with the ability to write trace records to the CICS trace table and trace data sets.

Trace Services is also responsible for writing any trace records created by a MAS to the trace table and data set of the managing CMAS. Tracing is a key part of CICSPlex SM serviceability. Because a failure could occur at any time during CICSPlex SM processing, Trace Services initializes as early as possible and terminates as late as possible in CICSPlex SM processing.

This component has an identifier of TRC and its module names have the characters XZ in the fifth and sixth positions (for example, EYU0XZPT).

Message Services

The Message Services component provides a common facility for building and issuing MVS console messages. The fixed text of messages and the variable text fragments used for insertion are defined in prototype tables.

Calling methods then ask for messages by number and insert the appropriate variable text. Message Services is also responsible for creating the consolidated message log called EYULOG and for writing MAS-generated messages to the managing CMAS.

This component has an identifier of MSG and its module names have the characters XM in the fifth and sixth positions (for example, EYU0XMSM).

Common Services

The Common Services component provides basic system services such as GETMAIN, FREEMAIN, POST, and WAIT processing.

By routing all requests for these services through a single component, most CICSPlex SM modules are isolated from the real environment in which they run. As a result, a relatively few methods (those that make up Common Services) need to be aware of the details of how these services are requested. One of the Common Services subcomponents provides timing services using the control task that runs as an MVS TCB. Another subcomponent provides locking services, both local (within a CMAS or MAS) and global (between a CMAS and its local MASs).

This component has an identifier of SRV and its module names have the characters XS in the fifth and sixth positions (for example, EYU0XSCG).

Data Cache Manager

The Data Cache Manager component implements logical cache storage for use by CICSPlex SM components.

. Each component can request a cache allocation and can allocate cache blocks within it. Several additional services are also provided by the Data Cache Manager:

- A quickcell service to improve the performance of getting and freeing frequently used blocks of a fixed size.
- A comprehensive set of list manipulation services for creating and maintaining ordered lists of data.
- Support for alternate indexing of cache lists.

This component has an identifier of CHE and its module names have the characters XC in the fifth and sixth positions (for example, EYU0XCLA).

Queue Manager

The Queue Manager component implements queues of data within a cache that is shared between a CMAS and all its local MASs.

Queues are often used to communicate between different CICSPlex SM methods when the data to be passed is a set. Records within a queue can be accessed either sequentially or directly by relative record number.

This component has an identifier of QUE and its module names have the characters XQ in the fifth and sixth positions (for example, EYU0XQGQ).

Data Repository

The Data Repository component provides methods for creating, accessing, updating, and deleting data in the CICSPlex SM data repository, which is the VSAM data set where system configuration and definition data is stored.

This component provides referential integrity support for the data repository and ensures proper rollback if an operation is only partially successful. Within this component are the following subcomponents:

- The Application Programming Interface provides access to CICS system management information and enables external programs to invoke CICSPlex SM services.

- The Managed Object Services translate requests for data, for example, requests from real-time analysis, into the method calls required to obtain the data.

This component has an identifier of DAT and its module names have the characters XD in the fifth and sixth positions (for example, EYU0XDGR).

Communications

Communications is one of the most complex components of CICSPlex SM. It is made up of many subcomponents that provide all the services for implementing CMAS-to-CMAS and CMAS-to-MAS communication.

In addition to the Communications component, CICSPlex SM makes use of MVS program call routines in the ESSS. For communication between a CMAS and its local MASs, these program call routines provide cross-memory services for more efficient communication.

Communication between one CMAS and another CMAS, can use either CICS intersystem communication (ISC) or interregion communication (IRC) services (usually referred to as multiregion operation, or MRO). Because routing of messages around the CMAS network does not require the user to define path or routing information, a subcomponent of Communications maintains a dynamic topology of the network and determines routes as required.

The Communications component implements a simple model for all other CICSPlex SM components, that of remote method call. A method merely builds a MAL and invokes Communications via the Access Services subcomponent, specifying the destination and type of processing required. Communications then transports the MAL and causes it to be run in the target locations. All data required for the remote running of a method is automatically transported as well. Because all methods and their MALs are clearly defined, Communications knows what data must be sent to the target and what data must be returned to the caller. The data that is transported can be simple data in a MAL itself, data pointed to by a MAL, or CICSPlex SM queues or cache lists.

This component has an identifier of COM and its module names have the character C in the fifth position (for example, EYU0CSLT).

Chapter 3. Identifying a problem

Before you can determine the cause of a problem, you need to collect as much information as you can about your system and the symptoms you are experiencing.

The following sections raise some basic questions that will help you identify the important information.

As you go through these questions, make a note of any changes to your environment and of any unusual occurrences, regardless of whether you think they are relevant. Even if the conditions you observe do not at first appear related to the problem, information about them could be useful later if you have to carry out systematic problem determination.

Has CICSPlex SM run successfully before?

If CICSPlex SM has not run successfully before, it is possible that the system has not been installed or set up correctly.

For information on installation and setup requirements, refer to the following information:

- *CICS Transaction Server for z/OS, Version 5 Release 4 Program Directory* (or other installation instructions)

In particular, you might want to try running the installation verification procedures, (IVPs), which are described in Verifying the CICS installation in Installing. These procedures are designed to verify the correct installation of CICSPlex SM libraries and components.

Have any changes been made since the last successful run?

If CICSPlex SM has run successfully in the past, review any changes that have been made to your data processing environment since that time.

Think about your operating systems, CICSPlex SM itself, the CICS systems it manages, the hardware they run on, and any related operational procedures.

- If an APAR or PTF was applied to any of your operating systems, CICS, or CICSPlex SM, check for error messages related to the installation. Also check for any unresolved ++HOLD ACTION items associated with the SMP/E maintenance. If the installation of maintenance was successful, check with your IBM Support Center for any known APAR or PTF error.
- If a hardware modification was made, it may have affected the systems on which CICSPlex SM runs or the connectivity between systems in a CICSplex.
- If your initialization procedures changed, check for messages sent to the console during CICSPlex SM or CICS initialization. It could be that changes to JCL, CICS system initialization parameters, or CICSPlex SM system parameters are causing a problem.
- If the configuration of one or more CICSplexes has changed, check the EYULOG consolidated message log for messages describing incorrect or incompatible definitions. For example, if you are migrating additional CICS systems to

© Copyright IBM Corp. 1997, 2017

management by CICSPlex SM, ensure that the topology definitions for the new systems have been added to the CICSplex.

Are there any messages that could explain the problem?

Check to see if there were any unusual messages issued during CICSPlex SM initialization or immediately before the problem occurred.

Also check for any messages related to a CICS system that is being managed by CICSPlex SM.

If you find any messages that you don't understand, refer to the appropriate messages manual for an explanation and a recommended course of action.

Does the problem occur at specific times?

If the problem seems to occur only at specific times of the day, consider what's happening in the system at that time.

- How many MASs are active? Where are they located and how are they communicating with the CMAS that is managing them? Have any MASs or CMASs recently become active and begun communicating with other address spaces?
- Could the problem be related to system loading? Is the number of MASs (with associated resource activity) at its peak? If your CICSPlex SM environment extends across more than one time zone, remember that the time of peak system usage may vary.
- What type of monitoring, workload management, or analysis definitions are in effect? Keep in mind that the use of time periods can cause definitions to automatically become active or inactive at preselected times of the day.

Does the problem affect specific parts of the environment?

If the problem seems to affect only certain parts of the CICSPlex SM environment, consider what is unique about those parts.

If, for example, just one CMAS is experiencing a problem, review its configuration definitions:

- What system parameters were used in its startup job?
- What other CMASs does it communicate with?
- What CICSplexes does it participate in managing?

Common types of problem

Common problems include abends, stalls, bottlenecks, or problems with CICSPlex SM system-management functions.

The following list summarizes common problems:

- An abend has occurred.

  CICSPlex SM-generated console, job log, or TSO terminal messages indicate that an abend occurred and provide an abend summary.
- A stall has occurred.

  The system is not responding to users logged on (to the MAS), or the system is using an abnormally low number of processor cycles or no processor cycles.

- A bottleneck has occurred.

  The system response is abnormally slow, or the system is using an abnormally high number of processor cycles.
- CICSPlex SM system-management functions are not working as expected.

  For example, monitor or analysis definitions are not active, real-time analysis events are not occurring or are not being resolved, or a workload is being routed incorrectly.

For more details, see "Investigating output and system management problems" on page 91.

Chapter 4. Sources of information

There are a number of sources of diagnostic information.

Your own documentation

This is the collection of information produced by your enterprise about what CICSPlex SM should do and how it is supposed to do it.

It could include:

- Flowcharts or other descriptions of system processing
- Record of configuration and topology definitions
- Record of resource monitoring, real-time analysis, and workload management activity
- Trace profiles for CMASs and MASs
- Performance statistics

Change log

An up-to-date change log can identify changes made in the data processing environment that may have caused problems with your CICSPlex SM system.

For your change log to be useful in problem determination, it should include the following information:

- Changes in the system hardware
- Changes to corequisite programs (MVS and CICS)
- Changes to CICS resource definitions
- Maintenance applied to MVS
- Maintenance applied to CICS
- Maintenance applied to CICSPlex SM
- Changes in operating procedures

Manuals

In addition to this manual, you may need to refer to other manuals in the CICSPlex SM library and the libraries for related products.

Make sure that the level of any manual you refer to matches the level of the system you are using. Problems often arise from using either obsolete information or information about a level of the product that is not yet installed.

Online diagnostic aids

Assuming you can sign on to CICSPlex SM or CICS, there are several online tools for collecting data about a problem.

- CICSPlex SM views that provide diagnostic information about:
  - CMAS and MAS status
  - Resource monitoring activity

© Copyright IBM Corp. 1997, 2017

  - Real-time analysis activity
  - Workload management activity
- CICS commands that produce data similar to CICSPlex SM data.
- The CICSPlex SM online utility transaction (COLU), described in "Using the online utility transaction (COLU)" on page 49.
- The CICSPlex SM interactive debugging transactions (COD0 and CODB), described in "Using the interactive debugging transactions (COD0 and CODB)" on page 54.

Messages

Messages are often the first or only indication to a user that something is not working. CICSPlex SM writes error and informational messages to a variety of destinations.

- The system console or system log
- The CMAS or MAS job log
- The EYULOG transient data queue
- The SYSOUT data set
- A CICS terminal
- The TSO READY prompt

Messages can be issued for many different reasons:

- An inappropriate user action
- Improper product installation or setup
- An error in CICSPlex SM code

Symptom strings

Any CMAS or local MAS can produce symptom strings in a system or transaction dump. Symptom strings describe a program failure and the environment in which the failure occurred.

All CICSPlex SM symptom strings conform to the RETAIN symptom string architecture. They are stored as SYMREC records in the SYS1.LOGREC data set.

A symptom string provides a number of keywords that can be directly keyed in and used to search the RETAIN database. If you have access to the IBM INFORMATION/ACCESS licensed program, 5665-266, you can search the RETAIN database yourself. If you report a problem to the IBM Support Center, you are likely to be asked to quote the symptom string.

Although symptom strings are designed as input for searching the RETAIN database, they can also give you information about what was happening at the time the error occurred. This information might point to an obvious cause for the problem, or a promising area in which to start your investigation.

LOGREC records

LOGRECs are records containing information about an abnormal occurrence within CICSPlex SM. The records are written to the SYS1.LOGREC data set and are available for analysis after a failure.

The LOGRECs produced by CICSPlex SM all contain the same data. The data includes extensive information about the state of CICSPlex SM components in the failing address space at the time the LOGREC is written, such as:

- Identification of the failing module
- Module calling sequence
- Recovery management information

Traces

The CICSPlex SM trace facilities provide a detailed record of every exception condition that occurs. They can also be used to trace various aspects of component processing.

In CMASs and managed CICS regions, CICSPlex SM writes user trace records to the CICS trace data set, as follows:

- If any local CICS region is in communication with a CMAS, trace data is shipped from the CICS region to the CMAS, and a full, formatted trace record is produced.
- If any local CICS region is not in communication with a CMAS, either because the Communications component is not yet active or because there is a problem with Communications itself, a full, formatted trace record is produced.

Chapter 5. Tracing CMAS components

CMAS component tracing is provided for the use of IBM service personnel. CMAS trace settings are normally not active. Activating tracing may have an adverse effect on performance.

You use the **CMAS detail** (EYUSTARTCMAS.TRACE) view to control the amount of tracing that occurs in a CMAS. For information about how to access and use this view, see Using the WUI to control CMAS and MAS tracing.

© Copyright IBM Corp. 1997, 2017

Chapter 6. Tracing MAS components

MAS component tracing is provided for the use of IBM service personnel. CMAS trace settings are normally not active. Activating tracing may have an adverse effect on performance.

You use the **MASs known to CICSplex** (EYUSTARTMAS.TRACE) view to control the tracing that occurs in a MAS. To open this view:

1. Click **CICSPlex SM operations views > MASs known to CICSplex** to open the tabular view.
2. Select a CICS system in the in the **CICS system name** column and click to open the **MASs known to CICSplex** detail view.
3. At the bottom of the detail view, click **Trace details (Alter trace flag settings only when asked to by IBM System Support Center personnel)**.

.

© Copyright IBM Corp. 1997, 2017

Chapter 7. Tools for problem determination

CICSPlex SM provides a number of tools that you can use for problem determination.

Using trace in CICSPlex SM

All CICSPlex SM address space (CMAS) and managed application system (MAS) components provide trace data.

Tracing in a CMAS

The CICS internal trace facilities must always be active in a CMAS.

When a CMAS initializes, CICSPlex SM ensures that the CICS trace facility is active and the trace table is large enough. The minimum trace table settings, along with the CICS system initialization parameters that you must use, are in the following table:

Table 1. Trace table setting required by the CMAS

Trace variable	Required setting	System initialization parameter option
Internal trace	On	INTTR=ON
Trace table size	12 MB minimum	TRTABSZ=12288
Master trace	Off	SYSTR=OFF
User trace	On	USERTR=ON

If the CICS trace facilities cannot be activated with these settings, CMAS initialization is canceled and you receive an error message.

Additionally, the CICS AUXTRACE facility should be active (for user records only) in a CMAS. If this facility is not active when a problem occurs, it might be necessary to re-create the problem with the facility turned on.

Tracing in a MAS

The CICS trace facilities do not have to be active in a MAS.

Provided CICSPlex SM communication facilities are available, MAS trace records are sent to a connected CMAS for recording; the only exceptions are trace records written for the CICSPlex SM communication facility itself. If communication is not available, or if you are diagnosing a problem in the MAS, you might need to activate CICS tracing in the MAS.

1. Although it is not required, it is strongly recommended that internal and AUXTRACE facilities are active (for user records only) in a MAS. CICSPlex SM writes only exception records in a MAS, unless other trace records are specifically requested.
2. If any local MAS is in communication with a CMAS, trace data is shipped from the MAS to the CMAS, and a full, formatted user trace record is produced.

© Copyright IBM Corp. 1997, 2017

Tracing in a WUI

Use of tracing in a CICSPlex SM Web User Interface (WUI) can help to diagnose any problems encountered while using the interface.

Attention: You are recommended to activate trace only at the request of your IBM support center.

To activate WUI trace, you must specify the USERTR and SYSTR CICS system initialization parameters in your WUI server start-up job. The AUXTR CICS system initialization parameter must also be activated.

Trace records are written to the local AUXTRACE only and are not sent to the CMAS. These trace records can be formatted by the standard CICSPlex SM trace formatter, EYU9XZUT.

You can control the amount of trace information produced by the Web User Interface by setting appropriate trace flags. Thirty one independent trace flags are provided, these can be enabled from the COVC transaction or using the WUITRACE system initialization parameter in the WUI server startup job

Types and levels of tracing

Each CMAS and MAS component can make use of three types and up to 32 levels of tracing.

Standard trace (levels 1 and 2)

Standard trace points are designed to track the normal processing path of a component.

There are two levels of standard tracing, level 1 and level 2. Trace points of this type are provided by every CMAS and MAS component. However, standard tracing is not normally active because it can cause additional overhead.

Usage Note

Level 1 and 2 trace points should be activated only for a specific CMAS or MAS component and only at the request of customer support personnel.

Level 1 trace points are used for:

- Module entry and exit
- Message parameter lists

Level 2 trace points provide information to supplement a level 1 trace and they require level 1 tracing to be active for the same component. Level 2 trace points are used for:

- Major data structures, including parameter list data addresses
- Other significant events

Note: Level 1 tracing must be active in order for level 2 traces to be collected. If level 2 tracing is requested for a component where level 1 is not active, no level 2 trace data is collected.

Special trace (levels 3–32)

Special trace points can be used by a component for special-purpose traces that are unique to its situation. Each CMAS and MAS component has levels 3 through 32 available for special tracing.

These trace levels provide detailed internal information about the component. For example, trace level 16, called a timing trace, is used by some components to record how long a request took to complete. This type of trace data can be used to evaluate the performance of a component under specific conditions.

Usage Note

Level 3–32 trace points should be activated only for a specific CMAS or MAS component and only at the request of customer support personnel.

Exception trace

Exception tracing is always performed by each CMAS and MAS component when it detects an exceptional condition.

The goal of this type of trace is *first failure data capture*, to capture data that might be relevant to the exception as soon as possible after it is detected. All CMAS and MAS errors result in an exception trace entry. Exception tracing cannot be disabled and all exception trace points are always active.

Controlling the amount of tracing in a CMAS or MAS

During normal CMAS and MAS processing all the standard and special trace levels (levels 1–32) are usually disabled. Exception tracing is always active and cannot be disabled.

You can turn tracing on for a specific CMAS or MAS component in one of the following ways:

- Specify system parameters on a CMAS or MAS startup job, as described in Chapter 11, “System parameters for problem determination,” on page 115.
- Use the WUI to activate one or more levels of tracing dynamically while CICSPlex SM is running. See “Using the WUI to control CMAS and MAS tracing”
- Use the COD0 transaction TRACE flag command as described in “Method-level debugging with COD0” on page 55.

Using the WUI to control CMAS and MAS tracing

You use the **CMAS detail** (EYUSTARTCMAS.TRACE) view and the **MASs known to CICSplex** (EYUSTARTMAS.TRACE) view to control the tracing that occurs in an active CMAS or MAS.

You can access the **CMAS detail** (EYUSTARTCMAS.TRACE) view in two ways:

- From the main menu, click **CPSM operations views**.
- Either:
  1. From the **CPSM operations views** menu, click **MASs known to CICSplex**.
  2. From the **MASs known to CICSplex** (EYUSTARTMAS.TABULAR) view, click an active CMAS name to display the **CMAS detail** (EYUSTARTCMAS.DETAILED) view.
  3. Click the **Trace details** link at the foot of the view to display the **CMAS detail** (EYUSTARTCMAS.TRACE) view.
- Or:
  1. From the **CPSM operations views** menu, click **CMASs know to local CMAS**.

2. From the **CMASs known to local CMAS** (EYUSTARTCMASLIST.TABULAR) view, click the **Type of access** field for a CMAS to display the **CMAS detail** (EYUSTARTCMAS.DETAILED) view.
3. Click the **Trace details** link at the foot of the view to display the **CMAS detail** (EYUSTARTCMAS.TRACE) view.

See Table 2 for an example tabular representation of a CMAS trace flags view.

To access the **MAS detail** (EYUSTARTMAS.TRACE) view:

1. From the main menu, click **CPSM operations views > MASs known to CICSplex**.
2. From the **MASs known to CICSplex** (EYUSTARTMAS.TABULAR) view, click an active CICS system name, to display the **MASs known to CICSplex** (EYUSTARTMAS.DETAILED) view.
3. Click the **Trace details** link at the foot of the view to display the **MASs known to CICSplex** (EYUSTARTMAS.TRACE) view.

Note: The MAS trace flags view is similar to that shown in Table 2. However, the MAS trace flags view has MAS services trace flags and does not have Monitoring trace flags.

Table 2. Example of CMAS trace flag settings

Trace Flags	
Business Application Services (BAS) trace flags	5, 7, 9, 11, 15, 17-25, 27, 29, 32
Cache services trace flags	5-12, 18, 24-28
Communications trace flags	
Data repository services trace flags	5-19, 21-24
Kernel linkage trace flags	4-7, 9, 11-12, 15, 17, 19-20, 26-29
Monitoring trace flags	
Message services trace flags	30-32
Queue services trace flags	12-19, 21, 25, 29
Real time analysis (RTA) trace flags	1, 5, 7, 9-11, 18, 21, 23, 28-31
External services trace flags	
Topology trace flags	
Trace services trace flags	
Workload management trace flags	3-11,14, 18, 20, 24, 26-28, 30

To change the trace settings for a specific component, such as Kernel Linkage:

1. Click in the box on the Kernel linkage trace flags line.
2. Edit the trace flags. The syntax rules are given in “Trace flag syntax”
3. When the trace settings are correct, click the **Apply changes** button.

Trace flag syntax:

Specify the trace flags as a list of discrete bit numbers and sequences of bit numbers, in the range 1-32. Use commas to separate the items in the list.

For example:

```
1, 5-7, 12, 14, 17-32
```

When you specify trace flags, observe the following rules:

- You can specify bit numbers in the range 1-32.
- You can insert spaces before the first item in the list, and before and after a comma.
- When you specify a sequence, the first number specified must be less than the second. For example, you can specify `1-3` but not `3-1`.

Interpreting CMAS and MAS trace entries

A single CMAS or MAS trace can produce multiple records. Each record consists of a standard header followed by up to 3900 bytes of unique trace data.

Within that data, each CMAS and MAS component uses a unique set of trace point IDs. Each trace point ID is used by only one trace point. A trace point ID consists of:

- Component ID
- Method ID
- Trace point number

Trace point numbers are assigned as follows:

Range Type of trace

0001–1024
Exception trace

1025–2048
Level 1 trace

2049–3072
Level 2 trace

3073–32767
Special trace (Levels 3–32)

Formatting CMAS and MAS trace entries

The CICSPlex SM trace format utility, EYU9XZUT, formats the raw trace records produced for a CMAS or MAS.

The host version of EYU9XZUT, formats the AUXTRACE records produced for a CMAS or MAS (including trace records sent to the CMAS).

Trace formatting options on the host

The EYU9XZUT trace format utility has options that allow you to select the specific trace records to be formatted.

You specify the formatting options you want to use on the SYSIN statement of the program's JCL, as described in “Trace formatting JCL” on page 33.

When no options are specified, all trace records in the trace data set are formatted.

EYU9XZUT supports the following options:

ABBREV
Provides an abbreviated trace, which has one line per trace record with a sequence number at the far right. Use the sequence number to select full trace formatting of specific records.

The abbreviated trace is written to a SYSOUT file named TRCEABB. You must provide a DD statement for this file when you request an ABBREV trace. If you do not provide the DD statement, an error message is produced and processing stops.

COMPID=xxx,...|ALL
Specify the three-character identifier of the components whose trace entries you want to format, or ALL for all CICSPlex SM components. For a list of component identifiers, see Chapter 10, "Major components of CICSPlex SM," on page 113.

EXCEPTION=ONLY|ALL
ONLY formats only those exception trace records that match all other criteria. ALL formats all exception trace records, as well as any other trace records that match all other criteria.

FULL Provides full trace formatting of trace records meeting all selection criteria.

The trace is written to the SYSOUT file named TRCEOUT. You must provide a DD statement for this file when you request a FULL trace. If you do not provide the DD statement, an error message is produced and processing stops.

METHOD=xxxx,...|ALL
Specify the four-character identifier of specific methods whose trace entries you want to format, or ALL for all the methods for a component.

If the trace entries for one or more specific methods are required, customer support personnel will provide you with the appropriate method IDs.

NAME=
Specify the 1- to 8-character name of a CMAS or MAS whose trace entries you want to format.

The name appears on the trace heading, following the heading NAME.

RECOVERY=ONLY|ALL
ONLY formats only abend trace records, regardless of any other criteria that may be specified. ALL formats all abend trace records, as well as all trace records that match any other specified criteria.

SEQ= Specify one or more sequence numbers to select specific trace records.

The sequence number for each trace record appears at the far right of the formatted trace heading. Sequence numbers can be from 1 to 9 characters in length. A sequence number of zero is not valid.

Sequence numbers can be specified as a single entry or as a range of entries separated by a hyphen. For example:

```
SEQ=1-99,103,12345-12399
```

You can use up to 50 SYSIN cards with the SEQ= option. Each SYSIN data set can have up to 200 specific sequence entries, as either individual numbers or ranges. Any additional entries are ignored.

If you rerun the trace format utility using SEQ=, in order to get the same trace records you must specify all of the same options that you specified on the first run.

SORT=TIME | TASK
Specifies whether TIME (defsult), or TASK ID, is the major sort key when the output is formatted.

TASKID=tasknum1,tasknum2,tasknum3...
Specify the task ID number for each task for which you want trace records. Up to 20 task IDs can be specified.

TRANID=trn1,trn2,trn3...
Specify the transaction ID of each transaction for which you want trace records.

The transaction ID appears in the formatted trace header, after TRANID.

USER=
Specify a TSO user ID.

The TSO user ID appears in the formatted trace header, after USER. Note that the USER= option is valid only for records that include an end-user interface unit of work.

You can request both an ABBREV and a FULL trace formatting in one run, by including both keywords in your SYSIN file and including the appropriate DD statements in the JCL.

Hierarchy of formatting options:

The combination of trace formatting options you select affects the output you receive.

When you select:

COMPID or METHOD
Records for the specified component or method are printed.

USER or NAME
Records for the specified TSO user or system are printed.

(COMPID or METHOD) and (USER or NAME)
Only those records for the specified component or method that are also associated with the specified TSO user or system are printed.

EXCEPTION
When you specify ALL, all exception records are printed regardless of the other options you specify.

When you specify ONLY, exception records are printed for only the specified component, method, name, or user.

SEQ Selected records are printed, depending on the sequence of records you specify.

TRANID
Selected records are printed, depending on what you specify for all other options.

Trace formatting JCL

This is an example of the JCL needed to run the host version of EYU9XZUT trace format utility.

```
//jobname  JOB (acct),'name',CLASS=x,MSGCLASS=x
//TRCLST  EXEC PGM=EYU9XZUT,REGION=2048K,PARM='NARROW'
//STEPLIB  DD  DSN=CICSTS54.CPSM.SEYULOAD,DISP=SHR
//SORTWK01 DD  SPACE=(CYL,(3,2)),UNIT=SYSDA
//SYSPRINT DD  SYSOUT=*
//SYSOUT   DD  SYSOUT=*
//TRCEIN   DD  DSN=cics.system.DFHTRACA,DISP=SHR
//         DD  DSN=cics.system.DFHTRACB,DISP=SHR
//TRCEOUT  DD SYSOUT=*,COPIES=1
//TRCEABB  DD SYSOUT=*,COPIES=1
//SYSIN    DD  *
ABBREVIATED
FULL
COMPID=MON
EXCEPTION=ONLY
METHOD=MSIN
RECOVERY=ALL
SEQ=1-55,77,999-1234567
TRANID=TRN1
/*
```

Figure 3. Example of JCL to execute the EYU9XZUT trace format utility

Note:

1. The `PARM='NARROW'` parameter on the TRCLST EXEC statement causes the trace records to be printed in an 80-character format for display on a terminal. If you omit this parameter, the trace records are printed in their normal 132-character format.
2. The data set specified by the `TRCEIN DD` statement is the CICS auxiliary trace data set from a CMAS or a MAS.

```
                    CVM.CICS.CVMSM2.DFHTRACB
            CPSM Selective Trace Format Parameters:
            ABBREVIATED=YES
            FULL REPORT=YES
            EXCEPTION=ALL
            RECOVERY=ALL
            TRANID=ALL
            COMPID=WLM
            METHOD=XCBA
            NAME=ALL
            USER=ALL
            SEQ=ALL
EYU9XZUT - CICSPlex SM Trace Formatter
 PROCESSING DATASET:CVM.CICS.CVMSM2.DFHTRACB
TASK:00034  METHOD:XQLK   PRIOR:XQGQ  DEBUG:QLOKEXC   POINTID:     2  TRANID:LP
            MAJOBJ:QUE    ENVRN:CMAS  TYPE:EXCEPTION  TOD:08:32:16.80730    CLO
            UOW(CPSM):    SYSTM:MVSH  NAME:CICSCMH    CICS-TASK:  34  TASK-STAR

        MethName  XQLK,XQGQ,CPTI,CPLT,XLOP


        MAL        LEN:0030       ALET:00000000 ADDR:0502DBD0
            EYUQXQLK IN
               *ENM FUNCTION( QUELOOK )
                CHR DEBUG(          )
               *QID QTOKEN( 800C500600001920 )
                SDT DELETE( FALSE )
            OUT
               *ENM RESPONSE( INVALID )
               *ENM REASON( QUEUE_ID_NOT_FOUND )
               *EPT CACHE_TOKEN( A= 00000000 O= 00000000 )


   PROCESSING DATASETS: CVM.CICS.CVMSM2.DFHTRACA

TASK:00042  METHOD:XSWX  ENVRN:CMAS TYPE:*ABEND* TRAN:MCCM TOD:05:24:31.48816

        ABEND CODE: AEXY
               PSW: 00000000 00000000
            OFFSET: 6F6F6F6F
            METHOD: XSWX
          INTERUPT: 00000000 00000000
           PROGRAM: EYU0XSWX
                    R04  06103D90 R05  00000008 R06  06104022 R07  06A37460
                    R08  06A36388 R09  05E64918 R10  05E65918 R11  06A3648C
                    R12  06A36080 R13  80045578 R14  85E6599E R15  85809080
      ABEND ARREGS: AR00 00000000 AR01 00000000 AR02 00000000 AR03 00000000
                    AR04 00000000 AR05 00000000 AR06 00000000 AR07 00000000
                    AR08 00000000 AR09 00000000 AR10 00000000 AR11 00000000
                    AR12 00000000 AR13 00000000 AR14 00000000 AR15 00000000
     ABEND STORAGE: -10  00000000 00000000 00000000 00000000
                    +00  00000000 00000000 00000000 00000000
```

Figure 4. Example of output from the EYU9XZUT trace format utility

Web User Interface trace services

The CICSPlex SM Web User Interface (WUI) provides a trace service to help diagnose any problems encountered while using the interface.

Attention: It is recommended that you activate trace only at the request of your IBM support center.

To activate WUI trace, you must specify the USERTR and SYSTR CICS system initialization parameters in your WUI server start-up job. The AUXTR CICS system initialization parameter must also be activated.

Trace records are written to the local AUXTRACE only and are not sent to the CMAS. These trace records can be formatted by the standard CICSPlex SM trace formatter, EYU9XZUT.

You can control the amount of trace information produced by the Web User Interface by setting appropriate trace flags. Thirty one independent trace flags are provided, these can be enabled from the COVC transaction or using the WUITRACE system initialization parameter in the WUI server startup job

Setting trace flags using the WUITRACE parameter

You use the WUITRACE parameter to set WUI trace flags.

For example, to activate trace levels 13,15, and 31 when the Web User Interface is started, specify the WUITRACE parameter as follows:

```
WUITRACE(13,15,31)
```

Setting trace flags through COVC

You can use the COVC transaction to set CICSPlex SM trace flags.

Run the COVC transaction and select **Trace Flags**. You are presented with Figure 5.

```
COVC                CICSPlex SM Web User Interface Control            EYUVCTT

                                Trace Flags

Overtype the trace, enter to update.

                              1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3
             1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Trace Flags : Y Y N N N N Y N N N Y N Y N Y N N N N N N N N N N N N N N Y




Aux. Trace Status : Started

Current Status : Ready                                      Time : 14:24:36
Applid         : IYCQCTA5                                   Date : 02/27/2001



PF   1 Help      3 Exit                                       12 Return
```

Figure 5. Trace flags

Figure 5 shows that trace levels 1,2, 7, 11, 13,15, and 31 are active. You can overtype any of the trace flags with Y or N to change the default settings.

Note: If you change any of the settings using the COVC transaction then restart the Web User Interface, the trace levels will be reset according to the levels specified on the WUITRACE parameter.

The available trace flags

Each Web User Interface trace level has a specific use.

Table 3 lists some available trace levels and their usage.

Table 3. Trace levels and their usage

Level	Usage
1	Method entry and exit (summary)
2	Method entry and exit (detail)
3	Method entry and exit (special)
7	Stack management
8	Storage management
9	Service queue processing
10	Service event processing
11	View Editor
12	Resource catalog
13	Stub management
14	View cache
15	HTTP requests and responses
18	API command
19	Data formatting
31	Soft exceptions

Exception traces

Exception tracing is always performed by the Web User Interface server when it detects an exceptional condition.

The goal of this type of trace is first failure data capture, to capture data that might be relevant to the exception as soon as possible after it has been detected.

Exception tracing cannot be disabled and all exception trace points are always active.

First-failure data capture is provided in two ways, as follows:

- Unexpected CICS and CICSPlex SM responses and other detectable error conditions will result in non-maskable trace records being written.
- Errors such as program checks and abends will also result in a system dump and abend operator messages.

Using dumps

CICSPlex SM can produce several types of dump.

Unexpected dumps

Because CICSPlex SM has a presence in two major parts of your environment, MVS and CICS, unexpected dumps may be produced at either level.

CICSPlex SM dumps under CICS

CICS causes a dump to be taken for a CICSPlex SM component when an abend occurs in a CMAS or MAS.

When an unexpected abend occurs under CICS, CICSPlex SM writes an abend indication and summary to the console and job log. The first message in the abend summary is usually:

```
EYUXL0900I Starting environment recovery
```

CICSPlex SM also writes a summary record to the CICS trace data set and takes a transaction dump, if appropriate. In addition, if the abend occurs in a CMAS or local MAS, CICSPlex SM produces SYMREC records and, when appropriate, takes an SDUMP.

Figure 6 is an example of a CICSPlex SM dump produced under CICS.

```
+EYUXL0900I   Starting Environment Recovery
+EYUXL0905E CICSCMH  ASRB IN MCCD, OFFSET 000003D0 PSW=078D4000 8818A880 LEVEL=JCI690M PFX=EYU
+EYUXL0905E INTC=0028 ILC=6 TXCP=0550D000 SCODE=S00E0 TRAN=MCCM TASK=0000041
+EYUXL0905E Methods=MCCD,MCCM,XLOP
+EYUXL0906I Registers at ABEND
 EYUXL0907I GPR0-GPR3  05402EB8  05401178  00001FA8  0818A4F0
 EYUXL0907I GPR4-GPR7  05401EB8  050271B0  0000000C  05400F10
 EYUXL0907I GPR8-GPRB  003BE000  0547E6D8  053DC40C  0818B4F0
 EYUXL0907I GPRC-GPRF  05400C88  05400F10  D8C3D900  07FD91E8
 EYUXL0907I ARR0-ARR3  00000000  00000000  00000000  00000000
 EYUXL0907I ARR4-ARR7  00000000  00000000  00000000  00000000
 EYUXL0907I ARR8-ARRB  00000000  00000000  00000000  00000000
 EYUXL0907I ARRC-ARRF  00000000  00000000  D4D6E2D5  00000000
+EYUXL0908I Storage At ABEND
 EYUXL0909I -20  337E4199  00104660  336847F0  39C89AEE
 EYUXL0909I -10  900058E0  9004B219  0200D203  D604E018
 EYUXL0909I +00  5810D604  88100001  5010D608  B2190000
 EYUXL0909I +10  D203D5FC  40105860  D55C9140  D5F047E0
+EYUXL0910I EYU9XLRV Dump,CICSMH  ,CICSMH  ,MVSH,CMAS,MCCM,0000041,
            ASRB,EYU0MCCD,08/26/95,09:55:07
+EYUXL0999I XLRV Exiting Successfully
```

Figure 6. Sample CICS abend indication and summary

Each CICS SDUMP has a title that consists of a summary of the abend. The title includes:

- The name of the recovery routine that requested the SDUMP
- The MVS jobname
- The name of the CMAS or local MAS (as known to CICSPlex SM)
- The 4-character MVS system ID
- The environment (CMAS or MAS)
- The CICS transaction ID
- The CICS task number
- The CICS abend code
- The full name of the CICSPlex SM method that abended
- The date and time of the abend

Here is an example CICS SDUMP title:

```
EYU9XLRV Dump,CICSCMH ,CICSCMH ,MVSH,CMAS,MCCM,0000041,ASRB,
EYU0MCCD,08/26/98,09:55:07
```

In this example:

Name of the recovery routine that requested the SDUMP:	EYU9XLRV
MVS jobname:	CICSCMH
Name of the CMAS or local MAS:	CICSCMH
4-character MVS system ID:	MVSH
Environment (CMAS or MAS):	CMAS
CICS transaction ID:	MCCM
CICS task number:	0000041
CICS abend code:	ASRB
Full name of the CICSPlex SM method that abended:	EYU0MCCD
Date and time of the abend:	08/26/98 09:55:07

TRANDUMP and SYSDUMP code entries in a MAS:

When the CICSPlex SM local MAS agent starts, it automatically adds one CICS TRANDUMPCODE (TRANDUMP) entry for transaction dump code EYUN and two SYSDUMPCODE (SYSDUMP) entries for system dump codes EYU0XZPT and EYU0XZSD.

When the CICSPlex SM local MAS agent starts, it automatically adds one CICS TRANDUMPCODE (TRANDUMP) entry for transaction dump code EYUN and two SYSDUMPCODE (SYSDUMP) entries for system dump codes EYU0XZPT and EYU0XZSD. These codes are primarily used for CICSPlex SM Web User Interface users who many want to use the ADD action from the EYUSTARTTRANDUMP and EYUSTARTSYSDUMP view sets to add their own TRANDUMP or SYSDUMP entries.

The presence of these entries does NOT indicate a problem, unless the current dump count for these entries is greater than zero, in which case the MAS job logs and dump may need to be investigated.

CICSPlex SM-requested dumps

There are 3 types of CICSPlex SM-requested dump.

- A CMAS initialization failure
- A MAS initialization failure
- An abend in an Environment Services System Services (ESSS) program call (PC) routine

CMAS initialization failures

If an abend occurs during CMAS initialization, the CMAS terminates.

CICSPlex SM takes an SDUMP with a dump code of EYUXL001 and writes a failure summary to the job log and console.

MAS initialization failures

If an abend occurs during MAS initialization, the MAS agent code terminates; the CICS system continues to initialize, but it is not known to CICSPlex SM.

CICSPlex SM takes a transaction dump with a dump code of EYUK and writes a failure summary to the job log and console.

ESSS program call (PC) routine failures

If an abend occurs while a CICSPlex SM PC routine is executing, the functional recovery routine (FRR) takes an SDUMP.

The format of the title is as follows:

```
CICSPlex SM (rrrr) Abend,(PC Set Name),(PC Routine Name),
(Job Name),(SID),(date),(time)
```

where:

rrrr Is the release of CICSPlex SM

PC Set Name
Is the descriptive name of the set of PC routines that encountered the error. It can be one of:

- Communication Services
- Dataspace Management
- Environment Services
- Lock Management
- MAS Assist Services

PC Routine Name
Is the name of the PC routine within the set

Job Name
Is the MVS Jobname

SID Is the MVS System ID

date Is the date in the form `MM/YY/DD`

time Is the time in the form `HH:MM:SS`

For each PC Set Name, the PC Routine Names are as follows:

Communication Services

- ADDTHRD
- BINDAPI
- BINDEICB
- EADDTHRD
- EREMTHRD
- POSTECB
- REMTHRD
- SETAPI
- SETICT

Dataspace Management

- CREATEDS
- DCMDS_INFO
- DELETEDS
- EXCELETE
- EXDCMDS_INFO
- EXDELGBL
- EXDELLCL
- EXEXTEND

- EXCREATE
- EXINFODS
- EXRELEASE
- EXTENDDS
- INFODS
- RELEASEDS

Environment Services

- APOTASK
- AUTHORIZE
- BIND
- CONNECT
- EAPITASK
- EXLCMAS
- EXLSIG
- EXRSIG
- FREE
- GSIGNAL
- IDENTIFY
- INQUIRE
- LISTCMAS
- LSIGNAL
- QUERY
- REGISTER
- RSIGNAL
- TERMINATE
- UPDPLEX

Lock Management

- ACQUIRE
- ADDLOCK
- EXACQLOCK
- EXADDLOCK
- EXRCVLOCK
- EXRELLOCK
- EXREMLOCK
- EXREMWAIT
- RCVLOCK
- RELEASE
- REMLOCK
- REMWAIT
- SSRCVLOCK
- SSREMLOCK

MAS Assist Services

- EMASINQ
- MASINQ

After it takes the SDUMP, the PC routine returns to its caller with a return code indicating that an abend occurred during processing.

User-requested dumps

You can request a dump of a CICSPlex SM address space at any time using the MVS DUMP command.

Using the MVS DUMP command

You can issue the MVS **DUMP** command from the console to dump an Environment Services System Services (ESSS) address space, a CMAS, or a MAS. Use the ASID= keyword to identify one or more address spaces and the DSPNAME= keyword to request data space dumps.

If you request a dump of data spaces, you must also dump the DMDS SYSID data space and the ESSS address space, because that component owns all CICSPlex SM data spaces. Data space names take the form:

```
cmpnsysid
```

where:

cmp Is either the three-character identifier of the component that uses the data space or DMDS, for the data cache master data space, which has controlling information for all data spaces. For a list of component identifiers, see Chapter 10, “Major components of CICSPlex SM,” on page 113.

n Is the sequential number of a component data space.

sysid Is the four-character system ID of the associated CMAS.

Note: You can use the MVS **DISPLAY ACTIVE** command to display the ASID of the ESSS address space (EYUX530) and the names of data spaces.

A sample dump command for a CMAS might look like this:

```
DUMP COMM=(CMAS DUMP)
R xx,ASID=(2A,55),CONT
R xx,DSPNAME=(55.DMDSHTC1,55.QUE1HTC1),END
```

Using dumps with the Web User Interface

The CICSPlex SM Web User Interface produces system dumps for all undetectable error conditions, including ASRA and AICA abends, but not for transaction dumps.

Attention: The Web User Interface server controller transaction COVC, should be used for debugging only at the request of your IBM support center. You must take steps to ensure that this transaction is used only by authorized personnel because of the access to system control areas that it provides. Improper or unauthorized use of COVC may have serious consequences, including without limitation loss of data or system outage. Customers are solely responsible for such misuse.

The available dump codes

The Web User Interface uses four dump codes.

Table 4. WUI dump codes

Dump code	Description
EYU0VWAN	Unexpected abend in Web User Interface CICS Web Interface analyzer program (EYU9VWAN)
EYU0VWCV	Unexpected abend in Web User Interface CICS Web Interface converter program (EYU9VWCV)
WUIABEND	Unexpected abend in Web User Interface server program (EYU9VKEC)
WUITRACE	Dump requested by Web User Interface server trace point in program (EYU9VKEC)

Displaying and formatting dumps with IPCS

The interactive problem control system (IPCS) provides you with an interactive facility for diagnosing software failures.

You can either view the dumps at your terminal or print them. Workstation dumps cannot be handled by IPCS.

CICSPlex System Manager provides a dump formatting routine that you can use with the **VERBEXIT** subcommand to format CMAS or MAS dumps.

For more information about IPCS, see the *z/OS MVS Interactive Problem Control System IPCS) User's Guide*. To format system dumps using IPCS, see Dump utilities (DFHDU680 and DFHPD680).

Using the CICSPlex SM dump formatting routine

You can use the CICSPlex SM dump formatting routine with the IPCS VERBEXIT command to analyze an SDUMP taken for a CMAS or MAS.

The formatting routine lets you process a dump selectively by identifying one or more CMAS or MAS components as parameters to the exit.

The routine is supplied as module EYU9D530, but can also be identified to IPCS as CPSM530 when member CICSTS54.CPSM.SEYUPARM(EYUIPCSP) is installed. You can specify either name with the VERBEXIT command.

Usage Notes

- This dump formatting routine should be used only at the request of customer support personnel.
- If you are asked to send a copy of an SDUMP to support, you must send the unformatted dump.
- To distinguish between problems in the MAS agent code and problems in the underlying CICS system, support personnel may also ask you to format a MAS dump using the CICS DFHPDnnn formatting routine. For more information about this routine, see Dump utilities (DFHDU680 and DFHPD680).

Formatting a CICSPlex SM SDUMP

You use the VERBEXIT command to format an SDUMP.

The syntax of the VERBEXIT command is as follows:

```
VERBEXIT CPSM530 'JOB=jobname,compid1,...,compidn,option,DLCT=nn...n,QID=nn...n'
```

where:

JOB= Identifies which CICSPlex SM address space in the dump is to be formatted. If this parameter is omitted, the first CICSPlex SM address space found is formatted.

If no additional parameters are specified, the formatting routine does the following:

- Locates the selected address space.
- If the address space is a MAS, displays the CICS exit processing block.
- If DMDSxxxx data spaces are found in the dump, attempts to create equate symbol records (ESRs) for the data cache list tables (DCLTs) and all CPSM ALET values.

The ESRs created by this routine can be used to display data space storage by the ALET value and offset found in any CICSPlex SM control block. ESRs for the ALETs are in the form `EYURECnnnnnnnn`, where `nnnnnnnn` is the ALET value. To browse storage, use standard IPCS commands, such as `L EYURECnnnnnnnn+yyyy`, where `yyyy` is the offset into the data space.

compid1,...,compidn

Identifies one or more specific CICSPlex SM components for which dump data is to be formatted. If no component IDs are specified, only the CICSPlex SM Kernel Linkage anchor block (XLWA) is formatted. For a list of component identifiers, see Chapter 10, “Major components of CICSPlex SM,” on page 113.

For individual components, you can control the level of information that is produced by specifying `compid=n`, where `n` is one of the following:

1 Displays summary information, including a summary of CICS tasks for the component.

Note: For the Trace Services (TRC) component, this option formats only the exception trace records.

2 Displays detailed information, including the MODB, the MOEB, and all kernel linkage information for the component.

3 Displays both summary and detailed information for the component.

If no level is specified, both summary and detailed information are produced (or, in the case of Trace Services, all trace records are formatted).

option Requests additional non-component related information. The following options are supported:

ESSS Displays the ESSS address space control blocks.

LOCKS
Displays a summary of resource locks.

QLIST
Displays a summary of all allocated queues. Requires that the QUE dataspace(s) be present in the dump.

TASKS
Displays a summary of CICS tasks for all CICSPlex SM components.

DCLT=nn...n
Identifies the DCLT to be displayed. DCLT identifiers, which are defined by the CPSM530 formatting routine as ESRs, can be from 12 to 16 bytes in length. (When the DCLT identifier is less than 16 bytes long, leading zeroes are assumed).

The DCLT control block and all elements associated with the DCLT are displayed. Each DCLT element is displayed as a separate block of storage.

Note: Both the data cache data space (DMDSxxxx) and the component data spaces containing cache list elements must be included in the dump for this routine to work properly. If the dump was produced by CICSPlex SM as the result of an abend, the required data spaces may not be present. If, however, the data spaces are explicitly requested by a user, rather than by CICSPlex SM, the processing should complete successfully.

QID=nn...n
Formats the selected data queue, showing the data queue service blocks, data queue record locate blocks, and the data queue record areas. The value `nn...n` is the 16-character data queue identifier.

The following is an example of a VERBEXIT command used to format dump data for specific components of a CMAS:

```
VERBEXIT CPSM530 'JOB=EYUCMS1A,TOP,RTA,MON=1,ESSS'
```

In this example, the address space to be formatted is EYUCMS1A. Dump data is produced for the Topology Services (TOP), real-time analysis (RTA), and Monitor Services (MON) components. For the Monitor Services component, only summary information is displayed. In addition to the component information, the ESSS control blocks are displayed.

CICSPlex SM SDUMP summaries:

Submitting the CPSM530 VERBEXIT to run in background produces summaries.

These summaries are as follows:

- Control block index, sorted by:
  - Area ID
  - Address space ID
  - Data space name
  - Location (either area address or data space offset)
- Control block index, sorted by
  - Address space ID
  - Data space name
  - Location (either area address or data space offset)
- Message index, containing the location of all messages.

The index contains a section of error message data and a section of informational message data. Each section contains a list of error messages sorted by message ID, and the page numbers of the output pages containing the message.

Formatting output for specific components:

You can specify the components for which you want to obtain SDUMP output.

- To obtain all available output for Monitor Services, real-time analysis, or Workload Manager, the format request must include the Topology Services component. Those components have areas anchored within the Topology Services control blocks.

  For example:

  ```
  VERBX CPSM510 'TOP,WLM'
  ```

- To obtain complete output, all data spaces associated with the selected components must be present in the dump.

  For the Monitor Services component, the MAS1xxxx data space must be present, in addition to the TOP1xxxx and MON1xxxx data spaces; if it is not present, the output is incomplete.

Using the ESSS utility (EYU9XEUT)

The Environment Services System Services (ESSS) component of CICSPlex SM is a limited function system address space that remains in the MVS image until the next IPL. ESSS implements a formal MVS subsystem for use by CICSPlex SM.

You can use the batch utility program EYU9XEUT to perform diagnostic and maintenance functions on ESSS and the MVS subsystem.

Important: This utility program should be used only at the request of customer support personnel.

The EYU9XEUT options

The EYU9XEUT batch utility program supports the options DUMP, RELOAD and TERMINATE.

You specify the option you want to use on the SYSIN statement of the program's JCL, as described in “The EYU9XEUT JCL” on page 48.

Dumping data structures (DUMP)

The DUMP option reports on the contents of data structures in both the ESSS and the MVS subsystem at the time the program is run.

The format of the DUMP option is:

```
DUMP  VERSION(nnn|ALL) [SUBSYSTEM] [ESSS] [LOCKS] [NOCML]
```

where:

VERSION
Identifies the version of CICSPlex SM for which a report is to be generated. *nnn* is a specific version of CICSPlex SM, for example 530 for CICSPlex SM for CICS Transaction Server for z/OS®, Version 5 Release 4 . ALL reports on each version of ESSS that has been created at your enterprise.

SUBSYSTEM
Limits the report to the MVS subsystem data structures.

ESSS
Limits the report to the ESSS data structures.

LOCKS
Produces a summary of the ESSS data structures used by CICSPlex SM locks.

NOCML
Prevents EYU9XEUT from trying to obtain the MVS cross-memory local lock (CML), which may be held by a program call routine.

Note: No CICSPlex SM lock summary is produced when NOCML is requested.

By default, the DUMP option generates a report containing MVS subsystem and ESSS data structures.

Reloading broadcast functions (RELOAD)

CICSPlex SM uses two MVS subsystem broadcast functions, end-of-task (EOT) and end-of-memory (EOM). As a result of program maintenance, it might be necessary to reload these functions in an existing ESSS address space.

The RELOAD option loads new broadcast functions from the utility library into the extended common service area (ECSA). You specify the location of the new functions on the UTILLIB statement of the program's JCL, as described in "The EYU9XEUT JCL" on page 48.

The format of the RELOAD option is:

```
RELOAD  VERSION(nnn)  EOT|EOM|ALL
```

where:

nnn
Identifies the version of CICSPlex SM for which broadcast functions are to be replaced. For example, specify 530 for CICSPlex SM for CICS Transaction Server for z/OS, Version 5 Release 4 .

EOT|EOM|ALL
Identifies the broadcast function to be replaced as end-of-task (EOT), end-of-memory (EOM), or both (ALL).

Stopping the ESSS (TERMINATE)

The ESSS address space might occasionally need to be stopped to pick up changes made by CICSPlex SM maintenance (PTFs), or when directed by IBM Support.

The TERMINATE option requests that the ESSS address space is stopped. It can only be used when no other address spaces (for example, CMASes, MASes, CPSM API programs) are connected to the ESSS. To check that no address spaces are connected to the ESSS address space, use the EYU9XENF utility.

Note: Before using the TERMINATE option, all CICSPlex SM CMASes, MASes, and CICSPlex SM API programs that use the same version of CICSPlex SM as the ESSS must be stopped.

The format of the TERMINATE option is :

```
TERMINATE VERSION(nnn)
```

where:

nnn
Identifies the version of CICSPlex SM for which the ESSS is to be stopped. For example, specify 530 for CICSPlex SM for CICS Transaction Server for z/OS, Version 5 Release 4 .

The EYU9XEUT JCL

You use JCL to run the EYU9XEUT utility program.

Figure 7 is an example.

```
//jobname    JOB  (acct),'name',MSGCLASS=x
//UTIL       EXEC PGM=EYU9XEUT
//STEPLIB    DD   DSN=CICSTS54.CPSM.SEYUAUTH,DISP=SHR
//UTILLIB    DD   DSN=CICSTS54.CPSM.SEYUAUTH,DISP=SHR
//SYSPRINT   DD   SYSOUT=*
//UTLPRINT   DD   SYSOUT=*
//SYSIN      DD   *
RELOAD VERSION(530) EOT
/*
```

Figure 7. Sample JCL for EYU9XEUT – RELOAD option

In this example, the RELOAD option is being used to load a new EOT broadcast function into the ECSA. The UTILLIB statement names the data set where the new broadcast function resides.

Note: To use this JCL for the DUMP option, delete the UTILLIB statement and change the RELOAD statement to a valid DUMP statement.

Using the ESSS Information Display Utility (EYU9XENF)

The Environment Services System Services (ESSS) information display utility is a TSO/E command processor that can be used to display information about a CICSPlex SM ESSS.

It is typically used to ensure that no CICSPlex SM address spaces (for example, CMASes, MASes, and CICSPlex SM API programs) are connected to the ESSS. Before EYU9XENF can be used, it needs to be defined as a TSO/E authorized command, and available to the TSO user as follows:

- Add EYU9XENF to the 'AUTHCMD NAMES' section of the IKJTSOxx SYS1.PARMLIB member used by the MVS image.
- Issue from TSO a 'PARMLIB UPDATE(xx)' command to implement the changes made to the IKJTSOxx member.
- Ensure that the SEYUAUTH library that is specified in the TSO user STEPLIB concatenation matches the version of CICSPlex SM that is specified in the EYU9XENF command.

For details of IKJTSOxx and defining an authorized command see the *z/OS MVS Initialization and Tuning Reference*.

The format of the EYU9XENF command is :

```
EYU9XENF nnn
```

where:

nnn
Identifies the version of CICSPlex SM for which the ESSS information is to be displayed. This version must match the version of CICSPlex SM specified in the STEPLIB concatenation.

Figure 8 shows an extract from a report produced by the EYU9XENF TSO command. From this figure we can see that two CMASes are active (in the ESSS), CMASPROD and CMASTEST. In addition three MASes are active PRODMAS1, PRODMAS2, and TESTMAS1.

Note: The CMAS information will only be removed after the CMAS has shutdown and all associated MASes and CICSPlex SM API programs have stopped.

```
Version 530  CICSPlex SM Connection Information ------------------------
  CMAS      Job/User   CICS   Job/Task    MAS       Job/User  CICSPlex  Job/Task
  Name        Name     Sysid    ID        Name      Name      Name      ID
 --------   --------   -----   --------   --------  --------  --------  --------
 CMASC002   CMASC002   C002    Inactive   <<SUR>>   APIPRG1             JOB70001
                                          MASM005   MASM005   PLEX2     JOB60001
                                          MASM004   MASM004   PLEX2     JOB50001
 CMASC001   CMASC001   C001    JOB00001   MASM003   MASM003   PLEX1     JOB40001
                                          MASM002   MASM002   PLEX1     JOB30001
                                          MASM001   MASM001   PLEX1     JOB20001
                                          WUIW001   WUIW001   PLEXW     JOB10001

Version 530  CICSPlex SM ESSS Program Information ------------------------------

 Program Name     Version     Load Point       Date     Time  Level

  EYU9X530         530        1F100810       01/20/15  19.55  JCI700M
  EYUTXEPC         530        1B9C4C80       01/20/15  19.54  JCI700M
  EYU9XEER         530        00CB9BF0       01/30/15  20.13  UI12345
  EYU9XEEM         530        1ECF7208       01/12/15  14.23  UI23456
  EYU9XEET         530        1ECEB9C8       01/12/15  14.23  UI23456
  EYU9XEEE         530        1BBEDA48       01/20/15  19.55  JCI700M

Version 530  CICSPlex SM ESSS Resource Usage Information ----------------------

Resource Name                            Origin     Length     Number     In Use

Connected ASID Table Elements           1F104F60   00050014   00000500   0000000B
Dynamic Work Area Elements              1F154F74   00140014   00000500   00000000
Lock Manager Resource Queues            20494F9C   00B00014   00040000   000004E3
Lock Manager Holder/Waiter Elements     1F294F88   01200014   00080000   00000000
CICSplex Name Blocks                    20F94FBO   00003014   00000400   00000012
Signal Blocks                           20F97FC4   00080014   00001000   00000000
Thread Elements                         21017FD8   00020014   00002000   00000001
Target Interest Elements                21037FEC   00001014   00000100   00000000
Requestor Interest Elements             21039000   00001014   00000100   00000000
```

Figure 8. Extract from ESSS information report from EYU9XENF

Using the online utility transaction (COLU)

The CICSPlex SM online utility (COLU) is a CICS transaction that can be used to generate reports about various CMAS and local MAS components.

Usage Note

This online utility should be used only at the request of customer support personnel.

The COLU transaction

To run the CICSPlex SM online utility, log onto a CICS system that is either a CMAS or a local MAS and enter the COLU command.

The format is as follows:

```
COLU  compid  keyword
```

where:

compid
Is one of the following 3-character component identifiers:

CHE Data Cache Manager

COM Communications

KNL Kernel Linkage

QUE Queue Manager

SRV Common Services

TOP Topology Services

keyword
Is a valid keyword for the specified component.

Valid keywords for component CHE

The valid COLU keywords for the CHE component are CACHE and LIST.

CACHE
Summarizes the data space usage of each component. This keyword can be issued only from a CMAS.

Figure 9 is an example of the report produced by the CACHE keyword. The CACHE report produced by COLU names the data space for each

```
  CICSPLEX SM 540 CICS/ESA SNAP Utility For JOB CMASJOB    2017/04/19
                    CPSM 540  DATA CACHE Dataspace Element Summary
CMAS Name:  CMASNAME     Date/Time:  2017/04/19 10:44:25.814
  Name      Token       ALET      Start       End      Used HWM    Size
DMDSC111   FF000050   01010102   00001000   00800000   0040FAA0   0040EAA0
DAT1C111   FF000060   01010105   00000000   00FFF000   00A02000   00A02000
DAT2C111   FF000160   01010106   00000000   00100000   00001000   00001000
QUE1C111   FF000070   01020104   00001000   00FFF000   001D1000   001D0000
QUE2C111   FF000170   01020103   00001000   00100000   00001000   00000000
COM1C111   AA000080   01010109   00001000   00FFF000   000B5000   000B4000
COM2C111   AA000180   0101010A   00001000   00100000   00001000   00000000
TOP1C111   FF000090   0102010F   00001000   00FFF000   000A9000   000A8000
TOP2C111   FF000190   0102010E   00001000   00100000   00001000   00000000
RTA1C111   AA0000A0   01010112   00001000   00FFF000   00105000   00104000
```

Figure 9. Sample CACHE report from COLU

component of the CMAS and shows its ALET, its location, and the amount of storage used. Items in the report are explained as follows:

ALET The ALET of the data cache.

End The end address of allocated data cache storage.

Name The name of the data cache. CPSM components can have multiple data cache.

Size The amount of data cache storage between the start and end addresses.

Start The start address of allocated data cache storage.

Token The token of the data cache.

Used HWM
The peak amount of allocated data cache storage that has been used.

LIST
Summarizes the data cache list usage of each CMAS component. This keyword can be issued only from a CMAS.

Figure 10 is an example of the report produced by the LIST keyword.

```
CICSPLEX SM 540 CICS/ESA SNAP Utility For JOB CMASJOB    2017/04/19               PAGE  00001
                          CPSM 540  DATA CACHE List Element Summary
          CMAS Name:  CMASNAME    Date/Time:  2017/04/19 10:50:38.457
CompId   Type   Technique  Indexes  Elem Len   Key Off  Key Len   Token      Alet     Start      End     Elem Cnt
 DAT   Standard  BinSrch     No     0000003D    0000       07    FF000060  01010105  0005B800  00063E2B  00000002
 DAT   Standard  BinSrch     No     0000001E    0000       06    FF000060  01010105  00057600  0005B7FE  0000000B
 DAT   Standard  BinSrch     No     00000018    0000       10    FF000060  01010105  00008B00  00008CB4  00000000
 DAT   Standard  BinSrch     No     00000014    0000       02    FF000060  01010105  00008900  00008A6C  00000000
 DAT   Standard  BinSrch     No     0000003D    0000       07    FF000060  01010105  0001C200  0002482B  00000000
 DAT   Standard  BinSrch     No     0000001E    0000       06    FF000060  01010105  0000C200  000103FE  00000000
 DAT   Standard  BinSrch     No     00000008    0000       08    FF000060  01010105  0000A400  0000B59C  00000002
 DAT   Standard  BinSrch     No     00000008    0000       08    FF000060  01010105  00008600  000086FC  00000000
 DAT   Standard  BinSrch     No     00000028    0000       08    FF000060  01010105  0001B900  0001C19C  00000000
```

Figure 10. Sample LIST report from COLU

Items in the report are explained as follows:

Alet The Alet of the data cache where the cache list is located.

CompId
The CPSM component name.

Elem Cnt
The number of elements in the cache list.

Elem Len
Length of the cache list elements.

End The end address of the cache list storage in the data cache.

Indexes
Specifies whether the cache list supports alternative indexes.

Key Len
Length of the element key that is used for searching.

Key Off
Offset into the element to the element key that is used for searching.

Start The start address of the cache list storage in the data cache.

Technique
The search method that is used to find entries. Only BinSrch (binary search) is supported.

Token The token of the data cache where the cache list is located.

Type The type of cache list. A cache list can be standard or generic.

Valid keywords for component COM

The valid COLU keywords for the COM component are MALRL, MASRL and NETOP.

MALRL
Lists all outstanding message argument lists (MALs) for the CMAS. This keyword can be issued only from a CMAS.

MASRL
Lists all outstanding message argument lists (MALs) for all MASs attached to the CMAS. This keyword can be issued only from a CMAS.

NETOP
Lists the communication network topology as it is known to the CMAS. This keyword can be issued only from a CMAS.

Valid keywords for component KNL

The valid COLU keyword for the KNL component is ESSSINFO.

ESSSINFO
Summarizes the resources in use by the Environment Services System Services (ESSS) address space. This keyword can be issued only from a CMAS.

Figure 11 is an example of the report produced by the ESSSINFO keyword. The ESSSINFO report produced by COLU provides information about active

```
      CICSPLEX SM 530 CICS/ESA SNAP Utility For JOB CVMD6C1    2015/02/04 185223..  PAGE  00001
               CPSM 530 Kernel Linkage CICSPlex SM ESSS Connection Information
  CMAS Name   Job Name   CICS SYSID   Job ID    MAS Name   Job Name  CICSPlex Name  Job ID
   CMASC002    CMASC002   C002       Inactive               APIPRG1                JOB70001
                                                MASM005     MASM005   PLEX2        JOB60001
                                                MASM004     MASM004   PLEX2        JOB50001
   CMASC001    CMASC001   C001       JOB00001   MASM003     MASM003   PLEX1        JOB40001
                                                MASM002     MASM002   PLEX1        JOB30001
                                                MASM001     MASM001   PLEX1        JOB20001
                                                WUIW001     WUIW001   PLEXW        JOB10001
      CICSPLEX SM 530 CICS/ESA SNAP Utility For JOB CVMD6C1     2015/02/04 185223..  PAGE  00002
               CPSM 530 Kernel Linkage CICSPlex SM ESSS Program Information
 Program Name     Version     Load Point       Date     Time  Level
  EYU9X530         530        1F100810       01/20/15  19.55  JCI700M
  EYUTXEPC         530        1B9C4C80       01/20/15  19.54  JCI700M
  EYU9XEER         530        00CB9BF0       02/30/15  20.29  UI12345
  EYU9XEEM         530        1ECF7208       01/12/15  14.23  UI23456
  EYU9XEET         530        1ECEB9C8       01/12/15  14.23  UI23456
  EYU9XEEE         530        1BBEDA48       01/20/15  19.55  JCI700M
      CICSPLEX SM 530 CICS/ESA SNAP Utility For JOB CVMD6C1     2015/02/04 185223..  PAGE  00003
               CPSM 530 Kernel Linkage CICSPlex SM ESSS Resource Usage Information
Resource Name                          Origin     Length     Number     In Use
Connected ASID Table Elements          1F104F60   00050014   00000500   0000000B
Dynamic Work Area Elements             1F154F74   00140014   00000500   00000000
Lock Manager Resource Queues           20494F9C   00B00014   00040000   000004E3
Lock Manager Holder/Waiter Elements    1F294F88   01200014   00080000   00000000
CICSplex Name Blocks                   20F94FBO   00003014   00000400   00000012
Signal Blocks                          20F97FC4   00080014   00001000   00000000
Thread Elements                        2AE6FFD8   00020014   00002000   00000000
Target Interest Elements               21037FEC   00001014   00000100   00000001
Requestor Interest Elements            21039000   00001014   00000100   00000000
```

Figure 11. Sample ESSSINFO report from COLU

CMASs and the MASs that are connected to them, the ESSS system programs, and the ESSS resource tables.

Valid keywords for component QUE

The valid COLU keywords for the QUE component are ALL, COMPID, METH and SUM.

ALL
Indicates that all allocated queues should be listed. When ALL is specified, no other keyword is permitted. This keyword can be issued from any CMAS or MAS.

COMPID(*xxx*)
Is a 3-character CICSPlex SM component ID. This keyword can be issued from any CMAS or MAS.

METH(*xxxx*)
Is a 4-character CICSPlex SM method name. This keyword can be issued from any CMAS or MAS.

SUM
Causes a summarization report to be generated. In the detailed report, each line describes an allocated queue. This keyword can be issued from any CMAS or MAS.

Figure 12 is an example of the report produced by the ALL keyword.
The QUE ALL report produced by COLU provides information about queue

```
CICSPLEX SM 540 CICS SNAP Utility For JOB CVMCJBC     2017/02/01
                       CICSPlex Allocated Queue Resources
  Queue Token     AllcStg  TotRec   Meth MaxRecln Mode Type DbugText
801B0001 00001060 00008000 00000000 XLNX 00000000 Del  Wait NTFYQUE
801B0001 00001080 00008000 00000000 XDIN 00000000 Rept Work
801B0002 000010A0 00008000 00000004 CIIN 00000078 Rept Work EYU0CIIN
801B0002 000010C0 00008000 00000001 CIIN 00000052 Rept Work EYU0CIIN
801B0003 000010E0 00008000 00000001 CSSR 00000052 Rept Work COMMDEFS
801B0002 00001100 00008000 00000000 CWIN 00000000 Del  Wait CWINXQCQ
801B0001 00001120 00008000 00000000 CPLT 00000000 Del  Wait EYU0CPLT
```

Figure 12. Sample QUE ALL report from COLU

resources allocated by the CMAS or MAS, including their location, allocated storage, total number of records, method, maximum record length, mode, type, and text used in debugging.

Valid keywords for component SRV

The valid COLU keywords for the SRV component are LOCKS and LOCKSUM.

LOCKS
Dumps the contents of all lock manager control blocks that are local to the CMAS or MAS. This keyword can be issued from any CMAS or MAS.

LOCKSUM
Summarizes the lock manager usage of all locks that are local to the CMAS or MAS. This keyword can be issued from any CMAS or MAS.

Figure 13 on page 54 is an example of the report produced by the LOCKSUM keyword.

```
CICSPLEX SM 530 CICS/ESA SNAP Utility For JOB CMAS001     2015/01/13
        CPSM 530  Common Services Lock Management Summary
Total number of Resource Queue pools                          1
First Resource Queue pool address                      21388F70
Total Resource Queue pool size                         0001A030
Total number of Resource Queues                           2,048
Number of Resource Queues in use                            139
Total number of Resource Holder/Waiter Element pools          1
First Resource Holder/Waiter Element pool address      213A2FA0
Total Resource Holder/Waiter Element pool size         0002C030
Total number of Resource Holder/Waiter Elements           4,096
Number of Resource Holder/Waiter Elements in use              2
CICSPLEX SM 530 CICS/ESA SNAP Utility For JOB CMAS001     2015/01/13
        CPSM 530  Common Services Lock Management Summary
Lock 21BFAEB0  Owner CMAS001   Resname LCL_LOCK
Holder CMAS001 Task 9035 Dsa 21C8F8C8 Mtd DBG1 Lvl EXCL Use  1   GMT date/time  1/13/15  17;20:25.56238
Waiter CMAS001 Task 9053 Dsa 21CA58C8 Mtd DBG2 Lvl SHR           GMT date/time  1/13/15  17;20:43.11484
Lock 00076F80  Owner CMAS001   Resname PSTIXSRS
Lock 00076C80  Owner CMAS001   Resname PSTIXSRS
```

Figure 13. Sample LOCKSUM report from COLU

The LOCKSUM report produced by COLU provides information about local locks in use by the CMAS or MAS, including their location, size, and number.

Valid keywords for component TOP

The valid COLU keyword for the TOP component is PLEX.

PLEX(*plexname* **[,***scope***])**
Lists the topology of the specified CICSplex as it is known to the CMAS. The optional *scope* value limits the report to a named CICS system or CICS system group within the CICSplex. This keyword can be issued only from a CMAS.

Using the interactive debugging transactions (COD0 and CODB)

The interactive debugging transactions COD0 and CODB provide access to the CICSPlex SM runtime environment. They can be used to format and manipulate the internal data structures of CICSPlex SM.

ATTENTION

THE CICSPlex SM INTERACTIVE DEBUGGING TRANSACTIONS COD0 AND CODB SHOULD BE USED ONLY AT THE REQUEST OF IBM CUSTOMER SUPPORT PERSONNEL. YOU MUST TAKE STEPS TO ENSURE THAT THESE TRANSACTIONS MAY BE USED ONLY BY AUTHORIZED PERSONNEL BECAUSE OF THE EXTENT OF THE ACCESS TO SYSTEM CONTROL AREAS THAT THEY PROVIDE. IMPROPER OR UNAUTHORIZED USE OF COD0 AND CODB MAY HAVE VERY SERIOUS CONSEQUENCES, INCLUDING WITHOUT LIMITATION LOSS OF DATA OR SYSTEM OUTAGE. CUSTOMER SHALL BE SOLELY RESPONSIBLE FOR SUCH MISUSE.

The debugging transactions can run in CMASs and in managed CICS regions that have terminal support.

Running the debugging transactions

To run the CICSPlex SM debugging transactions, log on to a CICS system and enter **CODU** or **CODB**.

COD0 To use the method-level debugging transaction, as described in "Method-level debugging with COD0" on page 55. This transaction

provides access to CICSPlex SM objects, methods, message argument lists (MALs), and outstanding requests. To exit this transaction, type EXIT on the command line.

CODB

To use the system-level debugging transaction, as described in "System-level debugging with CODB" on page 83. This transaction provides access to address space and data space storage, major control blocks, data queues, and CICSPlex SM entries in the CICS trace table. To exit this transaction, press PF3 or type END on the command line.

The following usage rules apply to the COD0 and CODB transactions:

- You issue a COD0 command by typing the command name on the command line. You issue a CODB command by typing its option number on the command line.
- The standard END and CANCEL commands are recognized. END completes the task in progress and returns you to the previous screen, while CANCEL cancels the task before returning.
- You can scroll a display by using the commands DOWN, UP, TOP, and BOT. With COD0, you can also enter a default scrolling amount in the Scroll==> field.
- On a selection list, any character that is not a blank or an underscore can be used to select an option.
- These transactions support only 3270 model 2 screens that is, 24x80 and 32x80 type screens.

Method-level debugging with COD0

After logging onto CICS, enter the COD0 transaction ID to display the COD0 main menu.

Figure 14 shows the COD0 main menu.
To issue a COD0 debugging command, enter it in the CMD=> input field.

```
COD1 CICSPlex SM Debugger
CMD=>                                                    Scroll=> PAGE
Welcome to CICSPlex SM Debugger. Commands available are:

     ALLOC       Allocate storage, cache list, queue, or eptr.
     ATTACH      Attach a method to run in CMAS/MAS.
     CALL        Call a CICS transaction or program.
     DUMP        Call DBUG transaction to display memory.
     EXEC        Executes a method directly from the debugger.
     EXIT        EXIT the debugger
     LIST        List methods, CPSM tasks and resources.
     PRINT       Print a CPSM data area to the JES Spooler
     PURGE       Delete a resource ALLOCated.
     POST        POST ECB or ECB list.
     SET         Change CPSM EYUPARM value.
     START       Starts a method running in CMAS.
     TRACE       Set CPSM component trace flags.
     TRACK       Set CPSM trace flags based on calling structure.
     TRAP        Set tracing flags for a single method.

 Enter HELP (command) for more help on commands.
```

Figure 14. COD0 debugging transaction menu

Commands can include one or more parameters, which must be separated by one or more spaces. Commas and quoted strings are not supported.

As in ISPF, function keys are prefixed to whatever is on the command line. The following function keys are in effect when COD0 is running:

Key	Description
F1	HELP
F3	END
F4	PREV
F5	NEXT
F7	UP
F8	DOWN

Issuing commands recursively

You can enter the debugger commands recursively from any screen in the COD0 transaction, effectively nesting the commands and their output.

When the LIST and HELP commands are entered recursively, the new output replaces the old. For example, if you issue the LIST START command followed by the LIST TASK command, the LIST TASK output replaces the LIST START output.

Issuing commands that alter CICSPlex SM

Certain COD0 debugging commands can be used to modify memory or some other aspect of CICSPlex SM operation.

The commands are:

- ATTACH
- EXEC
- POST
- START

When you issue one of these commands, you receive a warning and confirmation panel. You should proceed with the command only at the request of customer support personnel.

ALLOC (allocating a resource)

The ALLOC command allocates a resource so that you can refer to it by name in completing MALs.

The resource can be a cache list, a data queue, data space storage, or shared CICS storage.

The format of the ALLOC command is:

```
ALLOC /resname [optional parameters...]
```

where:

/resname
: Identifies the resource being allocated. The resource name can be no more than eight characters, including the required slash.

The optional parameters are:

QUEUE *compid*
: Creates a queue token and assigns it to the resource being allocated. *compid* is the 3-character component identifier, as listed in Chapter 10, “Major components of CICSPlex SM,” on page 113.

CLIST
Displays the Allocate CACHE LIST input panel (shown in Figure 15), which lets you create a CACHE LIST token and assign it to the resource being allocated.

STG *size* **[BELOW]**
Acquires an address of the specified size from CICS shared storage and assigns it to the resource being allocated. *size* is a number of bytes. The BELOW option requests storage from below the 16MB line; by default, storage is acquired above the line (in 31-bit mode).

EPTR *size*
Acquires a data space pointer of the specified size from a data space and assigns its ALET and OFFSET to the resource being allocated. *size* is a number of bytes.

```
COD1 CICSPlex SM Debugger
CMD=>                                                    Scroll=> PAGE

Allocate CACHE LIST

     Id of CACHE to create CACHE LIST: /@CACHE           (Optional)

     Estimated number of elements:

     Element size:

     Estimated free space:        (Optional)

     GENERIC if generic keys:          (Optional)

     Hash Table Size:        (Optional)

     Key Offset: 0      (Default 0)

     Key Size:

     Search method (BINSRCH/HASH): BINSRCH
```

Figure 15. Allocate CACHE LIST panel

Resources remain allocated across multiple COD0 transactions or between multiple COD0 transactions running concurrently in the same CICS system. In fact, all resources exist until you specifically purge them.

Note:

1. You can use the LIST ALLOC command to display a list of allocated resources.
2. You can use the DUMP /resname command to dump the storage, data queue, or cache list for an allocated resource.

ATTACH (attaching a method)

The ATTACH command starts a method running in the CICS systems identified by the specified context and scope values.

The format of the ATTACH command is:

```
ATTACH method context scope
```

where:

method
Is the ID of a CICSPlex SM method.

context
Is the name of a CMAS or CICSplex.

scope
Is the name of a CICSplex, CICS system group, or CICS system.

For a list of valid responses to this command, see "Running a method" on page 82.

Unlike the START command, which merely starts a CICS transaction within a CMAS, ATTACH crosses the boundary between a CMAS and a local MAS. (These methods may run in the CMAS, a different address space, or even a different processor in the CICSplex).

Figure 16 is an example of the display for a completed attached task that ran within a single CMAS or MAS.
Figure 17 is an example of the display for a completed attached task that either ran

```
COD1 CICSPlex SM Debugger
CMD=>                                                          Scroll=> PAGE
Enter END to exit or ENTER to view results.
Status for ATTACHed method  XQCQ

       Methods status:  Method completed.
       XLCI return description:  OK
       Method's RESPONSE was:    OK
       Method's REASON was:

       CONTEXT:  CVMCTSO1   SCOPE:  CSYSGRP1   REGION:  CSYSGRP1

       Unit of work
          SYSID:  TEST     USERID:  DEVOPER  TCB  00452160
          Major Object:  00   Component Id:  73


       The method executed in a single MAS so all information
       appears in the fields of the MAL.
```

Figure 16. Attached task display for a single CMAS or MAS

in multiple MASs, or ran multiple times in a CMAS.
When you press Enter, each of the MALs that ran in each region is reconstructed

```
COD1 CICSPlex SM Debugger
CMD=>                                                          Scroll=> PAGE
Enter END to exit or ENTER to view results.
Status for ATTACHed method  XQCQ

       Methods status:  Method completed.
       XLCI return description:  OK
       Method's RESPONSE was:    OK
       Method's REASON was:

       CONTEXT:  CVMCTSO1   SCOPE:  CSYSGRP1   REGION:  CSYSGRP1

       Unit of work
          SYSID:  TEST     USERID:  DEVOPER  TCB  00452160
          Major Object:  00   Component Id:  74

       The method executed in multiple MAS so a queue of OUT records
       was created.

       OUTQUE QUEUE ID:  A4957FBD B3E11932
                    Records :  0000013  Record Length:  0000018
```

Figure 17. Attached task display for multiple CMASs or MASs

and displayed individually, as shown in Figure 18 on page 59.

```
COD1 CICSPlex SM Debugger
CMD=>                                                        Scroll=> PAGE
NEXT/PREV to browse CICS region MALS. END=Exit.
MAL for CICS Region:CICSSY01
IN
   *ENM FUNCTION( CREQUE )
    CHR DEBUG(          )
    PTR ECB( 00000000 )
   *CMP MAJOR_OBJECT( KNL )
   *ENM TYPE( WORK )
   *SDT DELETE( TRUE  )
OUT
   *ENM RESPONSE( OK )
   *ENM REASON(                    )
   *ETK QTOKEN( A4957FC53998FB31 )
```

Figure 18. MAL display for a specific CICS region

Note also that the region the MAL ran in is shown on the header line for the display. You can use the NEXT (PF5) and PREV (PF4) keys to browse backwards and forwards between the regions. END (PF3) returns you to the attached task display.

CALL (calling external CICS programs and transactions)

The CALL command calls a CICS transaction or program with optional parameters.

The format of the CALL command is one of the following:

```
CALL cicstran [optional parameters...]

CALL PROGRAM cicsprog [optional parameters...]
```

Note: CICS can be used as a synonym for CALL.

cicsprog
Is a program ID that must be defined to CICS.

cicstran
Is a transaction ID that must be defined to CICS.

The parameters are passed as a TIOA area, so anything that can be entered at the transaction's or program's initial screen can be specified as an optional parameter. There is no validation of the optional parameters.

For transaction calls, the transaction ID is placed as the first field in the constructed TIOA (as it would be from the terminal). Make sure the transaction is defined as conversational. Pseudo- or nonconversational programs return immediately to COD0.

For program calls, you must enter the transaction ID as the first parameter, if the program you are calling expects this.

While the task is running, all the facilities of that CICS transaction or program are available to you. When you end the task, you return to COD0.

Note: You should not attempt to call:

- The COLU transaction, which is used by CICSPlex SM
- CICSPlex SM programs, which begin with the letters 'EYU'

CAPTURE (capturing and printing a table)

The CAPTURE command captures the communication between an API program, including a WUI session, and its connected CMAS when API resource table records are requested, or the communication between CICSPlex SM monitor programs and a MAS when Monitor data is collected in the MAS. In either case, data is written to a JES spool file called Sxxxxxxx, where xxxxxxx is a numeric identifier, under the CMAS (*tblname* option) or MAS (**MASMON* option).

The format of the CAPTURE command is one of the following:

```
CAPTURE tblname userid count   (available in a CMAS only)

CAPTURE *MASMON montype count  (available in a MAS only)
```

where:

tblname
: Is an API resource table name. The Resource Tables Reference manual lists all API resource tables.

userid
: Is the API/WUI user ID. Only requests from this user are captured.

count
: Is a number from 0 though 999. the number indicates how many times a capture is performed.

  You can reissue the CAPTURE command with the same table name and user ID to update the count. A count of zero deletes the CAPTURE entry.

***MASMON**
: Captures monitor data as it is collected by a MAS.

montype
: Is the type of monitor data to be captured:

  MCICS
  : CICS regions

  MCONN
  : Connections

  MDBX
  : DB2® and DBCTL resources

  MFILE
  : Files

  MGLBL
  : Global resources

  MJRNL
  : Journals

  MPROG
  : Programs

  MTDQS
  : Transient data queues

  MTERM
  : Terminals

  MTRAN
  : Transactions

Note: For *tblname*, the count is the number of GET requests captured. For **MASMON*, the count is the number of monitor intervals captured.

For example:

```
CAPTURE MONDEF USER39 3
```

captures the next three MONDEF table API/WUI GET requests issued by USER39. All related MALs and queues are printed.

CAPTURE (capturing and printing a view)

The CAPTURE command captures and prints all communications related to a CICSPlex SM end-user interface view being issued by a particular user. CAPTURE uses the CICS spool facility to write the data as an output file called Sxxxxxxx, where xxxxxxx is a numeric identifier.

The format of the CAPTURE command is one of the following:

```
CAPTURE viewname userid count

CAPTURE *MASMON montype count
```

where:

viewname
Is the name of the CICSPlex SM view to be captured.

userid
Is the TSO user ID of the user who will be issuing the view command.

count
Is the number of times the view should be captured.

A count is taken from the time the view command is entered until the user enters another view command or END. Pressing Enter repeatedly to refresh the data or perform some action against the view does not change the count of the view command.

You can reissue the CAPTURE command with the same view name and user ID to update the count. A count of zero deletes the CAPTURE entry.

***MASMON**
Captures monitor data as it is collected by a MAS.

montype
Is the type of monitor data to be captured:

MCICS
CICS regions

MCONN
Connections

MDBX
DB2 and DBCTL resources

MFILE
Files

MGLBL
Global resources

MJRNL
Journals

MPROG
Programs

MTDQS
Transient data queues

MTERM
Terminals

MTRAN
Transactions

For example:

```
CAPTURE MONDEF USER39 3
```

captures the next three MONDEF view commands issued by USER39. All related MALs and queues are printed.

DUMP (displaying and altering data)

The DUMP command displays a scrollable dump of memory. Some parameters of the DUMP command cause the CICSPlex SM system-level debugging transaction, CODB, to be invoked.

If you alter the displayed memory, you must enter UPDATE (or press PF11) to record the change. If you alter memory but do not enter UPDATE, a message is displayed to remind you to enter UPDATE.

The format of the DUMP command is:

```
DUMP [parameters...]
```

where the parameters are:

/resname
Displays the queue, EPTR, storage, or cache list allocated to the specified resource.

@*method*
Calls CODB with the entry point of the specified CICSPlex SM method.

hexadecimaladdress
Assumes the hexadecimal value is an address and enters CODB with ALET=0 and the address specified.

hexadecimalALET hexadecimaloffset
Displays the address of the specified ALET (first hexadecimal value) at the specified offset (second hexadecimal value).

hexadecimaladdress [length]
Displays the storage starting at the specified address. The amount of storage displayed is determined by the length parameter. Length is assumed to be a decimal value, unless a X'length' value is specified.

CACHE *cachetoken*
Displays the data identified by the specified cache-list token. The token is entered as two 8-byte hexadecimal character strings. To display the previous record, use PF4; to display the next record, use PF5; to display a specific record, enter REC *n*, where *n* is the record number.

CLIST *token*
Calls CODB with the specified token. The token is entered as two 8-byte hexadecimal character strings.

EIB *compid*
Displays the address of the CICS information block for the first transaction running under the specified component.

EIS *compid*
Displays the address of the CICS storage block for the first transaction running under the specified component.

MODB *compid*
Displays the address of the MODB for the specified component.

MODD
Displays the address of the MODD.

MOEB *compid*
Displays the address of the MOEB for the specified component.

OPB *compid*
Displays the address of the first OPB for the specified component.

QUE *token*
Displays the data identified by the specified token. The token is entered as two 8-byte hexadecimal character strings.

STAKEND *compid*
Displays the address of the last stack for the first transaction running under the specified component.

STAKSTRT *compid*
Displays the address of the initial stack for the first transaction running under the specified component.

XLWA
Displays the CICSPlex SM kernel linkage work area.

Figure 19 shows an example XLWA display.

```
COMMAND==> XLWA       COMP ID==>      ADDR==>           ALET==> 00000000
MSG==>
0012D0A8  00000000  08006EC5  E8E4D5E7  D2D5D3C3  E6C1C1C2  ..>EYUNXKNLCWAAB
0012D0B8  00000010  0D000200  00000000  00000000  00000000  ................
0012D0C8  00000020  8001608B  0000000E  0000EE00  11F20000  ..-.........2..
0012D0D8  00000030  0012D168  0000DE5C  113E8000  00000546  ..J....*........
0012D0E8  00000040  116D4468  00040000  11F2EE00  00100000  ._.......2......
0012D0F8  00000050  11F2EE00  FFFFFF6A  00000000  00000000  .2.....|........
0012D108  00000060  00000000  D2D3D7C2  116D8EB0  116DB328  ....KLPB._..._..
0012D118  00000070  116D7728  00CA812C  00000000  00000000  ._....a.........
0012D128  00000080  00000000  00000000  00000000  00346EC5  .............>E
0012D138  00000090  E8E4E7C5  C5E8E4D9  E7C5D3E2  01030000  YUXEEYURXELS....
0012D148  000000A0  00CC4008  0E1B9CE8  00CA8118  00CA812C  .. ....Y..a...a.
0012D158  000000B0  00F69A00  009C9520  0000011C  00000001  .6....n.........
0012D168  000000C0  00000000  11F20000  11F21100  11F22200  .....2...2...2..
0012D178  000000D0  11F23300  11F24400  11F25500  11F26600  .2...2...2...2..
0012D188  000000E0  11F27700  11F28800  11F2AA00  11F29900  .2...2h..2...2r.
0012D198  000000F0  11F2BB00  11F2CC00  00000000  00000000  .2...2..........
0012D1A8  00000100  11F2DD00  00000000  00000000  00000000  .2..............
0012D1B8  00000110  00000000  00000000  00000000  00000000  ................
0012D1C8  00000120  00000000  00000000  00000000  00000000  ................
```

Figure 19. An example DUMP XLWA display

Notes:

1. If you issue the DUMP command without parameters, the CODB main menu is displayed.

2. For CICSPlex SM components, CODB displays the first transaction running under that component, which is its first OPB. You can use the NEXT and PREV PF keys to display multiple transactions. You can also use the LIST TASK command to display all of the stacks and methods in all of the CICSPlex SM tasks, and then select specific stacks, methods, or OPBs to display.

EXEC (executing a method)

The EXEC command executes a method directly from the COD0 debugging transaction.

The format of the EXEC command is:

```
EXEC method
```

where:

method
Is the name of a CICSPlex SM method.

The formatted message argument list (MAL) for the method is displayed. For details about how to enter data from this display, see "Displaying a MAL from COD0" on page 79. For a list of valid responses to this command, see "Running a method" on page 82.

EXIT (exiting COD0)

The EXIT command exits the COD0 debugging transaction.

This command has no parameters.

You can use this command to exit the debugging transaction from any screen. A closing message is displayed; you can then clear the CICS screen and enter another transaction.

Note:

All allocated resources and started or attached tasks are recorded in a temporary storage record. The next time you enter COD0, all allocated resources are still available and all started or attached tasks can be displayed using the LIST START command.

HELP (getting online help)

The HELP command displays help text for COD0 commands.

The format of the HELP command is:

```
HELP [cmdname | COMPID]
```

where:

cmdname
Is the COD0 command for which help information is to be displayed.

COMPID
Produces a list of identifiers of CICSPlex SM components.

If you issue the HELP command without parameters, the initial help panel, which lists all COD0 commands, is displayed.

LIST (listing tasks and allocated resources)

The LIST command lists running CICSPlex SM tasks, the status of started and attached tasks, and the allocated resources available to you.

The format of the LIST command is:

```
LIST [parameters...]
```

where the parameters are:

ALLOC
Lists all allocated resources. You can purge or dump resources from this screen.

ATCB
Lists the API task control blocks used for processing CICSPlex SM API requests.

CACHE
Lists the data caches in use by a local MAS.

CLIST
Lists the data cache lists in use by a CMAS.

COMM
Lists the two communication MAL queues: one for methods executing via the CMAS, and the other for methods routed to a MAS.

METH [*compid***]**
Lists all methods within the specified component. If no component is specified, all methods are listed.

START
Lists all started and attached tasks and their current status. You can purge, display, or dump the MAL created from this screen as well as restart, attach, or execute the same MAL.

STCB [ERRORS]
Lists the server-client control blocks. The ERRORS option provides a description of any errors encountered.

TASK [*compid***]**
Lists the CICSPlex SM tasks from the specified component showing all active method calls. If no component is specified, all tasks are listed.

Note: You must issue END or CANCEL to terminate a LIST task.

LIST ALLOC:

LIST ALLOC lists all the resources that have been allocated by the ALLOC command.

Figure 20 on page 66 shows an example of the LIST ALLOC display.

```
COD1 CICSPlex SM Debugger
CMD=>                                                          Scroll=> PAGE
Select P=Purge resource  D=Dump resource
S Type  Name     Token             Length MajObj
_ CACHE /@CACHE  01FF0004 000026E0
_ QUE   /QUE     A44C5E58 27257332       MAS
_ CLIST /C       000026E0 00106DF8   256
_ STG   /STG     00000000 04289000  4096
_ EPTR  /E       01FF0005 00001B00  2048
```

Figure 20. An example LIST ALLOC display

The fields on this display are:

Field **Description**

Type The type of resource, as one of the following:

EPTR Data space pointer

CLIST Cache list

CACHE
Cache

STG CICS storage

QUE Queue ID

Token The 4-byte address or 8-byte token broken into two fullwords (ALET first).

Length
The size of allocated storage or the element length for a cache list.

MajObj
The major object, or component, used when allocating.

You can enter the following in the selection field:

Command
Description

P Purges the resource.

Note: You cannot purge /@CACHE, which is the cache created by COD0.

D Enters CODB and dumps the resource. This is the same as entering DUMP */resname* on the command line.

LIST ATCB:

LIST ATCB lists the API task control blocks. These control blocks are used when a CICSPlex SM API operation is in progress.

Figure 21 on page 67 shows an example of the LIST ALLOC display.

```
COD1 CICSPlex/SM Debugger                       APPLID=IYEGZGC0
      CMD=>                                          Scroll=> PAGE
      D=Dump ATCB  C=Dump CMDDesc  T=Task  END=Exit.
      S ATCB     Status   Task# Cmd  Origin        DispTime APITime
      _ 164B82B0 Active *  6337 GET_ IYEGZGW0/7186 00:00.00 00:32.93
      _ 164B78D0 Avail  *  6344
      _ 164B5ED0 Free
      _ 164B58D0 Free
      _ 1649EED0 Free
      _ 1649ECB0 Free
               Total API Commands:          114,842
```

Figure 21. An example LIST ATCB display

The fields on this display are:

Field **Description**

ATCB The address of the ATCB control block

Status The status of the control block, as one of the following:

Active
An API request is being processed.

In Use
An API request has been queued for processing.

Avail Available and waiting for work.

Free Available, but not waiting for work.

Task# The CICS task number

Cmd The CICSPlex SM API command being processed. This field is displayed only when Status=Active.

Origin
For CICS based requests, Origin is in the form: `aaaaaaaa/nnnnn` where `aaaaaaaa` is the CICS APPLID and `nnnnn` is the CICS task number making the CICSPlex SM API request being processed.

For non-CICS based requests, Origin is the MVS job name of the address space making the CICSPlex SM API request being processed.

This field is displayed only when Status=Active.

DispTime
The elapsed time it took to dispatch the current API request.

This field is displayed only when Status=Active.

APITime
The elapsed time is has taken to process the current API request (Dispatch time not included).

This field is displayed only when Status=Active.

The Total API Commands line displays the count of CICSPlex SM API commands that have been processed by all ATCBs.

You can enter the following in the selection field:

Command
Description

D Dumps the ATCB

C Dumps the API Command Descriptor if it is available.

T Displays LIST TASK output for the CICS task processing the ATCB.

LIST CACHE:

LIST CACHE lists the data caches in use by a local MAS.

Figure 22 shows an example of the LIST CACHE display. The fields on this display are:

```
COD1 CICSPlex SM Debugger
CMD=>                                                          Scroll=> PAGE
D=Dump lowest ALET:X'1000'
S Cache    ALET     Low Ofs  High Ofs   HWM Allocated (Hex)
_ DMDSCWW1 01FF001B 00000000 00801000       4,229,376 (00408900)
_ WLM1CWW1 01FF0007 00000000 00400000       3,543,040 (00361000)
_ RTA1CWW1 01010042 00000000 00400000         524,288 (00080000)
_ MON1CWW1 01010041 00000000 00400000         524,288 (00080000)
_ TOP1CWW1 01FF0006 00000000 00400000         655,360 (000A0000)
_ COM1CWW1 01010040 00000000 00400000       1,691,648 (0019D000)
_ MAS1CWW1 01FF0008 00000000 00400000         933,888 (000E4000)
_ DAT1CWW1 0101003F 00000000 00400000       1,527,808 (00175000)
_ QUE1CWW1 01FF0005 00000000 00400000       1,048,576 (00100000)
```

Figure 22. An example LIST CACHE display

Field Description

Cache The name of the data cache.

ALET The ALET of the data cache.

Low Ofs
The lowest offset allocated, which should always be X'00000000'.

High Ofs
The highest offset within the data cache allocated.

HWM Allocated
The number of bytes of the data cache in use.

(Hex) The HWM Allocated value expressed in hexadecimal.

Figure 23 shows an example of the LIST CACHE display. The fields on this display are:

```
DBG0 CICSPlex SM Debugger                              Applid:CICSWIN
  CMD=>                                                        Scroll=> PAGE
  _ Token             ElemLen Keylen Keyoff Records MaxRecs FreeRec    Storage
  _ 00000001-0123FA2C     300      5      0       3      10       1        760
```

Figure 23. An example LIST CACHE display (CMAS)

Field Description

Token The cache list token.

ElemLen
The length of the element.

Keylen
The length of the key.

Keyoff
The offset of the key in each record.

Records
The number of records in the cache.

MaxRecs
The maximum number of records the cache can hold before being expanded.

FreeRec
The number of free slots available.

Storage
The total storage size, including any overhead.

You can enter the following in the selection field:

Command
Description

D Dumps the cache list data. This is the same as entering DUMP CACHE *cachetoken* on the command line.

LIST CLIST:

LIST CLIST lists the data cache lists in use by the CMAS.

Figure 24 shows an example of the LIST CLIST display.
The fields on this display are:

```
COD1 CICSPlex SM Debugger
CMD=>                                                    Scroll=> PAGE
D=Dump the Cache List
S Token            DataAlet DataStrt DataEnd  EleSz ElemCnt Key Len T S Alt
_ 01FF001B00408990 01FF0007 00071000 00073C00    20       0   0  16 S B YES
_ 01FF001B00408830 01FF0007 0016E000 00177E5C    72       0   0  16 S B YES
_ 01FF001B004086D0 01FF0007 00165000 0016DCC4    64       0   0  16 S B YES
```

Figure 24. An example LIST CLIST display

Field **Description**

Token The cache list token.

DataAlet
The ALET of the cache list's data.

DataStrt
The starting offset within the ALET allocated to the cache list.

DataEnd
The highest offset within the ALET allocated to the cache list.

EleSz The size of each cache list element.

ElemCnt
The number of elements in the cache list.

Key The offset of the key within an element.

Len The length of the key.

T The type of cache, as either standard (S) or generic (G).

S The search type for the cache, as either binary (B) or hash (H).

Alt Indicates whether there is an alternate index cache available.

LIST COMM:

LIST COMM lists the two communication MAL queues: one for methods executing via the CMAS, and the other for methods routed to a MAS.

Figure 25 shows an example of the LIST COMM display.
The fields on this display are:

```
COD1 CICSPlex SM Debugger
CMD=>                                                        Scroll=> PAGE
S CSFM MAL      XLTD     Type     Node Type  Target  SysId Sequence
Response List for:  MAL List
_ TSQO 001ABC00 00E1E300 Outbound Local MAS  CMAS1AB  CAB1 00000012
```

Figure 25. An example LIST COMM display

Field **Description**

Response List for:

MAL List
MAL execution to or from a CMAS.

MAS List
MAL execution to or from a MAS.

CSFM The method ID for the MAL being run.

MAL The address of the relocated MAL.

XLTD The address of the MAL descriptor table in the CMAS.

Type The type of communications in progress:

Inbound
The MAL is being run locally from another CMAS.

Outbound
The MAL is being sent to another CMAS.

Response
The MAL response is being transferred.

Node Type
The type of node involved in the transfer:

CMAS
From a CMAS.

Local MAS
From a MAS in the same MVS image.

LIST METH:

LIST METH lists all methods within the specified component.

Figure 26 on page 71 shows a typical example of the LIST METH display.

```
COD1 CICSPlex SM Debugger
CMD=>                                                        Scroll=> PAGE
L=Dump Load Pt  END=Exit.
S Typ Meth Function Fmt Tran LoadPt   ServLevl Assembly Date  Status
_ PUB CWAA ADDTMED  01       077130C0 CPSM510  05/19/98 06.05 ACTIVE
_ PUB CWAB BROTMED  02       07713348 CPSM510  05/19/98 06.05 ACTIVE
_ PUB CWAD DELTMED  03       077138F0 CPSM510  05/19/98 06.05 ACTIVE
_ PUB CWAU UPDTMED  04       07713B40 CPSM510  05/19/98 06.05 ACTIVE
_
```

Figure 26. An example LIST METH display

The fields on this display are:

Field **Description**

Typ The type of method, as either public (PUB) or private (PRV).

Meth The method ID.

Function
The function name of the method.

Fmt The format ID of the method.

Tran If the method runs asynchronously, the CICS transaction ID used.

LoadPt
The load point of the method in memory.

ServLevl
The service level, or release level, of the method.

Assembly Date
The data and time at which the method was assembled.

Status The status of the method as one of the following:

ACTIVE
The method is loaded.

LOCK The method cannot be run locally. Either the method load detected errors or the method does not run in this environment.

NOTFND
The method is not in the load table for the specified release level of the CMAS or MAS.

NOTRAN
The transaction listed in the Tran field is not defined to CICS.

TRAP1
Trap level 1 is set for this method.

TRAP1-2
Trap levels 1 and 2 are set for this method.

TRAP1-32
Trap levels 1 – 32 are set for this method.

LIST START:

LIST START lists the status of all methods you've started or attached.

Figure 27 on page 72 shows an example of the LIST START display.

```
COD1 CICSPlex SM Debugger
CMD=>                                                        Scroll=> PAGE
Select P=Purge V=View MAL D=Dump MAL  END=Cont.
S Type   Meth Task N Status
_ START  XQCQ   828  Completed, RESPONSE:OK
_ START  NSCR   844  Completed, RESPONSE:OK
_ START  NQPG   860  Completed, RESPONSE:EXCEPTION(ABEND)
```

Figure 27. An example LIST START display

The fields on this display are:

Field **Description**

Type Either START or ATTACH, depending on which command you used to start the method.

Meth The name of the method.

Task The CICS task number of the method.

Status The method's status as one of the following:

- Waiting for method to start or attach.
- Method is running.
- Completed, RESPONSE:<response>(<reason>).
- Method is no longer running!

Note: The error "Method is no longer running!" means the status in an internal table indicates the method should be running but the CICS task has been found not active via a CICS inquiry. This error is also used for attached tasks that may have timed out trying to communicate a request back into the CMAS.

You can enter the following in the selection field:

Command **Description**

P Purges the MAL for this method.

Note: You cannot purge a MAL unless its status is "Completed".

V Formats the MAL.

D Calls the CODB transaction with the address of the MAL for hexadecimal dumps.

A Causes an ATTACH command to be created for the method with the context and scope of the original attach being viewed. The existing MAL is used as a starting point, but a new task will appear on the LIST START display.

E Causes an EXEC command to be created for the method with the context and scope of the original attach being viewed. Executed methods do not appear in the LIST TASK display; they are called directly by COD0 and the results are displayed immediately.

S Causes a START command to be created for the method with the context and scope of the original start being viewed. The existing MAL is used as a starting point, but a new task will appear on the LIST START display.

LIST STCB:

LIST STCB lists the server-client control blocks.

Lists the server-client control blocks. These control blocks are used by CICSPlex SM communications and the end-user interface to request work in a CMAS. Figure 28 shows an example of the LIST STCB display.
The fields on this display are:

```
COD1 CICSPlex SM Debugger
CMD=>                                                    Scroll=> PAGE
V=View MAL  D=Dump MAL  S=Dump STCB  X=Dump XLSP
S Address  Status  Last Usr From Error  CSFM Context  Scope  OutQue
_ 060C9BA0 Avail            COM      0  TSPV PLEX2C1  PLEX2C1  NO
_ 060C97C0 Avail            COM      0  TSCV                   NO
_ 060C93E0 Avail            COM      0  CSAC                   NO
```

Figure 28. An example LIST STCB display

Field Description

Address
The address of the STCB control block.

DataAlet
The status of the control block as one of the following:

Avail Available and waiting for work.

Free Available, but not waiting for work.

In Use
A MAL is being run.

Timeout
A conversation with the STCB timed out.

Last Usr
User ID associated with request if available.

From Where the request for this STCB came from: COM, for Communications.

CSFM The ID of the method last run using this STCB.

Context
The CMAS or CICSplex involved in the last request.

Scope The CICSplex, CICS system group, or CICS system involved in the last request.

OutQue
Indicates whether the status of each was reported individually or combined into a single response.

LIST TASK:

LIST TASK displays all CICSPlex SM tasks and the methods being called within them.

Displays all CICSPlex SM tasks and the methods being called within them. Figure 29 on page 74 shows an example of the LIST TASK display typical of those produced under CICS.

```
COD1 CICSPlex SM Debugger
CMD=>                                                        Scroll=> PAGE
DUMP L=Loadpt P=oPb O=Ossb S=Stack M=Mal B=modB E=moEb V=MAL END=Cont.
S Task # METH Load-pt  oPb      Ossb     Stack    Mal      modB     moEB
_    27  XLOP 00000000 00489FA8 00489FF0 0048A014 04283580 000CF820 00000000
_    27  DBG1 8A5B9690 00489FA8 00489FF0 0048A118 0A4602E4 000CF820 00000000

_    20  XLEV 00000000 00494FA8 00494FF0 00495014 04273580 000CF820 00000000

_    23  XLOP 00000000 00491FA8 00491FF0 00492014 04277580 000CF820 00000000
_    23  TIST 0A55C430 00491FA8 00491FF0 00492118 00490FD4 000D7D00 04274160
_    23  XSWC 0A514018 00491FA8 00491FF0 004925A0 0049244C 000D29F4 000DE0B0

_    24  XLOP 00000000 0048DFA8 0048DFF0 0048E014 0427F580 000CF820 00000000
_    24  RSWT 0A574728 0048DFA8 0048DFF0 0048E118 0048CFD4 000DAED4 042744D0
_    24  XSWC 0A514018 0048DFA8 0048DFF0 0048E4D8 0048E3BC 000D29F4 000DE0B0
```

Figure 29. An example LIST TASK display

This display shows one line per method with a space between CICSPlex SM tasks. The fields on this display are:

Heading
Description

Task # The CICS task number.

Note: Do not use the task number to purge the CICSPlex SM transaction, as CICSPlex SM recovery will not be entered and CICSPlex SM system control block chains will be destroyed.

METH
The name of the method running at that stack level.

Load-Pt
The address of the method's load point.

oPb The address of the object process block (one per CICSPlex SM CICS task) that points to all the OSSBs for this task.

Ossb The address of the stack segment block to which this method's stack is attached.

Stack The address of the method's stack.

Mal The address of the MAL for the method.

modB The address of the MODB for the component.

moEB The address of the MOEB for the component.

You can enter the following in the selection field:

Command
Description

L Calls CODB to display the load point of the method.

P Calls CODB to display the OPB.

O Calls CODB to display the OSSB.

S Calls CODB to display the stack.

M Calls CODB to display the MAL.

B Calls CODB to display the MODB.

E Calls CODB to display the MOEB.

V Formats the MAL display as you would have if you entered it.

U Allows updating of the MAL in-flight.

R Lists the contents of all the registers (AR and GP). From this list you can enter:

- **D** Calls CODB to display data at that location using the AR register.
- **A** Calls CODB to display data at that location using only the general purpose register (ALET will be zero).

CICSPlex SM chain checking:

During a LIST TASK command the entire chain of CICSPlex SM blocks that apply to a task are followed.

The eyecatcher for each of the blocks is checked, in addition to the forward and backward methods within stacks and possible recursive chains. If any errors are found, you may see one of the following error messages after the last valid entry:

```
Stack chain broken at AAAAAAAA
```

This error indicates that the stack chain ID of the previous method does not match the method's ID. This may be the case if code within the method overlays the stack header. AAAAAAAA is the address of the invalid stack frame.

```
OPB chain error at AAAAAAAA
```

Object process blocks are created for each CICS CICSPlex SM task. They are chained together for the component ID of the first method in the chain. If this chain points back to itself (a recursive chain), this message appears. AAAAAAAA is the address of the OPB that was next after the previously displayed OPB.

```
Eyecatcher failed for CSFM at AAAAAAAA
```

If an eyecatcher of a control block that is visited during a LIST TASK is incorrect, this message appears. AAAAAAAA is the address of the control block in question and CSFM is its name.

```
DFHEIBLK block invalid at AAAAAAAA, OPB at AAAAAAAA
invalid
```

The task's object process block is really the CICS DFHEISTG area. This area contains a pointer to the task's CICS EIB block, which is checked during LIST TASK commands.

POST (posting an ECB)

The POST command posts an ECB using the MVS POST command.

The format of the POST command is:

```
POST address
```

where *address* is a 1- to 8-character hexadecimal number that is the address at which the ECB resides.

No check is made to see whether an ECB exists at this address or whether it is already posted; an MVS POST command is issued.

Note: You can use the DUMP command or the CODB transaction to find the address.

PRINT (printing data areas under CICSPlex SM)

The PRINT command prints a CICSPlex SM data area. PRINT uses the CICS spool facility to write the data area as an output file called Sxxxxxxx, where xxxxxxx is a numeric identifier.

The format of the PRINT command is:

```
PRINT [parameters...]
```

where the parameters are:

/resname
Prints the specified allocated resource.

alet addr size
Prints an EPTR at the specified address for the specified number of bytes.

CLIST *token*
Prints the cache list of the specified token, where *token* is an 8-byte token entered as two 8-character hexadecimal fields.

EIB *compid*
Prints the EIB for the specified component.

EIS *compid*
Prints the EIS for the specified component.

hexaddr size
Prints memory at the specified address for the specified number of bytes.

MAL *addr*
Formats and prints the MAL at the specified address.

method
Prints the code for the specified method.

MODB *compid*
Prints the MODB for the specified component.

MODD
Prints the MODD.

MOEB *compid*
Prints the MOEB for the specified component.

OPB *compid*
Prints the object process block for the specified component.

QUE *token*
Prints the queue of the specified token, where *token* is an 8-byte token entered as two 8-character hexadecimal fields.

STAKEND *compid*
Prints the current stack for the specified component.

STAKSTRT *compid*
Prints the first stack for the specified component.

XLWA
Prints the XLWA.

PURGE (purging an allocated resource)

The PURGE command purges an allocated resource.

The format of the PURGE command is:

```
PURGE /resname
```

where:

/resname
Is the name of the resource you allocated. The storage assigned to the resource is removed from the system.

Note: You can also purge allocated storage using the P command from the LIST ALLOC display.

START (starting a method in the CMAS)

The START command starts a method running within the CMAS.

The format of the START command is:

```
START method [termid]
```

where:

method
Is the name of a CICSPlex SM method.

termid
Is a terminal ID.

The message argument list (MAL) of the method is displayed. For details about how to enter data from this display, see "Displaying a MAL from COD0" on page 79. For a list of valid responses to this command, see "Running a method" on page 82.

Figure 30 shows an example of the START display.
The START command starts a CICS task that eventually executes method DBG2.

```
COD1 CICSPlex SM Debugger
CMD=>                                                        Scroll=> PAGE
Overtype fields and press ENTER to edit, END to proceed, CANCEL to abort.    Dn
IN
    ENM FUNCTION( SETCRGN )
    CHR DEBUG(          )
    BIN SYSTEM_AKP(          )
    BIN SYSTEM_AMAXTASKS(          )
    BIN SYSTEM_CUSHION(          )
    CHR SYSTEM_DTRPROGRAM(          )
    BIN SYSTEM_ECUSHION(          )
    BIN SYSTEM_MAXTASKS(          )
    BIN SYSTEM_MROBATCH(          )
    BIN SYSTEM_PRTYAGING(          )
    BIN SYSTEM_RUNAWAY(          )
    BIN SYSTEM_SCANDELAY(          )
    BIN SYSTEM_SYSDUMP(          )
    BIN SYSTEM_TIME(          )
    BIN TRACEDEST_AUXSTATUS(          )
    BIN TRACEDEST_GTFSTATUS(          )
    BIN TRACEDEST_INTSTATUS(          )
    BIN TRACEDEST_SWITCHSTAT(          )
    BIN TRACEDEST_SWITCHACT(          )
```

Figure 30. An example START display

This method is created dynamically by COD0 in every component.

TRACE (setting CICS and CICSPlex SM trace flags)

You use the TRACE command to set CICS and CICSPlex SM component trace flags, and to control output to auxiliary trace data sets.

The format of the TRACE command is:

```
TRACE [parameters...]
```

where the parameters are :

ON [RESET|START]
OFF [RESET|STOP]
USER [RESET]
Controls the settings of the CICS component trace flags.

ON Turns all CICS component flags on, which produces slightly more output than the normal CICS trace settings.

OFF Turns all CICS component trace flags off, which results in almost no output at all (some CICS components do not have trace flags).

USER Traces only the CICS component application domains (AP0000 through APFFFF).

RESET
Causes tracing to start at the beginning of the auxiliary trace data set, overwriting any existing output.

START
Opens the auxiliary trace data set.

STOP Closes the auxiliary trace data set.

FLAG
Shows the trace flags of each CICSPlex SM component. You can change the trace flag settings of one or more CICSPlex SM components by overtyping the component's bit setting.

SWITCH
Switches the CICS auxiliary trace data sets and reports on which is active.

Changes made to CICS and CICSPlex SM trace settings from the COD0 debugging transaction remain in effect after you exit the transaction.

TRACK (setting trace flags by calling structure)

The TRACK command sets CICSPlex SM trace flags based on the calling structure.

The format of the TRACK command is:

```
TRACK target relation calling flags id
```

where the parameters are:

target
The name of the method to be traced. You can provide a generic method name by specifying an asterisk (*) at the end of the name or in place of the name (to indicate all methods).

relation
The relationship to the calling method as one of the following:

FROM
Sets the trace for the target only when the direct caller is the calling method.

STAK Sets the trace for the target only if the calling method is somewhere in the CICSPlex SM stack.

calling
The name of the method that calls the target method either directly or indirectly. You can provide a generic method name by specifying an asterisk (*) at the end of the name or in place of the name (to indicate all methods).

flags
The trace flags to be set. The trace flags are set according to group names and are dependent on the underlying trace facility. The trace flags will be provided by IBM support should you need to use this facility.

id An optional user or task ID:

Uxxxxxxx
where xxxxxxx is a 1-to 8-character user ID.

Tnnnnnnn
where nnnnnnn is a 1- to 7-position CICS task number that can be obtained by issuing either the LIST TASK or CEMT INQ TASK command.

For example:

```
TRACK XD* STAK CI* SPEC
```

activates all trace flags for any data repository method that is called directly or indirectly from any communication initialization method.

TRAP (setting trace flags for a method)

The TRAP command sets trace flags on for a specific CICSPlex SM method.

The format of the TRAP command is:

```
TRAP method [1|2vALLvOFF]
```

where:

method
Is the name of a CICSPlex SM method.

1|2vALLvOFF
Sets the trace flags for the specified method:

1 Sets level 1 trace flags on.

2 Sets level 1 and level 2 trace flags on.

ALL Sets level 1–32 trace flags on.

OFF Sets tracing for the method back to the flags specified on the COD0 TRACE command, the EYUPARMS start-up parameters, or the CMAS or MAS view command.

Displaying a MAL from COD0

When entering into a MAL formatted by the COD0 debugging transaction, all input is validated for both physical and logical properties.

Format of the MAL display:

When viewing or updating a MAL, either from a START, ATTACH, or EXEC command, or from LIST output the format of the display is as follows.

```
COD1 CICSPlex SM Debugger
CMD=>                                                    Scroll=> PAGE
Fields with "->" required.
IN
   *ENM FUNCTION( TEST )
    CHR DEBUG(          )
->  FLG FLAG_VALUES(              )
OUT
   *ENM RESPONSE(                 )
   *ENM REASON(                )
```

Figure 31. Sample formatted MAL display

As shown in Figure 31, IN and OUT eyecatchers separate the major sections of the MAL. Each field name in the IN and OUT sections can be preceded by three other indicators:

- An arrow, indicating the field is mutually required or mutually exclusive and in error.
- An asterisk, indicating the existence bit for the field is set on in the MAL (OUT fields always have the existence bit on).
- A 3-character code indicating the field type.

Note: The FUNCTION field is completed by the COD0 debugging transaction and cannot be changed.

Field types:

The three-character code that precedes a field determines what can be entered in the field and the kind of data that is displayed.

Table 5 shows the input allowed for each field type.

Table 5. Field types

Type	Format	Input allowed
BIN	BIN(n)	Hexadecimal number
BLK	BLOCK	Hexadecimal number for address or decimal number for length
BUF	BUFFER	Hexadecimal number for address, decimal number for length, or resource name (/resname)
CHR	CHAR(n)	Any character
CMP	COMPID	Component ID or '?' for a list
DEC	DEC(n)	Decimal number
EBK	EBLOCK	Hexadecimal number for ALET and OFFSET, decimal number for length, or resource name (/resname)
ENM	(names)	Names defined in format or '?' for list
EPT	EPTR	Hexadecimal number for ALET and OFFSET or resource name (/resname)
ETK	ETOKEN	Hexadecimal number or resource name (/resname)
FLG	FLAG	Hexadecimal representation of a flag or '?' for a list
LST	LIST	Hexadecimal number for address, decimal number for length, or resource name (/resname)
MPL	MAL	Hexadecimal number or resource name (/resname)
PTR	PTR	Hexadecimal number or resource name (/resname)
RES	RESTYPE	Resource name (/resname) or '?' for a list
SDT	SDT	TRUE or FALSE
STR	STRING(n)	Any character

Table 5. Field types (continued)

Type	Format	Input allowed
TIM	TIMESTAMP	Hexadecimal number
TKN	TOKEN	Hexadecimal number or resource name (/resname)

Field edits and display formats:

In a MAL display, input is edited and output is formatted according to certain rules.

Field type
Format

Hexadecimal
Hexadecimal characters 0–9 and A–F, in either upper or lower case. In output, the number is right-justified and padded with zeroes.

You can enter decimal characters instead of hexadecimal by preceding the value with a backslash, as in \1234. The decimal number is internally converted to hexadecimal.

Decimal
Numeric characters 0–9, without any sign. In output, the number is right-justified and padded with zeroes.

You can enter hexadecimal characters instead of decimal by preceding the value with a backslash, as in \ABCD. The hexadecimal number is internally converted to decimal.

ENM, CMP, or RES
One of the values shown in the message format. For example, if the format indicates MY_FIELD IS (A,B,C,D), you can enter A, B, C, or D. You can also enter a question mark (?) to display a list of possible values; you can select one to be copied into the MAL.

For a field type of RES, you can use the HELP command to display a list of known resource types, such as HELP RESOP or HELP CVDA.

Flags A hexadecimal value representing a flag name, including a combination of flag names that have been logically ORed. You can also enter a question mark (?) to display a list of possible values. You can select as many as apply; they are logically ORed and copied into the MAL.

Subfields:

Many fields in a MAL consist of multiple subfields, which are divided into multiple input fields and validated separately.

Each field is preceded by the suffix of the subfield. For example, the EPT field is made up of the ALET and OFFSET subfields, and looks like this when the MAL is displayed:

```
EPT YOUR_MAL_FIELD_NAME(A= alet O= offset)
```

The subfields associated with each field type are as follows:

Field type
Subfields

EPT A=alet, O=offset

BUF A=address, L=length, M=maximum length

BLK A=address, L=length

EBK A=alet, O=offset, L=length

LST A=address, N=number

Using allocated resources:

You can use the names of allocated resources (such as cache lists, data queues, data space storage, or CICS storage) in the input fields of a MAL.

If the field contains subfields, as described in "Subfields" on page 81, you need enter only the resource name in the first field; the COD0 debugging transaction determines the other field types and fills them in for you. For information on allocating resources, see "ALLOC (allocating a resource)" on page 56.

For example, you could use the ALLOC command to allocate 4K of data space storage to the resource called /workara, as shown in Figure 32.
Figure 33 shows the allocated resource, /workara, being used as input to a MAL.

```
COD1 CICSPlex SM Debugger
CMD=> ALLOC /workara EPTR 4096                                  Scroll=> PAGE

IN
```

Figure 32. Using ALLOC to allocate a resource

The COD0 debugging transaction places the ALET of the allocated storage area

```
COD1 CICSPlex SM Debugger
CMD=>                                                           Scroll=> PAGE

IN
   *ENM FUNCTION( TEST )
    CHR DEBUG(          )
    EPT OUT_ADDR(A= /workara   O=         )
```

Figure 33. Using an allocated resource in a MAL

into the A= field; the offset is automatically entered in the O= field.

Running a method:

From a formatted MAL display you can enter certain commands.

These commands are:

CANCEL
Returns control to the previous display without processing the MAL.

DUMP *mal-field*
Determines the type of the specified field and creates an appropriate DUMP command to call the CODB debugging transaction. *mal_field* can be any field on the formatted MAL display.

END Edits the MAL and then either ATTACHes, EXECutes, or STARTs the method. Control returns to the previous display.

If you return to the COD0 main menu while a method is running, a LIST START command is automatically issued.

FLAG *mal-field*
Displays a list of the specified field's bit values (that is, their names from the Message Argument Format). Those that are currently set are prefixed by a plus sign (+). *mal-field* can be any field on the formatted MAL display that has a type of FLG. If the name you enter is neither part of the MAL nor an FLG field, an error message is issued.

GO Edits the MAL and then either ATTACHes, EXECutes, or STARTs the method. Control remains at the formatted MAL display. You can enter the same or different data, and issue GO or END again.

You can use the LIST or LIST START command to check the progress of the started or attached method. When you END the LIST display control returns to the formatted MAL display.

NEXT For the results of an ATTACHed method that either ran in multiple MASs or ran multiple times in a CMAS, displays the MAL that ran next.

PREV For the results of an ATTACHed method that either ran in multiple MASs or ran multiple times in a CMAS, displays the MAL that ran previously.

Note:

1. If you press Enter without issuing a command, the MAL is edited, but not run.
2. For the DUMP and FLAG commands, only fields that appear in the current formatted MAL display can be used as parameters. If you want to name the field of another MAL, you must first display that MAL from the LIST START or LIST TASK screen.

System-level debugging with CODB

The CODB debugging transaction allows you to display and modify memory.

It is menu-driven and allows you to choose various CICSplex data areas using PF keys or command line keywords.

After logging onto CICS, enter the CODB transaction ID to display the main menu, as shown in Figure 34 on page 84. (CODB can also be entered from the DUMP command of the COD0 transaction.)

```
COMMAND==>           COMP ID==>      ADDR==>          ALET==> 00000000

         1.  XLWA
         2.  MODB
         3.  MOEB
         4.  OPB
         5.  EIS
         6.  EIB
         7.  STAKSTRT
         8.  STAKEND
         9.  MODD
        10.  MAL
        11.  PFKON
        12.  PFKOFF
        13.  END
        14.  CMASSTOP
        15.  TRACE
        16   QUES
        17   MENU

P1=TOP  P2=BOTM  P3=END  P4=PREV  P5=NEXT  P6=TOKEN  P7=BACK  P8=FRWD
P9=JUMP  P10=DSJUMP  P11=ALTER  P12=ALET/OFFSET
MSG==>
```

Figure 34. CODB debugging transaction menu

The first field is for the command, the second is for a component ID (which is required for some commands), the third is for the address (or AR mode offset), and the last is for an ALET or zeros.

Note: The CODB menu can be redisplayed at any time by issuing the MENU command.

CODB commands

Any CODB command shown on the menu, or its associated number, is valid at any time.

Some commands (such as MODB and MOEB) display a submenu listing the component ID and the address of the requested control block, if it can be located. The command name remains displayed until it is replaced by a new command, or a memory display is requested.

Command
Description

XLWA Sets the ADDR==> field to the CMAS or MAS external linkage work area (XLWA) and the ALET==> field to zero, and displays the CICSplex anchor block.

MODB
Displays the major object descriptor block (MODB) for the specified component.

MOEB
Displays the major object environment block (MOEB) for the specified component.

OPB Displays the first object process block (OPB) for the specified component.

EIS Displays the CICS EXEC interface storage (EIS) block for the specified component.

EIB Displays the CICS EXEC interface block (EIB) for the specified component.

STAKSTRT
Displays the first stack of the first transaction running for the specified component.

STAKEND
Displays the current stack of the first transaction running for the specified component.

MODD
Displays the major object director descriptor (MODD) block for the specified component.

MAL Displays the MAL currently initialized in the first transaction running for the specified component.

PFKON
Turns on the PF key prompts at the bottom of the screen.

PFKOFF
Turns off the PF key prompts at the bottom of the screen.

END Exits the CODB transaction.

CMASSTOP
Shuts down the CMAS by posting the termination ECB.

TRACE
Displays unformatted CICS internal trace table.

QUES Displays the queue token anchor block.

MENU
Redisplays CODB menu.

Note: The MODB, MOEB, OPB, EIS, EIB, STAKSTRT, STAKEND, MODD, and MAL commands require a component ID, as described in "The COMP ID field."

The MENU command can be issued at any time to redisplay the CODB menu.

The COMP ID field

CODB commands that display CICSPlex SM control blocks (such as MODB and MOEB) require you to specify a three-character component ID in the COMP ID field.

For a list of valid component IDs, see Chapter 10, "Major components of CICSPlex SM," on page 113.

When you have specified a component ID, it remains displayed until one of the following occurs:

- A new component ID is specified.
- The COMP ID field is blanked out.
- A memory display is requested.
- A command is entered that does not require a component ID.

So it is possible to display various control blocks belonging to a single component by establishing the component ID and then issuing different commands.

The ADDR field

Entering a value in the ADDR field produces a display of memory at the specified address, using the current ALET.

If the address cannot be accessed, a message appears in the MSG field at the bottom of the display.

Relative addressing is also supported in the ADDR field. You can enter a scroll amount, in bytes, as a signed (+ or -) hexadecimal number. For example:

```
ADDR==> +2D0
```

The ALET field

Entering a value in the ALET field sets the ALET value to be used for memory displays.

This field is normally filled in; it has an initial value of hexadecimal zeros.

The function key prompts

The function key prompt area contains a two-line list of the function keys supported and a brief description of their values.

This prompt can be turned off by the PFKOFF (12) command and turned back on by the PFKON (11) command.

The following function keys are in effect while the CODB transaction is running:

Key	Description
F1	TOP (valid only for control block displays). Repositions the display to the beginning of the control block. If the display was produced by a value in the ADDR field, a warning message appears in the MSG field.
F2	BOTTOM (valid only for control block displays). Repositions the display to the end of the control block. If the display was produced by a value in the ADDR field, a warning message appears in the MSG field.
F3	END. Exits the CODB transaction.
F4	PREV. Depending on the contents of the current display, displays the previous control block of the same type or the previous cache list or queue record. For a control block display, PREV is both command (control block) and component sensitive. If a submenu from a control block command is displayed, PREV displays the last component's control block, if it exists; if it does not exist, a warning message appears in the MSG field. If a component's control block is displayed, the previous component's control block is displayed. For a cache list or queue record display, if you issue PREV when the first record is displayed, a warning message appears in the MSG field.
F5	NEXT. Depending on the contents of the current display, displays the next control block of the same type or the next cache list or queue record. For a control block display, NEXT is both command (control block) and component sensitive. If a submenu from a control block command is displayed, NEXT displays the Kernel Linkage control block, if it exists; if it does not exist, a warning message appears in the MSG field. If a component's control block is displayed, the next component's control block is displayed. When displaying OPBs, NEXT runs down each component's OPB chain, if it exists, before going on to the next component. For a cache list or queue record display, if you issue NEXT when the last record is displayed, a warning message appears in the MSG field.

F6 TOKEN. Displays either the first record of the queue whose QTOKEN is pointed to by the cursor, or the first cache list element whose EPOINTER is pointed to by the cursor. The NEXT and PREV commands can be used to scroll forward and backward through the queue or cache list.

F7 BACKWARD. Scrolls the memory display backward one full page.

F8 FORWARD. Scrolls the memory display forward one full page.

F9 JUMP. Produces a display that starts at the address pointed to by the cursor, using an ALET of zero. The address pointed to can be the address field, the relative address field, the EBCDIC field, or an address in the hexadecimal data display. If the specified memory cannot be accessed, a warning message appears in the MSG field.

If a control block was being displayed, JUMP erases the current command and component ID and establishes the ADDR mode. After a JUMP command, it is possible to scroll beyond the bounds of the control block, even if the address selected is within the block. To reestablish control block mode, the intended command and component ID must be reentered.

F10 DSJUMP. Produces a display that starts at the address pointed to by the cursor, using the displayed ALET. The address pointed to can be the address field, the relative address field, the EBCDIC field, or an address in the hexadecimal data display. If the specified memory cannot be accessed, a warning message appears in the MSG field.

F11 ALTER. Allows you to alter memory.

F12 ALET/OFFSET. Produces a display that starts at the ALET/ADDRESS pair pointed to by the cursor. The ALET/ADDRESS pair must be in the hexadecimal data display and the cursor must be on the ALET portion of the pair. If the specified memory cannot be accessed, a warning message appears in the MSG field.

The MSG field

The MSG field is a one line area headed by: MSG==> that appears on all screens.

The MSG field is used for warning, informational, and error messages.

The memory display area

The memory display area contains hexadecimal and EBCDIC representations of the requested memory ALET/ADDRESS, or the requested control block.

Each line of the display contains an address, its offset from the beginning of the area (either the start of the control block or the address entered in the ADDR field), four full words of data in hexadecimal format, and the EBCDIC representation of those sixteen bytes. Figure 35 on page 88 is a sample CODB memory display.

```
COMMAND==> XLWA       COMP ID==>      ADDR==>           ALET==> 00000000

00077368  00000000  020C6EC5  E8E4E7D3  D2D5D3C3  E6C1C1C2  ..>EYUXLKNLCWAAB
00077378  00000010  01000200  000773E0  00000000  006B2F20  .......\....,..
00077388  00000020  00000000  0000000E  0000E888  00097820  ..........Yh....
00077398  00000030  00077470  0000A81C  0008D000  000003B6  ......y...}.....
000773A8  00000040  00084E90  00040000  043E0000  00000020  ..+.............
000773B8  00000050  043E0000  FFFFF34E  8A680940  006B2F20  ......3+... .,..
000773C8  00000060  00000000  D2D3D7C2  00085310  00085828  ....KLPB........
000773D8  00000070  0A62AD40  0093D154  00077384  000773C4  ... .lJ....d...D
000773E8  00000080  00000000  00000000  00000000  00346EC5  ..............>E
000773F8  00000090  E8E4E7C5  C5E8E4D9  E7C5D3E2  01030000  YUXEEYURXELS....
00077408  000000A0  009AFC38  03C6B150  0093D140  0093D154  .....F.&;lJ .lJ.
00077418  000000B0  00FAB580  006C1258  A458C562  09D66631  .....%..u.E..O..
00077428  000000C0  D7D9D4C2  000774B8  00077528  0007752C  PRMB............
00077438  000000D0  00077530  00077534  00077548  00077544  ................
00077448  000000E0  00077558  0007755C  0007756C  00077560  .......*...%...-
00077458  000000F0  00077568  00077564  00077570  00000000  ................
00077468  00000100  00000000  00000000  00000000  00097820  ................
00077478  00000110  000988BC  00099958  0009A9F4  0009BA90  ..h...r...z4....
00077488  00000120  0009CB2C  0009DBC8  0009EC64  0009FD00  .......H........
00077498  00000130  000A0D9C  000A1E38  000A2ED4  000A3F70  ...........M....
000774A8  00000140  000A500C  00000000  00000000  00000000  ..&;............
000774B8  00000150  A458C562  09D66631  006C1258  00FAB580  u.E..O...%......
000774C8  00000160  000000D4  00000002  E2E8E2C3  C3E5D4C3  ...M....SYSCCVMC
000774D8  00000170  E3E2D6F1  C3E5D4C3  E6404040  C3E6E6F1  TSO1CVMCW   CWW1
000774E8  00000180  DF80FCA0  00800000  00000000  00000000  ................
000774F8  00000190  00000000  00000000  00000000  00000002  ................
00077508  000001A0  04375000  00000000  00000000  00000000  ..&;............
00077518  000001B0  00000000  00000000  00000000  00000000  ................
00077528  000001C0  00000000  00000000  00000000  00000400  ................
00077538  000001D0  00040000  00000002  0000001C  00000000  ................
00077548  000001E0  C5E8E4C4  D9C5D740  00000000  00000000  EYUDREP ........
00077558  000001F0  00000000  00000000  00000000  00000000  ................
00077568  00000200  00000000  00000000  00000000            ............

P1=TOP  P2=BOTM  P3=END  P4=PREV  P5=NEXT  P6=TOKEN  P7=BACK  P8=FRWD
P9=JUMP  P10=DSJUMP  P11=ALTER  P12=ALET/OFFSET
```

Figure 35. Sample CODB memory display

CODB altering memory

The hexadecimal and EBCDIC data portions of the display can be modified.

You can overtype hexadecimal data using valid hexadecimal digits, or EBCDIC data using any keyboard character except the period. After overtyping the data, press PF11 (ALTER).

Note: The CODB alter memory function should be used only at the request of customer support personnel.

A warning message appears in the MSG field if:

- The memory is protected.
- You altered the screen but did not press PF11.
- The memory location being altered has changed since it was displayed.

Trying to modify protected storage causes an abend. The CODB recovery routine issues a message describing the abend to the console.

Accessing CODB from COD0

You can access CODB from the COD0 transaction using the DUMP command or by entering a **D** in a selection field, when allowed.

When you exit CODB (by issuing the END command) you are returned to the COD0 transaction.

There are some advantages to using COD0 to enter CODB:

- The DUMP command translates a method name into its entry addresses so you can dump or alter method code.
- From the LIST TASK screen you can dump individual stacks, MALs, OPBs, and OSSBs, for example.
- You can dump allocated resources (as defined by the ALLOC command) by name, and COD0 translates them into ALET/OFFSETS, ADDRESSES, or TOKENS, as required.
- You do not need to know the exact ALET/OFFSET or ADDRESS of the area you are dumping.

Chapter 8. Investigating and documenting a problem

Troubleshooting techniques help you determine the cause of a CICSPlex SM problem.

They might help you solve some problems yourself. If you cannot resolve a problem yourself, you need to gather the necessary documentation before contacting your IBM Support Center.

Investigating output and system management problems

This information describes some ways of solving typical problems with output and system management results.

If you have problems with unexpected or incorrect output from the WUI, use the following information to investigate the specific problem, and collect the relevant documentation for customer support personnel. Customer support personnel might also ask you to provide screen prints that show the problem.

Investigating abends

Because CICSPlex SM has a presence in two major parts of your environment (MVS and CICS) abends can occur in either place. Some CMAS abends occur under MVS; MAS and other CMAS abends, however, occur under CICS.

Use the information in this section to help you isolate the cause of an abend or to report the condition to customer support personnel.

MVS abends

What CICSPlex SM does

- Passes control to the appropriate recovery routine.
- Produces an SDUMP.
- Writes BBx and EYU messages to the console, job log, and EYULOG.

Documentation to collect

- System console log and EYULOG
- Unformatted SDUMP from the affected address space
- AUXTRACE data set, if available
- Any LOGREC entries

CICS abends

What CICSPlex SM does

- Passes control to CICS, which decides whether to take an SDUMP.
- Regains control from CICS.
- Produces a transaction dump and, possibly, an SDUMP.
- Writes an EYU failure summary to the console.
- Writes EYU messages to the job log and EYULOG.

Documentation to collect

- System console log and EYULOG

© Copyright IBM Corp. 1997, 2017

- Unformatted SDUMP from the affected address space
- AUXTRACE data set, if available

Investigating stalls

When CICSPlex SM doesn't seem to be responding, you should suspect a stall condition, which could be either a loop or a wait.

Note: In the case of a suspected loop or wait, you should request an SDUMP; CICSPlex SM will not take one automatically. However, do not cancel the task that appears to be stalled before requesting the dump. If you cancel the task, the CICS and CICSPlex SM recovery routines that get control will change the "picture" taken by the dump and you may lose valuable diagnosis information.

You will need to determine both at what stage of processing the stall occurred and where it occurred. Processing a CICSPlex SM request involves multiple address spaces. The process could stall in the TSO/ISPF session, in any of the CICS systems included in the current context and scope, or at any of several points in between.

Use the information in this section to help you isolate the cause of a stall or to report the condition to customer support personnel.

An undetermined stall condition

Ask these questions when you are dealing with a stall condition.

Questions to ask

- Did the stall occur during initialization?
  - How far did initialization progress?
  - Were there any definition or setup errors reported?
- Did the stall occur during operation?
  - Are the necessary communication links between CMASs and MASs available?
  - What type of request was being processed?
  - How big was the CICSplex involved?
  - How many CMASs and MASs were involved?
  - What types of monitoring, real-time analysis, and workload management were active?
- Did the stall occur during termination?
- Did the stall occur in a CMAS?
  - Did the request time out with an EYUEInnnn message?

    The local CMAS may be waiting for one or more CICS systems (or their CMAS) to return requested data. A CICSPlex SM view does not return until all the expected data is collected.
  - Did the request time out with a CICS message?
- Did the stall occur in a MAS?

  Try stopping the MAS agent code (using the STOP action command from the MAS view), then evaluate the underlying CICS system.
  - Is the CICS system taking an SDUMP?
  - Is the CICS system looping or hung?
  - Did the request time out with a CICS message?

– Is the CICS system experiencing a short on storage (SOS) condition, or has it reached its MAXTASK level?

Any one of these conditions could prevent some types of CICSPlex SM requests from completing.

Documentation to collect

- System console log and EYULOG
- CMAS job logs
- Unformatted SDUMP from the affected address spaces (TSO, CMAS or MAS)

A suspected loop

Ask these questions when you are dealing with a loop.

Questions to ask

- What are some possible sources of the loop?
- Is CPU usage particularly high?

Documentation to collect

- Appropriate job logs
- Selected trace data, as requested by support
- AUXTRACE data set, if available
- Transaction dump, if any
- CICS system dump, if any

A suspected wait

Ask these questions when you are dealing with a wait condition.

Questions to ask

- At what point is the wait occurring?
- Is CPU usage particularly low?

Documentation to collect

- Appropriate job logs
- Appropriate CICS CEMT queries
- Selected trace data, as requested by support
- AUXTRACE data set, if available
- Transaction dump, if any
- CICS system dump, if any

An unformatted dump is the preferred source of problem diagnosis information for a stall. You should format a CICSPlex SM dump only at the request of customer support personnel.

Investigating bottlenecks

Bottlenecks can be caused by various components of CICSPlex SM.

You need to be aware of how these components are defined and how they interact, as well as of the transactions underway when the bottleneck occurs.

Use the information in this section to help you isolate the cause of a bottleneck or to report the condition to customer support personnel.

Questions to ask

- What type of request was being processed?
- How big was the CICSplex involved?
- How many CMASs and MASs were involved?
- What types of monitoring, real-time analysis, and workload management were active?
- What are the dispatching priorities of the CMASs and MASs?

  The priority of a CMAS must be higher than that of the MASs it manages.
- Are the CICS SIT parameters correctly specified for the CMASs and MASs?
- How is the communications network performing?

Documentation to collect

To diagnose a performance problem such as a bottleneck, customer support personnel may ask you to turn on trace level 16 in selected CICSPlex SM components. Many components use trace level 16 to determine how long a request takes to complete. It may be possible, based on that data, to isolate the problem to outgoing or incoming processes. For information on controlling the trace levels in CICSPlex SM components, see "Controlling the amount of tracing in a CMAS or MAS" on page 29.

Incomplete operations data returned

Consider this example of incomplete data that is returned.

There is a CICS system known as EYUMAS1A with a context of EYUPLX01 and a scope of EYUCSG01. EYUMAS1A has been installed as a MAS and is currently running. EYUMAS1A should appear on the **CICS regions** view (CICSRGN object). However, EYUMAS1A is missing from the tabular view. To access this view, from the main menu, click **CICS regions**.

A good first step to determine what is wrong is to look at the **MASs known to CICSplex** view, using the same context (EYUPLX01) and scope (EYUCSG01) as the failing **CICS regions** view. To access the **MASs known to CICSplex** view, from the main menu, click **CICSPlex SM operations views > MASs known to CICSplex**. The **MASs known to CICSplex** view will show one of the following conditions:

- There is no entry for EYUMAS1A.
- The entry for EYUMAS1A shows a status of INACTIVE.
- The entry for EYUMAS1A shows a status of ACTIVE.

No entry for EYUMAS1A

There are three things to check if there is no entry for EYUMAS1A in the **MASs known to CICSplex** view.

1. Ensure that the CICS system definition (CSYSDEF object) exists in the data repository for the current context.
2. Ensure that the scope EYUCSG01 is correct. If EYUMAS1A is not a member of the CICS system group EYUCSG01, the scope is incorrect. To test that possibility, change the scope. You can do this in two ways:
   - On the main menu, on the main menu amend the **Scope** field and click the **Set** button. This sets the context for the CICSplex. Redisplay the **MASs known to CICSplex** view and check that EYUMAS1A is now present.

- On the **MASs known to CICSplex** view, amend the **Scope** field and click the **Refresh** button. This sets the context for the **MASs known to CICSplex** views only. Check that EYUMAS1A is present on the refreshed **MASs known to CICSplex** view.

3. Ensure that the context EYUPLX01 is correct. EYUPLX01 should have been the context when the CICS system definition (CSYSDEF object) for EYUMAS1A was created. If it was not, correct the context. You can do this in two ways:
   - On the main menu amend the **Context** field and click the **Set** button. This sets the context for the CICSplex. Redisplay the **MASs known to CICSplex** view and check that EYUMAS1A is now present.
   - On the **MASs known to CICSplex** view, amend the **Context** field and click the **Refresh** button. This sets the context for the **MASs known to CICSplex** views only. Check that EYUMAS1A is present on the refreshed **MASs known to CICSplex** view.

INACTIVE status

Whenever either a MAS or a CMAS is started, CICSPlex SM attempts to activate communication between the MAS and the CMAS. If both the CMAS and the MAS are running and the status on the **MASs know to CICSplex** view shows INACTIVE, you need to look at the JESMSGLG of the MAS and the EYULOG of the CMAS.

They may contain messages indicating that the connection process failed and suggesting what could be wrong.

It could be that the CICS system definition name does not match the EYUPARM parameter NAME in the startup JCL for the MAS. It is also a possibility that, if the default for the EYUPARM NAME is taken, EYUMAS1A is not the z/OS Communications Server APPLID. Here is an example of the JESMSGLG of the MAS when the NAME parameter is incorrect:

```
DFHSI1517 EYUMAS1A Control is being given to CICS.
EYUXL0003I EYUMAS1A  CPSM Version 320 LMAS startup in progress
EYUXL0022I EYUMAS1A  LMAS Phase I initialization complete
EYUXL0004I EYUMAS1A  ESSS connection complete
EYUCL0112E EYUMAS1A  Protocol Services initialization unable to perform ICT Attach
EYUCL0101E EYUMAS1A  Protocol Services initialization failed
EYUCI0101E EYUMAS1A  Communications initialization failed
EYUXL0112E EYUMAS1A  LMAS initialization failed
```

Figure 36. Example of JESMSGLG when EYUPARM NAME parameter is incorrect

The EYUPARM parameter CICSplex in the startup JCL for the MAS may not match the CICSplex name being used as the context for the **MASs know to CICSplex** view. If the CICSplex named in the EYUPARM is valid, the MAS probably connected successfully to that CICSplex, instead of to the CICSplex used as the context for the **MASs know to CICSplex** view that shows INACTIVE.

If SEC(NO) is coded in the EYUPARM parameters for a CMAS, and SEC(YES) is coded for a MAS that is connecting to that CMAS, the attempt to establish the connection between the CMAS and the MAS fails. The following message appears in the EYULOG of the CMAS:

```
EYUCR0007E 'Security mismatch between CMAS EYUCMS1A and MAS EYUMAS1A  .
           Connection Terminating.'
```

It is also possible to terminate the connection between a CMAS and a MAS using the **Stop** button on the **MASs know to CICSplex** view.

The preceding causes of the INACTIVE status have not dealt with the case where a CICSplex is managed by multiple CMASs. Consider the CICSplex shown in Figure 37.

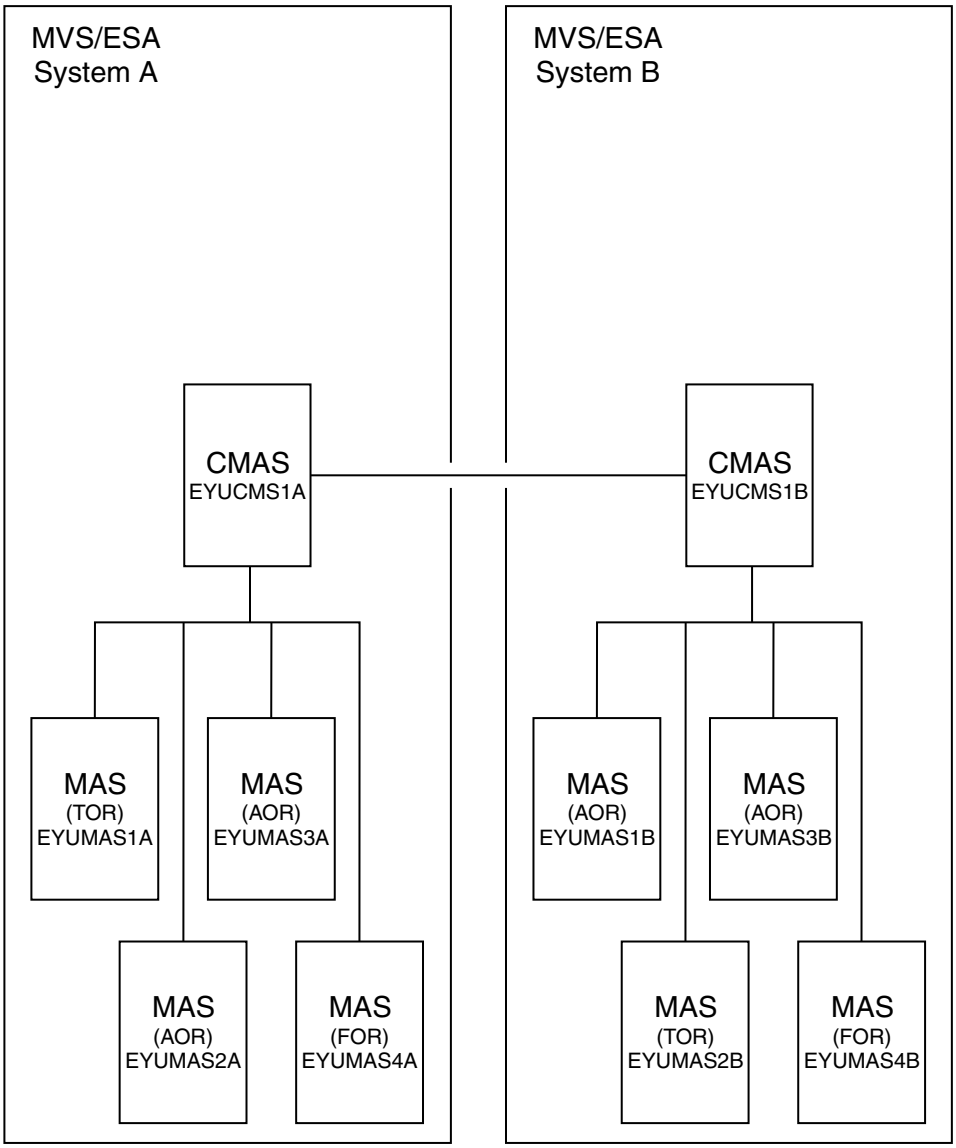


Figure 37. Example of a CICSplex managed by multiple CMASs

Let's say you are connected to CICSPlex SM with a context of EYUPLX01 and your server CMAS is EYUCMS1A. Set the context for the CICSplex on the main menu, using the following values:

```
CMAS context: EYUCMS1A
Context:      EYUPLX01
```

You know that all eight MAS regions are running, yet the **MASs known to CICSplex** view with a scope of EYUPLX01 shows an ACTIVE status for EYUMAS1A, EYUMAS1A, EYUMAS1A, and EYUMAS1A, but an INACTIVE status for EYUMAS1B, EYUMAS2B, EYUMAS3B, and EYUMAS4B.

In general, the CMAS serving a WUI request must have connectivity to the CMAS to which a MAS is connected; if it does not, that MAS does not appear active to the WUI.

The **CMASs managing CICSplex** view shows (from the perspective of one CMAS) the connectivity to the other CMASs managing a CICSplex. To access this view, from the main menu, click **CICSplex SM operations view > CMASs managing CICSplex**. On the **CMASs managing CICSplex** tabular view, set the context to the CMAS that is serving your WUI session (EYUCMS1A) and click the **Refresh** button.

If the **CMASs managing CICSplex** tabular view shows a CMAS with INACTIVE status, but you know that CMAS is running, you must investigate the communication links. CMAS-to-CMAS communication uses CICS services. Therefore, the MSGUSR log is likely to contain information concerning the nature of the communication failure.

ACTIVE status

An ACTIVE status indicates that a MAS is properly connected to the CICSplex. There should be no problem with missing data.

Missing monitor data

You might not receive monitor data from one of the Monitoring views for several reasons.

For example, you have a context of EYUPLX01 and a scope of EYUMAS1A, and you want to display data about intrapartition data queues. From the main menu, click **Monitoring views** > **Transient data queues (TDQ) monitoring views** > **Intrapartition**. However, the "Monitor data for intrapartition transient data queues" view (MNTRADQ object) shows no data.

Use the following procedure to resolve this problem:

- Set the scope to the CICS system from which you are receiving no monitor data. On the main menu, amend the Context field and click **Set**. This sets the context for the CICSplex.
- From the main menu, click **Monitoring views** > **Active monitor specification** to display the "Active monitor specifications" tabular view (POLMON object). See Table 6.

Table 6. Representation of data in an "Active monitor specifications" tabular view

CICS system	Definition name	Definition status	Activation period	Resource name pattern	Monitoring resource class	Monitoring inclusion status	Resource status facility monitoring status
EYUMAS1A	*	ACTIVE		*	MCONN	YES	NO
EYUMAS1A	*0000004	ACTIVE		*	MFILE	YES	NO
EYUMAS1A	*0000008	ACTIVE		CEMT	MTRAN	YES	NO
EYUMAS1A	*0000010	ACTIVE		*	MPROG	YES	NO
EYUMAS1A	HTTRAN	ACTIVE		*	MTRAN	YES	NO
EYUMAS1A	ZDZMON2	ACTIVE		S123*	MTERM	NO	NO
EYUMAS1A	ZDZTERM	ACTIVE		S*	MTERM	YES	NO

Look for the following things:

- Verify that the monitor definition has an active status. It is possible that a period definition is causing the monitor definition to be in a pending status.
- Click the name of the monitoring definition to display the "Active monitor specifications" detailed view. Check that the "Monitoring inclusions status" field is set to Yes.
- Monitoring data is not accessible through the "Active monitor specifications" views until one Sample Interval has completed. Therefore, depending on when a monitor definition was installed in relation to the sample interval cycle, you might have to wait through two sample intervals before monitoring data is accessible through the "Active monitor specifications" views. Check the "MASs know to CICSplex" view to see what the sample interval is for each resource type.
  - From the main menu, click **CICSPlex SM operations views** > **MASs known to CICSplex**. The "MASs known to CICSplex" tabular view shows you the monitoring status of each MAS.
  - In the "MASs known to CICSplex" tabular view, the Monitoring status field for the MAS should be YES.
  - Click **YES** to display the "MASs known to CICSplex" detail2 view. The view shows the sample intervals for each resource type.
- Verify that the monitor definition controlling the resource in question is in the list, by checking the "Active monitor specifications" view. If it is not, check the "MASs know to CICSplex" view to confirm that monitoring is active and that there is a nonzero sample interval for that particular resource type, as described earlier.

Unexpected real-time analysis results

Here are two sample problems to discuss ways to approach unexpected real-time analysis results.

One problem deals with system availability monitoring (SAM), the other with MAS resource monitoring (MRM).

An example SAM problem

A CICS system is known to be running and short on storage, yet the condition does not show up in the **RTA outstanding events** view.

From the main menu, click **Real time analysis (RTA) outstanding events** to display this view.

- Check data on the **MASs known to CICSplex** view (MAS object):
  1. From the main menu, click **CICSPlex SM operations views** > **MASs known to CICSplex**.
  2. Verify that the **MASs known to CICSplex** view shows an active status for that CICS system.

     If this view does not show an active status, see "Incomplete operations data returned" on page 94.
  3. Verify that the **Real Time Analysis Status** field on the **MASs known to CICSplex** view indicates YES. This is required for CICSPlex SM to perform system availability monitoring for any of the predefined conditions (SOS, SYSDUMP, TRANDUMP, MAXTASK, STALL). To make a real-time analysis active immediately:
     a. Select a record by clicking on the check box.

b. Click a CICS system name to display the **MASs known to CICSplex** detailed view.
c. Set the **Real Time Analysis Status** field to YES.
d. Click **Apply changes**.

To make the change permanent, you must update the CICS system (CSYSDEF) definition. For more information about CICS system definitions, see Administering CICSPlex SM.

- Use the following steps to determine which action definition controls what happens for the short-on-storage (SOS) condition.
  1. From the main menu, click **CICSPlex SM operations views** > **CICS system definitions**.
  2. Click the check box by relevant CICS system record and click the **CICS system name** field. The **CICS system definitions** detailed view is displayed.
  3. Scroll down to the **Action on Short on Storage (SOS) Event** field and note the name of the action definition.

  The default action is to issue a CICSPlex SM event and to send condition entry and condition exit WTO messages.
- To see which type of external notification is supposed to be issued for this action, you need to look at the action definition:
  1. From the main menu, click **Administration views** > **RTA system availability monitoring**.
  2. Click **Actions** to display the **Action definitions** tabular view.
  3. Click the action name to display the **Action definitions** detailed view, which shows the actions taken and messages generated when a short-on-storage condition is raised.
  4. Check that the **Generate action** field contains NO.

An example MRM problem

MAS resource monitoring (MRM) can be used to generate an event whenever any of a specific group of transactions is disabled in a specific MAS.

The "Local or dynamic transactions" view (LOCTRAN object), with scope set to that MAS, shows that one of the transactions is disabled, yet no event shows up in the "RTA outstanding events" view (EVENT object).

1. Verify that the real-time analysis definition is active.
   - From the main menu, set the scope to the MAS in question.
   - Click **Real time analysis (RTA) views** > **Real time analysis (RTA) installed analysis and status definitions**.
   - On the "Real time analysis (RTA) installed analysis and status definitions" tabular view (RTAACTV object), check the status of the analysis definition.

Definition name	CICS system name	Definition status	Period definition name	Interval between evaluations (seconds)	Associated action name	Analysis definitions type (analysis or status)
DSAGETMN	EYUMAS1A	PENDING	TVSHIFT2	60	DSAGMACT	RTADEF
TRANDIS	EYUMAS1A	ACTIVE		60	DSALMACT	RTADEF
LFILEDEL	EYUMAS1A	PENDING	TVSHIFT2	300	LFILDACT	RTADEF
LFILEOPN	EYUMAS1A	ACTIVE		300	LFILOACT	RTADEF
PGMUSE	EYUMAS1A	ACTIVE		60	PGMUSACT	RTADEF

Definition name	CICS system name	Definition status	Period definition name	Interval between evaluations (seconds)	Associated action name	Analysis definitions type (analysis or status)
PGM1	EYUMAS1A	PENDING	TVSHIFT2	60	PGMUSACT	RTADEF

If the analysis definition is not in the list, or is in the list with a PENDING status, that explains why nothing shows up in the "RTA outstanding events" view. The PENDING status indicates that the analysis definition is not within the Period shown. Absence from this active list indicates the analysis definition was either discarded (by clicking **Discard** on the "Real time analysis (RTA) installed analysis and status definitions" tabular view) or never installed.

2. Examine the analysis definition and related evaluation definitions and action definitions. If the analysis definition is listed in the "Real time analysis (RTA) installed analysis and status definitions" tabular view (RTAACTV object), you should reexamine the analysis definition, the evaluation definitions that make up the analysis definition's evaluation expression, and the associated action definitions.

   Here are some points to consider:

   a. Sample Interval

      The sample interval affects how soon the occurrence of a particular condition (such as a transaction becoming disabled) results in a real-time analysis notification. Also keep in mind that there are two sample intervals: the *evaluation* definition has a sample interval, which determines how often a resource is sampled, and the *analysis* definition has a sample interval, which determines how often an evaluation expression is evaluated.

   b. Entry and Exit Intervals

      An analysis definition's entry and exit intervals have an effect on when a real-time analysis notification follows the occurrence of a certain condition.

   c. Action definitions

      You should ensure that the action definition associated with an analysis definition is set up to deliver the action that you expect. It is possible that a notification results in an SNA generic alert and not in an external message or a CICSPlex SM event.

Unexpected workload management routing decision

You may need to investigate questionable or misunderstood dynamic routing decisions.

For example, you might expect a specific dynamic routing request to be routed to the healthiest target region in a group of target regions. However, you might find that the request is always routed to one particular target region, regardless of the health of the target region.

The approach described here is as follows:

1. Make sure that dynamic routing is enabled for the work requests
2. Determine which workload is active
3. Determine whether the workload is separated by TRANSID, LUNAME or USERID
4. Determine whether there are active affinities

Is dynamic routing enabled?

You should check whether or not dynamic routing is enabled.

- In the transaction definition, the **Dynamic routing option** and **Dynamic routing status** fields should be set to Yes. To check this:
  - From the main menu, click **Administration views** and either **Basic CICS resource administration views** or **Business Applications Services (BAS) administration views**.
  - From the menu, click **Resource definitions > Transaction definitions**.
  - Click the transaction name to display the **Transaction definitions** detailed view. Scroll down to check the settings for the **Dynamic routing option** and **Dynamic routing status** fields.
- In the program definition, the **Dynamic routing status** field should be set to Yes. To check this:
  - From the main menu, click **Administration views** and either **Basic CICS resource administration views** or **Business Applications Services (BAS) administration views**.
  - From the menu, click **Resource definitions > Program definitions**.
  - Click the program name to display the **Program definitions** detailed view. Scroll down to check the setting for the **Dynamic routing status** field.
- If you are using BAS, the program should not be defined to the local system.
- The program may not be picking up the correct transaction id. Transaction ids are selected in the following order of precedence:
  - The transaction id specified in the EXEC CICS LINK command takes priority over a transaction id supplied in any other way.
  - The transaction id supplied in EYU9WRAM, the communication area for the dynamic routing user exit EYU9XLOP.
  - The transaction id specified in the program definition, if there is no transaction id specified in either the EXEC CICS LINK command or EYU9WRAM.
  - By default, if all other possibilities are blank, the CICS mirror transaction CSMI.

Which workload is active?

The first step is to determine which workload is active in the region from which the dynamic request is routed.

- From the main menu, click **Administration views > Workload management administration views**.
- From the **Workload management administration views** menu, click either **Specifications to system links** or **Specifications to system group links**.

A routing region can be associated with only one workload specification. In either the **WLM specifications to system links** view or **WLM specifications to system group links**, look in the **CICS system** field for the routing region you are concerned with, and find the name of the associated workload specification. This name is the name of the workload that is activated when the requesting region starts.

One thing to remember about the **WLM specifications to system links** view or **WLM specifications to system group links** view (and all other workload views) is that it reflects information that is in the data repository. It is possible that the data repository has been modified since its definitions were installed into running

systems. Therefore, you must use the active workload views to see which definitions are installed and active in running systems.

To verify that a workload is active:

- From this main menu, click **Active workload views > Active workloads** to display the **Active workloads** tabular view (WLMAWORK object).
- Click the workload name to display the **Active workloads** detailed view and check that the **Workload Status** field for the workload is set to ACTIVE.

Now you need to ensure that the workload is actively associated with the routing region you are interested in. From the **Active workloads** tabular view, click the **Active routing regions** field to display the **Active workload routing regions** view (WLMAWTOR object). The **Active workload routing regions** view shows which routing regions are actively running a given workload.

Is the workload being separated?

After you know which workload is active on the routing region, the next step is to find out if the workload is being separated based on TRANSID, USERID, LUNAME, or a combination of these.

To do that, take the request in question (the one defined as dynamic, initiated by terminal input) and see whether it is a member of any active transaction groups:

- From the main menu, click **Active workload views** > **Dynamic transactions**.
- The "Active workload dynamic transactions" tabular view (WLMATRAN object) is displayed.

If the transaction in question is listed in this view, the routing decision is possibly based on a workload definition associated with the transaction group of which the transaction is a member. Note the name of the transaction group.

Now look at the active workload definitions:

- From the main menu, click **Active workload views** > **Definitions**.
- The "Active workload definitions" tabular view (WLMAWDEF object) is displayed.

This view shows you which workload definition, if any, applies to the routing request in question. You know the USERID and LUNAME from which the routing request came. You also know whether the transaction is a member of an active transaction group, and, if it is, you know the name of the transaction group. Given these three things, you can tell which workload definition, if any, controls the routing decision. The following pseudocode explains the logic:

```
IF dynamic transaction in question is a member of an active transaction group

 THEN  IF there is a workload definition associated with that transaction group
        THEN  IF the USERID and NAME match the pattern on that workload definition
               THEN  that workload definition will control the routing decision
               ELSE  the workload default controls the routing decision
        ELSE  the workload default controls the routing decision

 ELSE  IF there is a workload definition not associated with a transaction group
        THEN  IF the USERID and NAME match the pattern on that workload definition
               THEN  that workload definition will control the routing decision
               ELSE  the workload default controls the routing decision
        ELSE  the workload default controls the routing decision
```

To illustrate this logic, here are some examples using the data on the "Active workload definitions" tabular view, which is shown in Table 7.

Table 7. Tabular representation of an "Active workload definitions" view (WLMAWDEF object)

Name	System ID of workload owner	Workload definition	Transaction group	Terminal LU name	User ID	Process type	Scope name of set of target regions
T123DEF	EYUWLS02	HTC1		.++++T123	*		EYUMAS1B
WMDFAFFA	EYUWLS02		HTC1	WMTAFFA	.*		EYUMAS1B
WMDFAAB	EYUWLS02	HTC1	WMTAFFB	.*	DEPT02*		EYUMAS2B
WDMFAFFC	EYUWLS02	HTC1	WMTAFFC	.*	*		EYUCSG02

Example 1
The transaction is a member of active transaction group WMTAFFA. The USERID is DEPT01DZ. The LUNAME is NET1.IYJFT123. The routing decision is controlled by workload definition WMDFAFFA.

Example 2
The transaction is not a member of an active transaction group. The USERID is DEPT01DZ. The LUNAME is NET1.IYJFT123. The routing decision is controlled by workload definition T123DEF.

Example 3
The transaction is a member of active transaction group WMTAFFB. The USERID is DEPT01DZ. The LUNAME is NET1.IYJFT123. The routing decision is controlled by the workload default.

When you know which workload definition is controlling the routing decision, the Target Scope field on that same "Active workload definitions" view (WLMAWDEF object) shows you the target region or target region group to which the transaction is routed. If the workload default is controlling the routing decision, the Default target scope field on the "Active workloads" view (WLMAWORK object) shows where the transaction is routed.

Are there any active affinities?

Given that a transaction is routed to a specific target region group, an active affinity forces the transaction to go to a particular target region in that group.

Affinities are associated with a transaction group. To see whether there are any active affinities for a transaction group, display the **Active workload transaction groups** view (WLMATGRP object) to show all active transaction groups and click the **Default affinity type** field. If there is no active transaction group involved, the default transaction group comes into play. To see whether there is an affinity associated with the default transaction group, click the **Default affinity type** field of the **Active workloads** view (WLMAWORK object).

Application programming interface problems

For problems with a program written using the CICSPlex SM application programming interface (API), first rule out potential sources of problems. Then check for error messages and abends, and collect the relevant information before contacting your IBM Support Center.

The first step is to rule out the following potential sources of problems:

- Coding errors in the program itself.

- Incompatibilities between the program and the environment where it runs.
  - If the API program is a REXX exec, ensure that the API function package (module EYU9AR00 with the aliases of EYU9AR01 and IRXFUSER) is in an authorized library that is in the MVS linklist or allocated to the STEPLIB DD in the address space in which the REXX exec is running.
  - If the API program is an assembled or compiled program, verify that the program assembled or compiled correctly and that it was link-edited with the appropriate API stub for the environment in which the program runs. The API stub for a CICS environment is EYU9AMSI. The API stub for a non-CICS environment is EYU9ABSI.

If, after you rule out these factors, the program still does not run successfully, use the following procedure:

1. Check for error messages and abends.

   Such messages could be issued by:
   - The CMAS to which the API processing thread is connected.
   - The MAS or user address space where the program is running.

   If the program is running under MVS as a batch or NetView® program, error messages are written to the MVS console. If the program is running under CICS, error messages are written to the CICS message log.
2. Collect the following documentation:
   - Program source
   - Program listing (for compiled or assembled programs)
   - Linkage editor map (for compiled or assembled programs)

   Also, collect as much of the following as possible:
   - AUXTRACE data set for the CMAS, if available
   - Formatted EYU_TRACE output (for REXX programs).
   - System console log
   - Appropriate job logs
   - System or transaction dump, if any

   When you have all the relevant information, contact your IBM Support Center.

Investigating Web User Interface problems

This section describes how to diagnose and solve problems with the Web User Interface.

Server and web browser messages

During the operation of the Web User Interface, messages are written to the console, web browser and EYULOG.

Server messages

The Web User Interface server messages are mainly written to the CICSPlex SM EYULOG of the Web User Interface server (and not of the CMAS).

Some messages are also written to the console. The Web User Interface server messages are explained in the Web User Interface Message Help.

Web browser messages

Three types of Web User Interface messages are written to the web browser.

- Client

These messages reflect status during the operation of the Web User Interface.

- Editor

  These messages reflect status during the operation of the View Editor.

- HTTP

  These messages reflect the HTTP response codes. Web User Interface HTTP messages can sometimes be hidden by substitute HTTP messages that are issued by the web browser.

You can obtain help for the client and editor messages by clicking the message number or by accessing the contents page of the Web User Interface Message Help. You can obtain help for HTTP messages by accessing the contents page of the Web User Interface Message Help.

COVC status panel

The COVC status panel returns status information about the Web User Interface server.

```
COVC               CICSPlex SM Web User Interface Control            EYUVCTS

                              Status Details

         CMAS Sysid            : QSTX
         Server Sysid          : QSGW
         CICSPlex SM Release   : 0210

         Secure Sockets        : No
         Port                  : 05126
         Hostname              : mvsxx.company.com

         TCP/IP Service Name   : EYUWUI
         TCP/IP Service Status : Open              TCP/IP Family : IPV4
         TCP/IP Address        : 127.10.10.12

Current Status : Ready                                    Time : 20:40:36
Applid         : IYCQSTGW                                 Date : 02/27/2001




PF   1 Help      3 Exit                                      12 Return
```

Figure 38. COVC status panel

COVC debugging commands

The COVC transaction provides access to the Web User Interface run-time environment.

It can be used to format and manipulate the internal data structures of the Web User Interface.

Running the COVC transaction for debugging

To run the COVC transaction, log onto the Web User Interface server and enter the COVC transaction ID, together with one of the following debugging commands,

- STARt

  This starts the Web User Interface server if not already started during PLTPI processing.

- STOp

This shuts down the Web User Interface server.

- TRace

  This displays the COVC trace flags panel, as shown in Figure 5 on page 36, giving you the opportunity to set appropriate trace levels.

- Dump

  This displays the Web User Interface control blocks using the CODB memory display. You can display a control block directly by specifying the control block on the COVC DUMP command, for example 'COVC DUMP ANCHOR' displays the global anchor block. The control blocks that you can display are listed in Table 8.

Table 8. Dump control blocks

Dump control blocks	Meaning
Anchor	Global anchor block
Gslrt	Global task block
Res	NLS resource block
Mos	Managed object block
Mos VOMO `object`	Managed object block for object `object`
Mos VOMA`object attribute`	Managed object block for attribute `attribute` of object `object`
Mos VOMX `object action`	Managed object block for action `action` of object `object`
Mos VOMP `object action parameter`	Managed object block for parameter `parameter` for action `action` of object `object`
View	View cache block
Cwi	Web interface block
Tasks	User tasks block
EYU0Vccc	Entry point and module header for named method within EYU9VKEC load module.

Attention: The CICSPlex SM COVC DUMP keyword should be used only at the request of your IBM support center. You must take steps to ensure that this transaction is used only by authorized personnel because of the access to system control areas that it provides. Improper or unauthorized use of COVC DUMP may have serious consequences, including without limitation loss of data or system outage. Customers are solely responsible for such misuse.

Note: In both the COVC commands and the COVC DUMP commands listed in Table 8, the characters written in uppercase are the minimum number of characters you need to type to issue the command.

Typical end-user problems

Here are some typical end-user problems that you might encounter with possible solutions.

Table 9. Typical end-user problems

Problems	Possible solutions
User unable to sign on, even when specifying reconnect	Check HTTP cookie support is enabled in the user's web browser.

Table 9. Typical end-user problems (continued)

Problems	Possible solutions
Apparently random characters displayed on web browser	Ensure correct code page translation table (DFHCNV) is in use.
Can't see graphical attribute presentations	Check that you are using a Java-enabled web browser and that it is enabled.
Message EYUVC1200E displayed	Ensure that the user's web browser has HTML frame support and that it is enabled.
Message EYUVH0400E displayed (or HTTP 400 message)	Ensure that the HTTP request is set up correctly: • Check that your web browser is using HTTP 1.0 or 1.1 • Check your web browser service level is appropriate
Message EYUVH0503E displayed (or HTTP 503 message) or no response from server	Check that your Web User Interface server is active and the CICS Web Interface operational.
Message EYUVH404E displayed (or HTTP 404 message)	Check that the URL being used to access the server was entered correctly and valid. If accessing a help page ensure that the help page exists.
Missing data fields in views	Some attribute fields are derived from CICS CMF performance class monitoring data. In order for these fields to function correctly, you need to ensure that the CICS monitoring facility is active by setting the CICS system initialization parameters **MNPER** and **MNRES** to YES.

For further information about the messages, access the contents page of the Web User Interface Message Help. For the client message, you can also obtain help by clicking on the message.

Chapter 9. CICSPlex SM naming standards

CICSPlex SM has several naming standards.

The format of names

The names of modules, macros, and other source members distributed with CICSPlex SM have a particular format.

The names of modules, macros, and other source members distributed with CICSPlex SM take the form:

```
prdtccxx
```

where:

prd Is the module prefix see "Module prefixes" on page 110.

t Identifies the type of element, as listed in "Element type identifiers."

cc Is a component identifier, as listed in "EYU components" on page 110.

xx Is a unique identifier assigned by each component.

For example, EYU0MMIN is an executable module for the Monitor Services component.

Element type identifiers

Here is a list of element type identifiers.

ID	Description
$	Selection menus
0	Executable modules (C or assembler)
6	Dynamically acquired control blocks or data areas
7	Module entry point descriptors
8	Function/service definition tables and assembled control blocks
9	Load modules
B or R	Assembler mapping DSECTs
C	C code generation macros
D	ISPF display or data entry panels
E	CLISTs
F	Function variables
G	ISPF message definitions
H	ISPF help panels
J	Screen definitions
M	C structure TYPEDEFs
P	Profile variables or USERFILE members

© Copyright IBM Corp. 1997, 2017

Q Assembler code generation macros

T View, message, and action tables

U Assembler equate files

V C equate files

W or X
Assembled help modules

Z View definitions

Module prefixes

CICSPlex SM modules begin with the prefixes CJA, CJB, CJC, CJD, CJE, CJF, CJG, and EYU. The prefix relates to the CICS release specific agent code of the underlying module.

The prefixes and their associated CICS releases are as follows:

Prefix	CICS release identifier
CJA	CICS 0640
CJB	CICS 0650
CJC	CICS 0660
CJD	CICS 0670
CJE	CICS 0680
CJF	CICS 0690
CJG	CICS 0700
CJH	CICS 0710
EYU	All supported CICS releases

EYU components

The EYU components are as follows.

ID	Description
B*x*	Business Application Services
C*x*	Communications
E*x*	End-user Interface
M*x*	Monitor Services
N*x*	Managed Application System
P*x*	real-time analysis
T*x*	Topology Services
W*x*	Workload Manager
XC	Data Cache Manager
XD	Data Repository
XE	Environment Services System Services
XL	Kernel Linkage
XM	Message Services
XQ	Queue Manager
XS	Common Services

XZ	Trace Services

Chapter 10. Major components of CICSPlex SM

The major components of CICSPlex SM and their 3-character identifiers are as follows.

Component Name	Identifier
Business Application Services	BAS
Common Services	SRV
Communications	COM
Data Cache Manager	CHE
Data Repository	DAT
Environment Services System Services	ESS
Kernel Linkage	KNL
Managed Application System	MAS
Message Services	MSG
Monitor Services	MON
Queue Manager	QUE
real-time analysis	RTA
Topology Services	TOP
Trace Services	TRC
Workload Manager	WLM

© Copyright IBM Corp. 1997, 2017

Chapter 11. System parameters for problem determination

CICSPlex SM system parameters are used to identify or alter the attributes of a CMAS or MAS.

Some system parameters are required in a CMAS or MAS startup job. However, the system parameters described here are optional and are used primarily for problem determination. In the course of diagnosing a problem, IBM customer support personnel might ask you to start a CMAS or MAS with one or more of these parameters specified.

Specifying system parameters

System parameters are specified by means of an extrapartition transient data queue with a destination ID of COPR.

The parameters may be assigned to a DD * file, sequential data set, or a partitioned data set member. The DD name for the extrapartition transient data queue is EYUPARM.

The parameters are coded as 80-byte records. Multiple system parameters can be specified on a single record as long as they are separated by commas and do not exceed 71 characters in length. The format of a system parameter is:

```
keyword(value)
```

where:

keyword
Is the name of a CICSPlex SM system parameter.

There is a problem determination parameter for each CMAS or MAS component. The parameter is named as follows, where xxx is the 3-character component identifier:

xxxTRACE
Turns one or more levels of tracing on for the component. By default, component tracing is not active when a CMAS or MAS starts.

value
Is the alphanumeric data value assigned to the parameter.

For the trace and message parameters shown here, you can specify one or more values between 1 and 32. Values of 1 and 2 provide standard trace entries and messages; values of 3 through 32 cause special trace entries and messages to be recorded.

You can specify multiple values on a single parameter. To specify individual values, separate the values with a comma. To specify a range of values, separate the low and high values with a colon. For example:

```
KNLTRACE(1:3,16,28:32)
```

turns on trace levels 1 through 3, 16, and 28 through 32 in the Kernel Linkage (KNL) component.

© Copyright IBM Corp. 1997, 2017

To request multiple values for the same parameter, you must specify them as a single entry. If the same parameter is specified more than once, only the last entry is used.

Note: Once a CMAS or MAS has been started, you can use the WUI to control the trace settings in a component by using the following views:

- **CMAS detail** (EYUSTARTCMAS.TRACE) view to change CMAS component trace settings
- **MASs known to CICSplex** (EYUSTARTMAS.TRACE) view to change MAS component trace settings

The problem determination parameters

You might be asked for certain CICSPlex SM system parameters to help with problem determination.

As indicated in Table 10, some of the parameters can be used in the startup job for both CMASs and MASs; other parameters are specific to either a CMAS or a MAS.

Table 10. System parameters for problem determination

Name	Description	Values	Default 1	Used by
BASTRACE	Business Application Services trace settings	1–32	None	Both
CHETRACE	Data Cache Manager trace settings	1–32	None	Both
CICSDUMPS	CICS system dumping active	NO | YES	NO	CMAS
COMTRACE	Communications trace settings	1–32	None	Both
DATTRACE	Data Repository trace settings	1–32	None	Both
ESDUMP	Take SDUMP on all CMAS and MAS failures	YES | NO | NEVER	NO	Both[1]
ESDUMPCOM	Suppress communication task dumps	YES | NO	NO	CMAS
ESDUMPLIMIT	Controls the number of dumps collected for a given failure.	0–999	1	CMAS[2]
KNLTRACE	Kernel Linkage trace settings	1–32	None	Both
MASTRACE	Managed Application System trace settings	1–32	None	MAS
MONTRACE	Monitor Services trace settings	1–32	None	CMAS
MSGTRACE	Message Services trace settings	1–32	None	Both
QUETRACE	Queue Manager trace settings	1–32	None	Both
RTATRACE	real-time analysis trace settings	1–32	None	Both[3]
SRVTRACE	Common Services trace settings	1–32	None	Both
TOPTRACE	Topology Services trace settings	1–32	None	Both
TRCTRACE	Trace Services trace settings	1–32	None	Both
WLMTRACE	Workload Manager trace settings	1–32	None	Both[4]

Notes:

1 You should only modify the default values at the request of your IBM Service Representative.

1. ESDUMP - SVC dumps are issued by CICSPlex SM through EYU9XLRV during abend recovery or on demand through EYU0XZPT and EYU0XZSD.
   - When ESDUMP(YES) is specified, no SVC dumps are suppressed.
   - When ESDUMP(NO) is specified, duplicate dumps taken by EYU9XLRV, EYU0XZPT and internally by EYU0XZSD may be suppressed, depending upon the value of the ESDUMPLIMIT EYUPARM. SVC dump requests made through EYU0XZSD from the COD0 transaction continue to be honored.

- When ESDUMP(NEVER) is specified, all SVC dumps taken by EYU9XLRV, EYU0XZPT, and internally by EYU0XZSD are suppressed. SVC dump requests made through EYU0XZSD from the COD0 transaction continue to be honored.

It is strongly recommended that ESDUMP(NEVER) not be used, because it might affect the ability to debug problems. If a problem occurs with SVC dump suppression active, you might be required to reproduce the problem without SVC dump suppression active before debugging can be performed for the problem.

To deactivate SVC dump suppression without restarting the CMAS or MAS, the ESDUMP value can also be updated using one of the following methods:

- Using the COD0 SET command.
- For a CMAS only, by updating the CMAS resource table SDUMP field using the API or WUI

2. If the ESDUMP system parameter is set to NO, the SVC dump suppression is controlled by the ESDUMPLIMIT CICSPlex SM system parameter. You can use it in the startup job for the CMAS. This parameter controls the number of dumps collected for a given failure. The default is 1.

   For example, if you want to capture dump diagnostic information for the first 3 instances of an error, add the following to the CMAS's EYUPARM DD statement:

   ```
   ESDUMP(NO)
   ESDUMPLIMIT(3)
   ```

   An abend control block of entries is created whenever a CICSPlex SM dump is requested. Each entry contains information about the number of different abends that have occurred. Separate entries are maintained for CMASes and MASes.

   The instances of an abend are determined by a combination of abending program name, abend codes, abend offset and region type (CMAS or MAS).

   For dumps requested by a MAS, the ESDUMPLIMIT of the CMAS that the MAS connects to is used. For a MAS, the dump limit is shared among all the MASes that connect to the CMAS. For example, if ESDUMPLIMIT(5) is set in a CMAS, and 10 different MASes all receive the same abend, dumps are requested only by the first 5 MASes.

   The number of abends that have occurred is reset to 0 for MAS-related abend entries when the last MAS disconnects from a CMAS.

   The number of abends that have occurred is reset to 0 for CMAS related abend entries during CMAS initialization.

   All abend entries are reset for both MASes and CMASes if a CMAS and all the MASes that connect to it are shut down at the same time.
3. RTATRACE is valid in a MAS only if status definitions are installed and being used by a user-written status program.
4. WLMTRACE is valid in a MAS only if it is a local MAS acting as a TOR in a CICSPlex SM workload.

Notices

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

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

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

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

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

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

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

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

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

© Copyright IBM Corp. 1997, 2017

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

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

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

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

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

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

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

COPYRIGHT LICENSE:

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

Programming interface information

CICS supplies some documentation that can be considered to be Programming Interfaces, and some documentation that cannot be considered to be a Programming Interface.

Programming Interfaces that allow the customer to write programs to obtain the services of CICS Transaction Server for z/OS, Version 5 Release 4 are included in the following sections of the online product documentation:

- Developing applications
- Developing system programs
- RACF security overview
- Developing for external interfaces
- Reference: application developmenth
- Reference: system programming
- Reference: connectivity

Information that is NOT intended to be used as a Programming Interface of CICS Transaction Server for z/OS, Version 5 Release 4 , but that might be misconstrued as Programming Interfaces, is included in the following sections of the online product documentation:

- Troubleshooting and support
- Reference: diagnostics

If you access the CICS documentation in manuals in PDF format, Programming Interfaces that allow the customer to write programs to obtain the services of CICS Transaction Server for z/OS, Version 5 Release 4 are included in the following manuals:

- Application Programming Guide and Application Programming Reference
- Business Transaction Services
- Customization Guide
- C++ OO Class Libraries
- Debugging Tools Interfaces Reference
- Distributed Transaction Programming Guide
- External Interfaces Guide
- Front End Programming Interface Guide
- IMS Database Control Guide
- Installation Guide
- Security Guide
- Supplied Transactions
- CICSPlex SM Managing Workloads
- CICSPlex SM Managing Resource Usage
- CICSPlex SM Application Programming Guide and Application Programming Reference
- Java Applications in CICS

If you access the CICS documentation in manuals in PDF format, information that is NOT intended to be used as a Programming Interface of CICS Transaction Server for z/OS, Version 5 Release 4 , but that might be misconstrued as Programming Interfaces, is included in the following manuals:

- Data Areas
- Diagnosis Reference
- Problem Determination Guide
- CICSPlex SM Problem Determination Guide

Trademarks

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

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

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

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

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

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

Terms and conditions for product documentation

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

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

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

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

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

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

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

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

IBM online privacy statement

IBM Software products, including software as a service solutions, (“Software Offerings”) may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering’s use of cookies is set forth below:

For the CICSPlex SM Web User Interface (main interface):
Depending upon the configurations deployed, this Software Offering may use session and persistent cookies that collect each user’s user name and other personally identifiable information for purposes of session management, authentication, enhanced user usability, or other usage tracking or functional purposes. These cookies cannot be disabled.

For the CICSPlex SM Web User Interface (data interface):
Depending upon the configurations deployed, this Software Offering may use session cookies that collect each user's user name and other personally identifiable information for purposes of session management, authentication, or other usage tracking or functional purposes. These cookies cannot be disabled.

For the CICSPlex SM Web User Interface ("hello world" page):
Depending upon the configurations deployed, this Software Offering may use session cookies that collect no personally identifiable information. These cookies cannot be disabled.

For CICS Explorer®:
Depending upon the configurations deployed, this Software Offering may use session and persistent preferences that collect each user’s user name and password, for purposes of session management, authentication, and single sign-on configuration. These preferences cannot be disabled, although storing a user's password on disk in encrypted form can only be enabled by the user's explicit action to check a check box during sign-on.

If the configurations deployed for this Software Offering provide you, as customer, the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM Privacy Policy and IBM Online Privacy Statement, the section entitled “Cookies, Web Beacons and Other Technologies” and the IBM Software Products and Software-as-a-Service Privacy Statement.

Index

Special characters

-requested dumps
  during CMAS initialization 39
  during ESSS PC routine execution 40
  during MAS initialization 39

A

abend, investigating 91
activating component tracing
  in one or more CMASs 23
  in one or more MASs 25
affinities 103
agent code, MAS 9, 10
ALLOC debugging command 56
API program problems 103
application programming interface problems 103
ATCB parameter 66
ATTACH debugging command 57
AUXTR 35
AUXTRACE facilities
  in a CMAS 27
  in a MAS 27

B

bottleneck, investigating 93

C

CALL debugging command 59
CAPTURE debugging command 60, 61
change log contents 19
CICS AUXTRACE facilities
  in a CMAS 27
  in a MAS 27
CICS system initialization parameters
  AUXTR 35
  SYSTR 35
  USERTR 35
CMAS
  description 5
  maintenance point 7
  networks and registration 7
  structure 8, 9
CMAS component tracing 23
CMAS trace 29
COD0 debugging commands
  ALLOC 56
  ATTACH 57
  CALL 59
  CAPTURE 60, 61
  DUMP 62
  EXEC 64
  EXIT 64
  HELP 64
  LIST 65
  POST 75
COD0 debugging commands *(continued)*
  PRINT 76
  PURGE 76
  START 77
  TRACE 78
  TRACK 78
  TRAP 79
COD0 transaction 55
CODB debugging commands 84, 85
CODB transaction 83
COLU transaction 49, 50
common components of
  Common Services 12
  Communications 13
  Data Cache Manager 12
  Data Repository 12
  Kernel Linkage 11
  Message Services 11
  Queue Manager 12
  Trace Services 11
common problems 16
Common Services component 12
Communications component 13
component identifiers
  three-character 113
component identifiers module prefixes
  in element names 110
controlling the amount of trace
  using system parameters 115
  using the WUI 29, 30
COVC
  debugging commands 105
    DUMP 105
    START 105
    STOP 105
    TRACE 105
  dump 42
  status details 105
  trace flags 35

D

Data Cache Manager component 12
Data Repository component 12
data spaces
  description 10
  dumping 42
debugging transactions
  method-level (COD0) 55
  running 54
  system-level (CODB) 83
diagnostic documentation
  for a bottleneck 93
  for a stall 92
  for an abend 91
  for problem determination
    related products 19
    site documentation 19
dump codes
  EYU0VWAN 43
  EYU0VWCV 43
dump codes *(continued)*
  WUIABEND 43
  WUITRACE 43
DUMP debugging command 62
dump facilities
  IPCS tools
    dump formatting routine 43, 46
  types of dumps 37
dump formatting routine 43, 46
dump types
  -requested
    during CMAS initialization 39
    during ESSS PC routine execution 40
    during MAS initialization 39
  unexpected dumps
    under CICS 38
  user-requested
    using the MVS DUMP command 42
dumps
  Web User Interface 42

E

element types, 109
Environment Services System Services (ESSS)
  description 5, 10
  utility 46, 48
ESSS 48
ESSS (Environment Services System Services)
  description 5, 10
  utility 46, 48
ESSS Information Display Utility 48
ESSS utility (EYU9XEUT)
  JCL 48
  options 46, 47
exception trace 37
exception tracing 29
EXEC debugging command 64
EXIT debugging command 64
EYU0XZPT system dump code 39
EYU0XZSD system dump code 39
EYU9D 43
EYU9D420 43
EYU9XENF 48
EYU9XEUT utility
  JCL 48
  options 46, 47
EYU9XZUT 35
EYU9XZUT utility
  JCL 33
  options 31, 33
  sample output 34

© Copyright IBM Corp. 1997, 2017

F

format of names 109
formatting dumps with IPCS 43, 46
formatting options, trace 31, 33
formatting trace entries 31, 35

H

help
  Web User Interface message help 105
HELP debugging command 64

I

interpreting trace entries
  in a CMAS or MAS 31
investigating specific problems
  abends 91
  bottlenecks 93
  incomplete operations data 94, 97
  missing monitor data 97
  stalls 92
  unexpected RTA results 98, 99
  unexpected WLM routing 100, 103
  with the API 103
IPCS dump formatting routine 43, 46
IPCS VERBEXIT command 44

K

Kernel Linkage component 11

L

LIST ATCB 66
LIST debugging command
  ALLOC parameter 65
  ATCB parameter 66
  CACHE parameter 66
  CAPTURE parameter 68
  COMM parameter 70
  METH parameter 70
  START parameter 70
  STCB parameter 73
  TASK parameter 73
  VIEWS parameter 75
logical records 21
LOGREC data set 20, 21

M

maintenance point CMAS 7
major object descriptor block (MODB) 9
major object environment block (MOEB) 9
MAL (message argument list) 9
managed application system (MAS)
  agent code 9
  description 5
MAS (managed application system)
  agent code 9
  description 5
MAS agent 9, 10
MAS component tracing 25
MAS trace 29
message argument list (MAL) 9
Message Services component 11
messages 104
  as a source of information 20
  preliminary check for 16
  server 104
  web browser 104
method call environment 9
method-level debugging with COD0
  allocating a resource 56
  attaching a method 57
  calling CICS programs 59
  capturing a table 60
  capturing a view 61
  commands that alter 56
  displaying a MAL 79
  displaying and altering data 62
  entering CODB 62
  executing a method 64
  exiting 64
  function key assignments 55
  listing tasks and resources 65
  main menu 55
  online help 64
  posting an ECB 75
  printing data areas 76
  purging a resource 76
  recursive commands 56
  setting CICS and trace flags 78
  setting trace flags based on call structure 78
  setting trace flags for a method 79
  starting a method in a CMAS 77
missing data fields 107
MODB (major object descriptor block) 9
MOEB (major object environment block) 9
monitor data, missing 97

N

naming convention 115

O

online diagnostic aids
  debugging transactions
    method-level (COD0) 55
    system-level (CODB) 83
  description 19
  online utility (COLU) 49, 54
online utility, COLU 50
output problems 94, 97
overview of
  agents in a MAS 9, 10
  CMAS networks and registration 7
  CMAS structure 8, 9
  common components 10, 13
  ESSS and data spaces 10
  structure of system 5

P

parameters, system
  for problem determination
    list of 116
    types 115
  specifying 115
POST debugging command 75, 76
preliminary checks
  affecting specific parts of system 16
  changes since last run 15
  has system run before 15
  messages 16
  occurring at specific times 16
problem determination
  description 1
  preliminary checks 15, 16
  system parameters
    list of 116
    specifying 115
    types 115
problem types 16
PURGE debugging command 76

Q

Queue Manager component 12

R

RTA results, unexpected
  MAS Resource Monitoring 99
  System Availability Monitoring 98

S

setting CMAS and MAS trace flags 29
sources of information
  change log 19
  LOGREC records 21
  manuals 19
  messages 20
  online diagnostic aids 19
  site documentation 19
  symptom strings 20
  traces 21
special trace levels 29
stall, investigating 92
standard trace levels 28
START debugging command 77
structure of 5
structure of a CMAS 8, 9
symptom strings 20
symptoms of a problem 16
SYS1.LOGREC data set 20, 21
SYSDUMP code entries in a MAS 39
system management problems
  missing monitor data 97
  unexpected RTA results 98, 99
  unexpected WLM routing 100, 103
system parameters
  for problem determination
    list of 116
    types 115
  specifying 115

system-level debugging with CODB
accessing from COD0 89
altering memory 88
commands 84, 85
function key assignments 86
main menu 83
SYSTR 35

T

tools for problem determination
debugging transactions
method-level (COD0) 55
system-level (CODB) 83
dump facilities
IPCS tools 43, 46
types of dumps 37
ESSS utility (EYU9XEUT) 46, 48
online utility (COLU) 49, 54
trace
Web User Interface 35
TRACE debugging command 78
trace facilities
controlling the amount of trace
using system parameters 115
using the WUI 29, 30
description of 21
for a CMAS 27
for a MAS 27
for a WUI 28
formatting trace entries 31, 35
interpreting trace entries
in a CMAS or MAS 31
types and levels of trace 28, 29
exception 29
special 29
standard 28
trace flag syntax 31
trace flags 37
trace format utility (EYU9XZUT)
JCL 33
options 31, 33
sample output 34
trace formatting options 31, 33
Trace Services component 11
tracing
CICS trace table settings 27
exception 29
in a CMAS 27
in a MAS 25, 27
in a WUI 28
special 29
standard 28
trace facilities
use of CICS trace table 27
trace table, CICS 27
TRACK debugging command 78
TRANDUMP code entries in a MAS 39
TRAP debugging command 79
types of dumps
-requested
during CMAS initialization 39
during ESSS PC routine
execution 40
during MAS initialization 39
unexpected dumps
under CICS 38
types of dumps *(continued)*
user-requested
using the MVS DUMP
command 42
types of problems 16

U

unexpected dumps
in a MAS 39
under CICS 38
user-requested dumps
using the MVS DUMP command 42
USERTR 35

V

VERBEXIT command 44

W

Web User Interface 6
dumps 39, 42
trace 35
typical problems 106
Web User Interface message help 105
Web User Interface server initialization
parameters
WUITRACE 35
WLM routing, unexpected 100, 103
WUITRACE 35
WUITRACE parameter 36

