CA MIC Message Sharing for z/OS

CA MIC Programming Guide

Release 12.0

CA Technologies Product References

This document references the following CA products:

- CA 1® Tape Management (CA 1)
- CA 7 Workload Automation (CA 7)
- CA Common Services for z/OS
- CA MIC Message Sharing (CA MIC)
- CA MII Data Sharing (CA MII)
- CA MIM™ Resource Sharing (CA MIM)
- CA OPS/MVS® Event Management and Automation (CA OPS/MVS)
- CA Remote Console™

Contact CA Technologies

Contact CA Support

For your convenience, CA Technologies provides one site where you can access the information that you need for your Home Office, Small Business, and Enterprise CA Technologies products. At http://ca.com/support, you can access the following resources:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Providing Feedback About Product Documentation

If you have comments or questions about CA Technologies product documentation, you can send a message to techpubs@ca.com.

To provide feedback about CA Technologies product documentation, complete our short customer survey which is available on the CA Support website at http://ca.com/docs.

Documentation Changes

The following documentation updates have been made since the last release of this documentation:

Note: In PDF format, page references identify the first page of the topic in which a change was made. The actual change may appear on a later page.

- Updated the Report Generation for CA MIC (see page 89) section.

Contents

Chapter 1: Introduction

This section contains the following topics:

Who Should Use This Book

This book is intended for system programmers and operators responsible for the installation, customization, and day-to-day operation of the CA MIM product. System programmers using this book should have a solid understanding of z/OS operating system internals, subsystems, and components. System operators using this book should understand the hardware controls and general organization of the z/OS operating system.

Overview

CA MIC is the component of the CA MIM product that provides you with cross-system command and message routing services in a multiple-system or multiple-image environment.

CA MIC provides the following:

- Allows you to issue commands from a local command source to one or more external systems and receive the responses back to the local command source.
- Allows you to import messages from external systems and route them to local destinations.
- Allows you to create a statistical report concerning CA MIC activity using a report facility.
- Provides cross-system communication in the following ways:
  - Through a shared DASD control file
  - Through CTC devices
  - Through XCF
  - By activating the ICMF, which interfaces with the communication server component of CA L-Serv. Both GCMF and ICMF must be activated to communicate through CA L-Serv

CA MIC consists of the following facilities:

- Global Command and Message Facility (GCMF)

  The GCMF allows you to route messages and commands to any or all systems in a complex

- Intersystem Communication Facility (ICMF)

  The ICMF provides cross-system transaction routing services for CA MIC. ICMF interfaces with the CA L-Serv product, XMVS, which transports CA MIC data through VTAM cross-domains. This communication option is particularly useful to customers who want to transport messages and commands between geographically remote systems.

  Geographically remote systems typically do not have the shared DASD or CTC connectivity required by the other CA MIM communication options. ICMF can be activated as a stand-alone communication option or used in conjunction with the other communication options.

Notation Used for Components

To aid readability, the CA MIM guides use abbreviated names when referring to the components and facilities, except in cases where the abbreviation would obscure the intended meaning. The abbreviations are as follows:

Abbreviation	Name of Component or Facility
ECMF	Enqueue Conflict Management Facility
EDIF	Enhanced Data Set Integrity Facility
GDIF	Global Data Integrity Facility
GTAF	Global Tape Allocation Facility
TPCF	Tape Preferencing and Control Facility
GCMF	Global Command and Message Facility
ICMF	Intersystem Communication Facility

Chapter 2: Planning Initial Settings

This section contains the following topics:

Overview

The MIMPARMS data set contains parameter values that CA MIM uses as input during product initialization to define the characteristics of the MIMplex. You specify the MIMPARMS data set on the //MIMPARMS DD statement in the JCL procedure used to start CA MIM. A sample MIMPARMS data set is installed with the CA MIM product into data set CAI.CBTDPARM.

You need to set initial values for statements and commands in the following CA MIM MIMPARMS parmlib members:

- MIMINIT
- MIMCMNDS
- MIMSYNCH

Note: For detailed planning information, see the following:

- Advanced Topics (see page 17)
- *Statement and Command Reference Guide*
- *CA MIM Programming Guide*

General Statements

The following general statements can be used in any of the CA MIM parmlib members:

- IFSYS
- ENDIF
- INCLUDE
- LOG
- NOLOG

Note: For more information, see the *CA MIM Statement and Command Reference Guide*.

Facility-specific Members

No facility-specific member is executed for the Global Command and Message Facility (GCMF). However, if you choose to use the Intersystem Communication Facility (ICMF), then you need an IDEFSYS statement to define the ICMF-connected systems. For an IDEFSYS statement example, see the sample MIMINIT member.

Plan the MIMINIT Member

The MIMIINT member of the CA MIM parameter data set contains statements that specify initialization values for CA MIM. The initialization members specific to CA MIC do the following:

- Determine which CA MIC facilities are activated at initialization. Use the following parameters:
  - MIMINIT GCMF
  - MIMINIT ICMF
- Define which communication method to use. Use the following parameter:
  - MIMINIT COMMUNICATION
- Define I/O buffer sizes. Use the following parameters:
  - MIMINIT BLKSIZE
  - MIMINIT VCFBUFFERSIZE
  - MIMINIT VCFMAXBLOCKS

- Define SAF command security options. Use the following parameters:
  - MIMINIT SAFCMDAUTH
  - MIMINIT SAFPREFIX
- Specify which PDS members to use. Use the following parameter:
  - MIMINIT MEMBER
- Initialize values unique to the GCMF, which is a part of the CA MIC component. Use the following parameters:
  - GCMINIT EDITMESSAGE
  - GCMINIT EXTCON
  - GCMINIT JOBID
  - GCMINIT REPLYLIMIT
  - GCMINIT SAFNOTOKEN
  - GCMINIT CONSLIST
  - GCMINIT SUPPRESS
  - GCMINIT SYSNAME
  - GCMINIT SYSTYPE
  - GCMINIT TRANSLATE
- Define the initialization values unique to the Intersystem Communication Facility (ICMF). This facility is a cross-system communication option available only through the CA MIC component. Use the following parameters:
  - ICMINIT AUTOIDEFSYS
  - ICMINIT ICMNAME
  - ICMINIT ISSNAME
  - ICMINIT SYSID
- Identify the systems that use the ICMF communication method. Use the IDEFSYS command.

For more information, see the CA MIM *Statement and Command Reference* Guide.

Plan the MIMCMNDS Member

The MIMCMNDS member of the CA MIM parameter data set contains commands that specify operating values for CA MIM. The commands specific to CA MIC do the following:

- Manage cross-system messages traffic. Use the COLLECT command.
- Create, modify, and delete linkages that enable consoles, products, internal and instream command sources, and TSO users on the local system to issue commands to any system. Use the LINK command.
- Set values for the collection of statistical records. Use the following parameters:
  - SETOPTION GCMF STATCOLLECT
  - SETOPTION GCMF STATCYCLE
  - SETOPTION GCMF STATINTERVAL
- Obtain diagnostic information. See the section Activating Tracing in the chapter “Advanced Topics” in this guide. Use the following parameters:
  - SETOPTION GCMF RESETPRINT
  - SETOPTION GCMF RESETTRACE
  - SETOPTION ICMF RESETPRINT
  - SETOPTION ICMF RESETTRACE
  - SETOPTION GCMF SETPRINT
  - SETOPTION GCMF SETTRACE
  - SETOPTION ICMF SETPRINT
  - SETOPTION ICMF SETTRACE
- Set operating values for the GCMF. Use the following parameters:
  - SETOPTION GCMF ACTIONCODE
  - SETOPTION GCMF ACTIONPREFIX
  - SETOPTION GCMF AUTODELETE
  - SETOPTION GCMF BROADCAST
  - SETOPTION GCMF BUFLIMIT
  - SETOPTION GCMF CAPTURELOG
  - SETOPTION GCMF CART

  - SETOPTION GCMF COLORSUPP
  - SETOPTION GCMF DELETE
  - SETOPTION GCMF DELETEINTERVAL
  - SETOPTION GCMF EXCLUDEJOB
  - SETOPTION GCMF EXCLUDEPREFIX
  - SETOPTION GCMF GCMDUMP
  - SETOPTION GCMF GCMRETRY
  - SETOPTION GCMF MAXCONS
  - SETOPTION GCMF MINCONS
  - SETOPTION GCMF MSGFILTER
  - SETOPTION GCMF SAFSYSTEMS
  - SETOPTION GCMF SYSLOG
- Set operating values for the ICMF. Use the following parameter:
  - SETOPTION ICMF AUTOIDEFSYS

Plan the MIMSYNCH Member

The MIMSYNCH member of the CA MIM parameter data set can contain CA MIM and z/OS commands you want to execute when CA MIM synchronizes with the other systems in the MIMplex.

This member name is pointed to in the MIMPROC member by the PROC SYNCH=MIMSYNCH statement. Commands found in this member execute after MIMINIT statements and MIMCMNDS commands have executed, and only after all CA MIM address spaces in the complex have synchronized.

Any of the general statements listed under General Statements in this chapter or any of the commands referenced in this guide may optionally be placed in the MIMSYNCH member.

Chapter 3: Advanced Topics

This section contains the following topics:

Command Processing

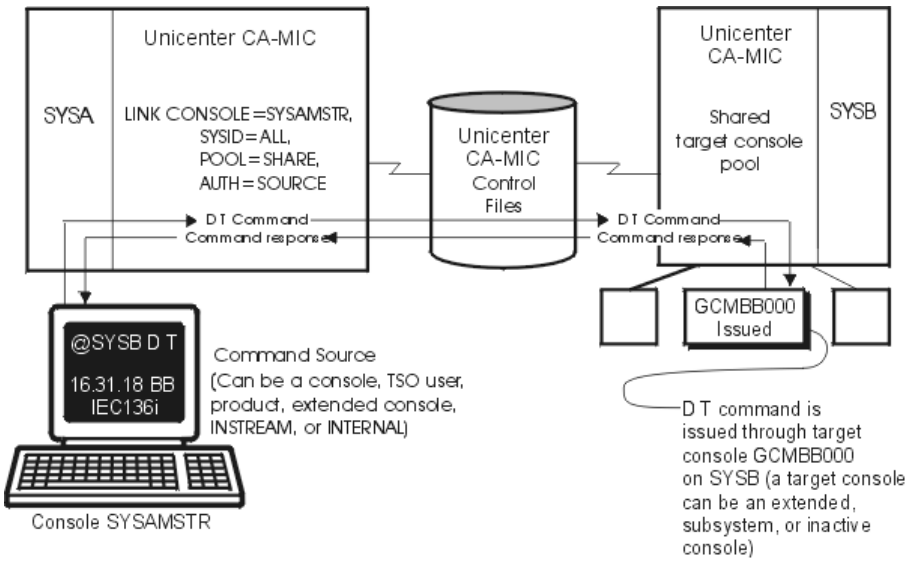
CA MIC allows you to route operator commands from a local command source to one or more external systems, and receive the responses to those commands at the local command source.

How Command Routing Linkages are Defined

You must first define a linkage using the LINK command on the local system to provide a command routing path to the desired external systems. The linkage you create can limit both the external systems to which commands from this source are routed, and the command authority granted to the issuer on the external systems.

Cross-system commands are issued on the external systems and their responses are then sent back to the originating system.

The following illustration shows how commands are sent from a command source to a target console and how the response is returned to the command source:



How Linkages are Created

Command routing paths, or *linkages*, enable you to issue cross-system commands and to receive the cross-system responses to those commands. When you create a linkage, you authorize a command source on the local system to issue commands to a target system. CA MIC uses a console you designate on the target system to execute commands issued through a linkage. The designated console is known as the *target console* for the linkage.

The LINK command lets you create, modify, and delete linkages that enable consoles, products, internal and instream command sources, and TSO users on the local system to issue commands to any system. Each linkage you create by using the LINK ADD command is referenced by the command source you specify. The following are examples of linkage command sources:

```
CONSOLE=TAPECON1
PRODUCT=RCS
TSOUSER=DSIAZ11
```

After a linkage is defined, you can modify or delete the linkage by specifying the same command source named on subsequent LINK commands. You also use the command source to delete a linkage. Except for exclusive linkages, there cannot be two linkages with the same command source.

When you specify the SOURCE operand, the command source for the linkage is the source from which the LINK command is being issued.

Note: For more information about the LINK command, see the *Statement and Command Reference Guide.*

How Command Sources on Linkages are Identified

To identify command sources, specify one of the local command source values on a LINK command.

A command source may qualify for more than one linkage to a particular system. For example, TSO user USERID1 matches both TSOUSER=USERID1 and TSOUSER=ALL. In this case, CA MIC uses the more specific linkage for the command source (TSOUSER=USERID1). If there is no specific linkage for that command source, then CA MIC uses the general linkage (TSOUSER=ALL). You can specify different command authority levels for specific and general linkages.

Note: You can only specify one command source identifier per LINK command.

How Products are Identified as Command Sources

There are two ways to identify products as command sources:

- Specifying a PRODUCT linkage
- Specifying a CONSOLE linkage

Because of integration between CA products, the CA MIC PRODUCT linkage can be used for CA Remote Console and CA OPS/MVS, regardless of the type of console (subsystem or extended) through which these products issue commands.

PRODUCT Linkages

To allow these products (or any product using subsystem consoles) to issue cross-system commands, specify a PRODUCT linkage for the product using the following command syntax:

```
LINK PRODUCT=ssss   SYSID=    AUTHORITY=    POOL=
```

ssss

Specifies the subsystem name of the product. The default subsystem name for CA Remote Console is RCS, and for CA OPS/MVS it is OPSS.

CONSOLE Linkages

To allow products using extended consoles (except CA Remote Console and CA OPS/MVS) to issue cross-system commands, specify a CONSOLE linkage for the product using the following command syntax:

```
LINK CONSOLE=conname    SYSID=    AUTHORITY=    POOL=
```

conname

Specifies the name of the extended console the product is using to issue commands. In many cases, products use more than one extended console to issue commands. The asterisk (*) and pound sign (#) wildcard characters are supported on the CONSOLE keyword so that a single console LINK can be used for more than one console.

For example, if product XYZ allocates extended consoles with names beginning with ABCD, then the following LINK command syntax covers all extended consoles used by product XYZ:

```
LINK CONSOLE=ABCD* SYSID= AUTHORITY= POOL=
```

Cross-system Command Authority Level for a Command Source

Use the AUTHORITY parameter on the LINK command to secure the types of cross-system commands that can be issued through the linkage. All z/OS commands are assigned to an operator command group, or *command authority level*. To determine the authority level of a particular z/OS command, see the IBM documentation on system commands. The AUTHORITY parameter on the LINK command determines the cross-system command authority level for the command source specified on the linkage.

The LINK command can increase or decrease the authority level of a local command source for cross-system commands. For example, a local console defined with SYS authority, with a linkage assigning MASTER authority, can only issue SYS-level commands locally, but can issue MASTER-level commands cross-system.

You can also use the AUTHORITY parameter to decrease the cross-system commands authority of a local console.

Important! You need SYS authority to issue the LINK command. You need only SYS authority to assign MASTER authority through the LINK command.

Note: For more information, see the *CA MIM Statement and Command Reference Guide*.

Command Source Examples

The following examples show how you can create linkages with products as command sources:

- To allow CA Remote Console (RCS) to issue cross-system commands to all external systems (regardless of whether it is using subsystem or extended consoles) with SYS authority, through a shared console pool, issue the following command:

  ```
  LINK PRODUCT=RCS SYSID=EXTERNAL AUTHORITY=SYS POOL=SHARE
  ```

- To allow CA OPS/MVS (OPSS) to issue cross-system commands with MASTER authority to all systems, through a shared console pool, issue the following command:

  ```
  LINK PRODUCT=OPSS SYSID=ALL AUTHORITY=MASTER POOL=SHARE
  ```

How Linkage Types are Defined

The linkage type defines how CA MIC selects a target console on the target system. CA MIC selects a target console on each system to which a cross-system command is issued. You can choose how it selects target consoles for a linkage using one of the following methods:

- Tell CA MIC to select a console from the pool of target consoles available on the target system. The consoles in the pool are shared by all external command sources you identify on LINK commands. To do this, define a shared linkage.
- Tell CA MIC to select a console from the pool of target consoles available on the target system. Once selected, the target console is removed from the pool and dedicated to the external command source. To do this, define a dedicated linkage.
- Specify the console name of a specific target console that you want to assign to a command source. To do this, define an exclusive linkage.

The method you use depends upon factors such as the availability of target consoles, operating system level, and the number of command sources. Use the POOL or TGTCONS parameters on the LINK command to define the type and allocation method to be used with the target console pool. The following sections describe the advantages and disadvantages of each type of linkage and tell you how to create each linkage.

Note: We recommend you use SHARED type linkages whenever possible.

Shared Linkages

A *shared linkage* lets a command source issue commands through a target console that CA MIC selects from the console pool. The advantages of using shared linkages are:

- You do not need to define a particular target console to be assigned to each command source.
- Many command sources from multiple external systems can share a relatively small number of local target consoles.

We recommend this type of linkage for most command sources. Once the target console pool reaches its maximum size, CA MIC continually reassigns the least recently used target console to the current source, and then executes the command. The following is an example of a shared linkage:

```
LINK ALL SYSID=ALL POOL=SHARE
```

Dedicated Linkages

A *dedicated linkage* lets a command source issue commands to a target console that CA MIC selects from the console pool. However, once CA MIC selects a target console, it is removed from the console pool so that it cannot be reassigned to another command source. If many dedicated linkages are defined on external systems, then the local target console pool could be depleted.

The advantages of a dedicated linkage are primarily as follows:

- You do not need to define the particular console to be assigned to each command source.
- Because a target console cannot be assigned to more than one command source, a command response or WTOR message is never misrouted.

Example: Dedicated linkage

```
LINK CONSOLE=TAPECON1 SYSID=ALL POOL=DEDICATE
```

Note: You cannot use dedicated linkages with console masking on the link.

Exclusive Linkages

An *exclusive linkage* assigns a specific target console to a specific command source. When you define an exclusive linkage, you specify the name of the target console that you are assigning to the command source. You can use an active MCS, inactive MCS, or a subsystem-allocatable console as the target console for an exclusive linkage. If you choose an active console as the target console, then all commands and responses issued through this linkage are displayed on the target console. Because the echoing of commands and responses can cause confusion, we recommend caution when selecting an active MCS console as a target console.

You would choose an exclusive linkage primarily for security purposes. This type of linkage establishes a one-to-one relationship between a command source on the local system and the target console on one *and only one* external system. It ensures that a command source always issues commands through the specified target console.

Notes:

- You cannot use exclusive linkages with console masking on the link.
- You cannot define an exclusive linkage if you specify ALL, EXT, or multiple system IDs on the SYSID parameter of the LINK command. You cannot issue cross-system commands with a scope value of ALL or EXT through an exclusive linkage.

The following is an example of an exclusive linkage:

```
LINK CONSOLE=TAPECON1 SYSID=SYSB TGTCONS=SYSCMSTR
```

How Target Systems on a Linkage are Defined

Use the SYSID parameter on the LINK command to define the scope (target systems) to which cross-system commands can be issued. Specify one or more systems by system ID, or use an appropriate keyword to identify a group of systems.

Note: For more information, see the *CA MIM Statement and Command Reference Guide*.

Target Console Pool

A *target console* is a console through which CA MIC can issue commands on behalf of cross-system command issuers. The command response is then directed by the command processor to this console, CA MIC then redirects this response back to the original cross-system command issuer.

For CA MIC to be able to service operators issuing cross-system commands to a given target system, the target system must have a pool of target consoles through which these commands can be issued on the target system.

This section describes how this pool of consoles is built, what types of consoles are allocated to it, and how many to allocate. There are two different methods for building this pool of target consoles:

- **Dynamic Allocation-**CA MIC builds the target console pool automatically
- **Direct Allocation-**You build the target console pool

We recommend that you use the Dynamic Allocation method.

Dynamic Allocation

CA MIC can dynamically choose and allocate target consoles for use when issuing imported cross-system commands. To enable this function, specify a non-zero value on the SETOPTION MAXCONS command.

The MAXCONS value specifies the maximum number of dynamically allocated target consoles CA MIC acquires throughout the life of the CA MIC address space. Each console is allocated on demand until the MAXCONS limit is reached. Once the limit is reached, CA MIC continually re-assigns the least recently used target console to the current command source.

Note: Target console reassignment can be prevented with a POOL=DEDICATE type linkage.

Note: For more information, see How Linkage Types are Defined (see page 21) in this chapter.

By default, CA MIC allocates extended MCS consoles for all dynamically allocated target consoles. If, however, you have GCMINIT EXTCON=NONE specified to prevent CA MIC from allocating extended MCS consoles, then CA MIC attempts to allocate subsystem consoles for target consoles.

Note: For more information, see the *CA MIM Statement and Command Reference Guide*.

Dynamic Allocation of Extended MCS Consoles

Extended MCS consoles provide a programmatic means through a defined interface to issue commands, receive solicited command responses, and receive unsolicited WTO/WTOR/DOM message traffic. IBM has designed the extended console functional interface such that virtually an unlimited number of extended MCS consoles can exist in a single system or in a sysplex environment.

You do not need to specify anything to allow CA MIC to use extended MCS consoles for the dynamic target console pool. However, you can specify the GCMINIT EXTCON statement to customize the usage of extended consoles.

Note: For more information, see the *CA MIM Statement and Command Reference Guide*.

There are many advantages to using extended MCS consoles for the target console pool, including the following:

- **Alleviates MCS console constraint.** MCS consoles, both real and subsystem, are limited in number. For sysplex MCS console maximums, see the current IBM MVS Initialization and Tuning Reference.
- **Larger target console pools are possible.** This virtually eliminates the possibility of misrouting command responses due to target console reassignment.
- **Eliminate contention with other products for consoles.** CA MIC defines and allocates its own consoles. No other products have access to these consoles.
- **'SYS1.PARMLIB(CONSOLxx)' modifications are not necessary.** Extended MCS consoles are dynamically defined, allocated, and de-allocated. You do not need to do anything to exploit the use of extended consoles.

Dynamic Allocation of Subsystem Consoles

To allow CA MIC to use subsystem consoles for the target console pool, you must specify an adequate number of subsystem consoles (for CA MIC and other products) in SYS1.PARMLIB(CONSOLxx). See How You Define Additional Subsystem-Allocatable Consoles (see page 27) in this chapter.

Note: CA MIC cannot allocate any subsystem consoles already allocated to other subsystems. If all available subsystem-allocatable consoles are allocated to other subsystems, then CA MIC cannot build a target console pool.

Direct Allocation

To use the direct allocation method, choose what consoles are used by CA MIC to issue commands imported from external systems.

We *do not* recommend using the direct allocation method for several reasons:

- There is more work required of you, the installer, to build and maintain the target console pool.
- If you assign a specific subsystem console using the direct allocation method, then there is no guarantee that the console will be available when CA MIC attempts to allocate it. Another subsystem could have the specified consoles previously allocated, causing the CA MIC allocation efforts to fail.
- If you need to change the target consoles in the pool (that is, change the CONSLIST parameters), then you must shutdown and restart CA MIC for the change to take effect.

However, if you choose to use the direct allocation method, the following sections will help you implement it.

Consoles Eligible for Direct Allocation

Only the following types of consoles are eligible for direct allocation:

- **Inactive MCS Consoles** are MCS consoles that are not online when the system is initialized or that are varied offline currently.
- **Subsystem-Allocatable Consoles** are consoles that are defined to z/OS, but do not represent a real I/O device. Products or subsystems can use this type of console to enter commands to the z/OS operating system.

Note: Active MCS consoles may not be specified on a GCMINIT CONSLIST statement.

How You Implement the Direct Allocation Method

Before you can choose what consoles are to be used, you must first identify what consoles are available for direct allocation. Once you have chosen what consoles can be used, you tell CA MIC to allocate these consoles at startup by specifying them on the GCMINIT CONSLIST statement.

To implement the direct allocation method

1. Find the console names of inactive or subsystem consoles on each system by issuing the either of the following z/OS commands:

   ```
   D C,N
   D C,SS
   ```

   The consoles must be defined to z/OS and cannot be allocated to any other product or subsystem.

2. Use the GCMINIT CONSLIST statement to inform CA MIC which consoles to use for the pool on each system.

   For example, assume that SYSA has subsystem consoles named SYAASS05 , SYAASS07, and SYAASS12 available, and that SYSB has consoles SYAASS02, SYAASS03, and SYAASS07 available. You would include the following statements in the MIMINIT member of the MIMPARMS data set:

   ```
   IFSYS SYSA
      GCMINIT CONSLIST=(SYAASS05,SYAASS07,SYAASS12)
   ENDIF

   IFSYS SYSB
      GCMINIT CONSLIST=(SYAASS02,SYBBSS03,SYAASS07)
   ENDIF
   ```

3. Issue the following statement:

   ```
   SETOPTION MAXCONS=0
   ```

   This restricts the target console pool to consist only of those consoles identified in the GCMINIT CONSLIST statement. If MAXCONS is not set to 0, CA MIC allocates the consoles in the CONSLIST list first, and proceeds to dynamically allocate target consoles until the MAXCONS limit is reached.

How You Define Additional Subsystem-Allocatable Consoles

If you choose the direct allocation method or if you have GCMINIT EXTCON= NONE, then you may need to define additional subsystem-allocatable consoles to z/OS for use by CA MIC and possibly other products.

To add subsystem-allocatable consoles, add the following statement to SYS1.PARMLIB(CONSOLxx):

```
CONSOLE DEVNUM(SUBSYSTEM) AUTHORITY(ALL)
```

Each statement creates one subsystem-allocatable console during the next IPL.

How You View the Contents of the Target Console Pool

You can display the members of target console pool anytime by issuing the following command:

```
DISPLAY POOLCONSOLES
```

This display shows extended, subsystem, and inactive consoles assigned to the pool.

Note: If extended consoles are in use, then the following z/OS command also displays the extended consoles in use by CA MIC:

```
D C,KEY=micname
```

micname

Specifies the name of the CA MIC started task. You can see the key value associated with the CA MIC extended consoles by issuing the CA MIC DISPLAY POOLCONSOLES command.

How You Add or Delete Consoles from a Target Pool

You can add consoles to or delete consoles from a target pool. The maximum number of consoles in the pool can be increased dynamically by reissuing a SETOPTION MAXCONS command. Consoles are then added to the pool as cross-system commands are issued until the new MAXCONS limit is reached.

Consoles can also be deleted dynamically from the target console pool by issuing the FREECONS command. The SETOPTION MINCONS command can be used in conjunction with the FREECONS command. For example, setting MAXCONS=5 and MINCONS=2 in the MIMCMNDS member of the MIMPARMS data set allows CA MIC to dynamically allocate up to five consoles for the pool. The MINCONS value establishes the minimum number of consoles CA MIC retains should you issue a FREECONS MINCONS command. In our example, if CA MIC had five consoles in the pool, and a FREECONS MINCONS command is issued, then CA MIC would deallocate three consoles, and save two consoles for the console pool.

You can also use the FREECONS command to free all consoles, or a specific console, from the pool.

Note: For more information on the FREECONS command, see the *Statement and* Command *Reference Guide.*

How You Create Linkages

You can establish linkages to enable consoles, products, INTERNAL, INSTREAM, and TSO users to issue commands to any system. A linkage is referenced by the command source defined on the linkage. For an explanation of the LINK command and its parameters, see the *Statement and Command Reference Guide.* Use the following LINK command parameters to create a shared, dedicated, or exclusive linkage:

- Use the ADD parameter of the LINK command to indicate you are creating a linkage. If you specify ADD on a LINK command and the linkage already exists, then it is modified.
- Use the CONSOLE, TSOUSER, PRODUCT, SOURCE, INSTREAM, INTERNAL, or ALL parameter on the LINK command to enable a command source to issue cross-system commands.
- Use the SYSID parameter on the LINK command to restrict the target systems to which a command source can issue cross-system commands. Specify a single system ID, a list of system IDs, or a keyword specifying a group of systems.
- Use the AUTHORITY parameter on the LINK command to dictate the cross-system command authority level assigned to the command source. If the command source is an active MCS console, then the default cross-system command authority level is equal to the authority level that is currently assigned to the console by its SYS1.PARMLIB definition. If a console has MASTER authority, then the default is to allow it to have MASTER authority for cross-system commands as well.
- Use the TGTCONS or POOL parameters on the LINK command to define the type of linkage associated with the command source.

Examples: Creating Linkages

- To enable all local active MCS consoles to issue cross-system commands to any or all external systems, have CA MIC select target consoles from a shared console pool, and use the authority level of the command source (authority of the console issuing the cross-system command), issue the following command:

  ```
  LINK ADD ALL SYSID=EXTERNAL POOL=SHARE AUTHORITY=SOURCE
  ```

  You could omit POOL=SHARE and AUTHORITY=SOURCE because they are default values for the LINK command. ALL does not include extended consoles.

- To enable console MSTR to issue system control commands to system S2, and allow CA MIC to select and remove a target console from the console pool, issue the following command:

  ```
  LINK ADD CONSOLE=MSTR SYSID=S2 POOL=DEDICATE AUTHORITY=SYS
  ```

  You must make consoles available for the console pool on system S2 before console MSTR can issue commands through this linkage.

- To enable a local active MCS console named TAPECON to issue INFO, SYS, CONS, and IO authority level commands cross-system through target console SYSCMSTR on system SYSB, issue the following command:

  ```
  LINK ADD CONSOLE=TAPECON SYSID=SYSB AUTHORITY=ALL TGTCONS=SYSCMSTR
  ```

  Note: Console SYSCMSTR must be defined on SYSB before console TAPECON can issue commands to SYSB.

- To enable TSO CONSOLE user DA1GS21 to issue INFO authority level commands across systems, issue the following command:

  ```
  LINK ADD CONSOLE=DA1GS21 SYSID=ALL AUTHORITY=INFO
  ```

- To enable all consoles (real MCS or extended) with names beginning with ABC to issue MASTER, INFO, SYS, CONS, and IO authority level commands to all external systems, and select target consoles from a shared pool, issue the following command:

  ```
  LINK CONSOLE=ABC* SYSID=EXTERNAL AUTHORITY=MASTER POOL=SHARE
  ```

  Note: You can use the asterisk (*) or pound sign (#) wildcard characters only with the CONSOLE parameter.

How You Modify Linkages

Once a linkage for a command source has been created, you can modify it by:

- Changing its command authority level
- Adding target systems to a shared or dedicated linkage
- Changing a dedicated linkage to a shared linkage (or change a shared linkage to a dedicated linkage)
- Changing the target console for an exclusive linkage

For example, suppose you used this command to create a shared linkage:

```
LINK ADD CONSOLE=TAPECON1 SYSID=ALL AUTH=SYS
```

To increase the cross-system command authority on this linkage, issue the following command:

```
LINK ADD CONSOLE=TAPECON1 AUTH=MASTER
```

How You Delete Linkages

To delete a linkage, use the DELETE parameter on the LINK command and reference the command source. For example, suppose you used this command to enable product RCS to issue system control commands to system SYS1:

```
LINK PRODUCT=RCS SYSID=SYS1 AUTH=SYS
```

Because PRODUCT=RCS is the command source, you would issue the following command to delete the linkage for product RCS:

```
LINK DELETE PRODUCT=RCS
```

How You Modify Command Processing Settings

Site-specific command processing enhancements or restrictions can be implemented by customizing and activating one of three exit routines:

- GCMCMDXT-This exit routine prevents specific local commands from being routed to external systems.
- GCMRCVXT-This exit routine prevents specific external commands from being executed on the local system.
- MIMCMDXT-This exit routine prevents specific commands from being issued locally or externally.

How You Issue Cross-system Commands through Linkages

Once you create a linkage between a local command source and a target system, the command source can issue commands to the target system.

The authority level you assigned to that linkage determines what types of commands the command source can issue through the linkage. When you issue a cross-system command, CA MIC directs the command to the appropriate external system. The format for issuing cross-system commands is:

c (*scope*) *command*

c

Specifies the command prefix that tells CA MIM to intercept this command.

Note: For more information, see the description of the CMDPREFIX operand of the SETOPTION MIM or MIMINIT statements in the *Statement and Command Reference Guide.*

scope

Identifies the target systems to which the command should be routed and executed. (Specify ALL, ALLICMF, ALLSYS, EXTERNAL, EXTSYS, LOCAL, a system ID, a list of system IDs, or an ALIAS defined by the DEFALIAS command.)

command

Identifies the command to be executed. Specify the command as you would if you were issuing it from an MCS console, including a command character if appropriate. For example, include the command character for a CA MIM command, the JES command character ($) for a JES command, and so on. Do not use a command character for z/OS commands.

Sample Cross-system Command

The following is a sample cross-system command:

```
@SYSB @SETOPTION MAXCONS=6
```

@

This is the CA MIM command prefix that tells CA MIM to intercept and issue the cross-system command.

SYSB

Specifies a single system ID, a list of system IDs, an alias, or a keyword used to direct the command to the appropriate systems.

@

This is the CA MIM command prefix that tells CA MIM on SYSB to execute the SETOPTION command. You could specify F MIMGR in place of the @ prefix.

SETOPTION MAXCONS=6

This is the command that CA MIM on SYSB executes.

How You Define Target System Aliases

You can use the DEFALIAS command to define site-specific aliases that represent a group of system IDs. When issuing cross-system commands to those systems, use the defined alias keyword instead of listing system IDs separately.

For example, you could issue the following DEFALIAS command to establish an alias that represents three systems:

```
DEFALIAS COMPLEX1 (SYS1,SYS2,SYS3)
```

Then, if you issue the command @COMPLEX1 D T, the D T command is issued on systems SYS1, SYS2, and SYS3.

How You Issue Cross-system Commands from Other CA Software Products

If you are using the CA Remote Console or CA OPS/MVS products, then you can enable these products to issue cross-system commands. To do this, issue these LINK commands:

```
LINK PRODUCT=RCS  SYSID=ALL AUTHORITY=SOURCE
LINK PRODUCT=OPSS SYSID=ALL AUTHORITY=SOURCE
```

In these commands, RCS and OPSS are the subsystem names used to identify CA Remote Console and CA OPS/MVS, respectively, to the operating system.

Note for CA OPS/MVS Users: For more information, see CA OPS/MVS Considerations (see page 44) in this chapter.

Examples:

To reply WAIT to outstanding WTOR reply ID 3 on SYSB, issue a command in one of these ways:

- @SYSB 3,WAIT
- @BB 3,WAIT
- The BB value represents the system alias as defined in the DEFSYS statement.
- @02 3,WAIT

The 02 value in the last command represents the internal index number of SYSB. This number is determined by the order in which systems are defined on the DEFSYS statement. In this example, it is the second system defined. The system name and alias fields are also acquired from the DEFSYS statement.

Note: You can check system names, aliases, and index numbers by issuing the DISPLAY SYSTEMS command.

1. Assume TSO user USERID1 is logged on to SYSA. To enable USERID1 to issue display commands to the TEST system, issue this cross-system LINK command:

   ```
   @SYSA @LINK TSOUSER=USERID1 SYSID=TEST AUTH=INFO
   ```

   CA MIM on the local system routes this command to CA MIM on SYSA.

2. To issue the JES2 DISPLAY INITIATOR command to all systems except the local system, issue this cross-system command:

   ```
   @EXT $DI
   ```

   The dollar character ($) is the JES2 command character. CA MIM on the local system routes this command to all external systems. The responses are returned to the local console that issued the command.

3. To issue the z/OS DISPLAY TIME command on all systems, including the local system, issue this cross-system command:

   ```
   @ALL D T
   ```

4. To issue the z/OS DISPLAY ACTIVE command to systems SYSA, SYSB, and SYSC using the alias SITE1, you would issue the following two commands in the order shown:

   ```
   DEFALIAS SITE1 (SYSA,SYSB,SYSC)
   @SITE1 D A,L
   ```

5. To issue a DISPLAY OVERLAYS command to a command processor named OLDCMDPR, issue this cross-system command on system SYSB:

   ```
   @(SYSB) F OLDCMDPR,DISPLAY OVERLAYS
   ```

How You Issue Commands through Indirect Command Routing

Through indirect command routing, you can route commands between systems that are not directly connected to each other. Indirect routing can be used to issue cross-system commands, but not to collect unsolicited WTOs. Indirect command routing is done through an intermediary system.

For example, if systems A and B are communicating through a DASD control file, and systems B and C are connected through ICMF, then you can issue a command from system A to C, using system B as the intermediary. System A receives the response to the command executed on system C.

You can also use indirect command routing to issue commands between systems that use different control files. Although you can use indirect routing along with direct CA L Serv routes, you can use indirect routing *instead* of CA L Serv routes when you do not want to create more CA L Serv routes.

To create an indirect route, issue a series of LINK commands that connect a source system to an intermediary system and that connect the intermediary system to the system where the command will be executed. The LINK commands create a series of connections like this:

Source System > Intermediary System > Target System

Note the following important considerations:

- If you are using inactive or subsystem consoles as target consoles on the intermediary system, then you must specify PRODUCT=GCM on the LINK command you issue from the intermediary system.
- If you are using extended consoles as target consoles on the intermediary system, then you must specify CONSOLE=*prefix** on the LINK command you issue from the intermediary system (where *prefix* is the prefix specified on the GCMINIT EXTCON command, and * is the wildcard character).
- If you use a mixture of console types for target consoles on the intermediary system, then you need to specify both types of LINK commands.

For example, suppose you are running CA MIC on three systems: SYSA, SYSB, and SYSC. Also suppose these systems are connected as in the previous section, with SYSA the source system, SYSB the intermediary system, and SYSC the target system.

1. To create an indirect route between console TAPECON1 on system SYSA and system SYSC, issue the following command on system SYSA:

   ```
   LINK CONSOLE=TAPECON1,SYSID=SYSB,AUTHORITY=INFO
   ```

2. Then, to create the intermediary linkage from system SYSB to system SYSC, issue one of the following commands on system SYSB:

   ```
   LINK PRODUCT=GCM,SYSID=SYSC
   LINK CONSOLE=GCM*,SYSID=SYSC
   ```

   GCM

   Specifies the prefix specified on the GCMINIT EXTCON statement

Commands directed to system SYSC are executed, and the responses are routed back to the command source on SYSA through the intermediary system SYSB.

Assuming that the command character for CA MIM on both SYSA and SYSB is @, the following example show you how to issue a z/OS D J,L command from console TAPECON1 on system SYSA and have that command execute on system SYSC:

```
@SYSB, @SYSC, D J,L
```

When you issue this command, CA MIM on SYSA intercepts it and directs it to system SYSB through the linkage for console TAPECON1. System SYSB then receives and executes the command @SYSC, D J,L.

CA MIM then uses the prefix @SYSC to direct this command to system SYSC through the linkage to GCMF (created by specifying either PRODUCT=GCM or CONSOLE=GCM*).

SYSC then receives and executes the command D J,L. CA MIM on SYSC routes the command response back to console TAPECON1 on SYSA through the intermediary system SYSB.

Message Processing

You can manage cross-system message traffic by defining message routing definitions.

How You Create Message Collection Sets

Message routing definitions, or *collection sets*, are defined using the COLLECT command. A collection set is identified by its local destination unless the SETNAME parameter is also specified.

The COLLECT command differs from the LINK command in that collection sets are used to route UNSOLICITED WTOs, such as tape mounts, job start and end messages, and printer messages. Linkages are used to issue cross-system commands and automatically receive command responses (SOLICITED WTOs). These responses are returned to the issuing system, regardless of any collection sets defined.

Collection sets define the routing of unsolicited cross-system messages. The COLLECT command defines a collection set by identifying:

- Types of messages that need to be imported to the local system
- Systems from which the messages are to be imported
- Local destination to which the imported messages are to be routed

Message selection criteria are quite flexible, as is the number of local destinations that can receive imported messages. As a result, you can create collection sets tailored to the requirements of your site. Some of the most common uses of collection sets are:

- Direct imported messages to all local consoles, individual local consoles, local products, or local TSO users
- Record imported messages to the local SYSLOG
- Direct messages from specific jobs to specific TSO users

Note: For more information about the COLLECT command and COLLECT command syntax, see the *Statement and Command Reference Guide.*

You can create a collection set that broadcasts messages to all consoles assigned the same routing code or monitor type. For example, you can route cross-system tape messages to any local console that receives local tape messages, while cross-system unit record messages are only presented to local consoles accepting local unit record messages.

How You Modify Collection Sets

To modify a collection set, specify the ADD parameter on your COLLECT command. If the collection set already exists, then CA MIC modifies it.

You can modify only certain parts of a collection set. You cannot change the destination and SETNAME parameters. However, you can modify a collection set in these ways:

- Add collection criteria to receive different types of messages.
- Add source systems to receive messages from more systems.
- Change the value for the inclusion parameter (ALL or ANY) to receive messages meeting a different number of requirements.

How You Add Collection Criteria to a Collection Set

To add inclusion or exclusion parameters to a collection set, specify those parameters on your COLLECT command. Also provide the destination and name of the collection set so that CA MIC knows which collection set to modify.

Suppose a collection set named DEFAULT is collecting messages from all external systems and is routing those messages to the console named MSTR. To begin collecting messages from job LEDGER (in addition to any messages already being collected), issue the following command:

```
COLLECT CONSOLE=MSTR JOBNAME=LEDGER
```

The default value SETNAME=DEFAULT is omitted from this command.

You cannot add a parameter that is mutually exclusive with any other parameter in that collection set. For example, you cannot add the NOACTION parameter if the ACTION parameter is already specified for that collection set.

Example:

To add new collection criteria to a collection set, specify the new criteria on a COLLECT command. The following example shows you how CA MIC adds collection criteria to an existing collection set:

Start with the following collection set:

```
CONSOLE=MSTR SETNAME=SAMP13,
SYSID=EXTERNAL MSGID=($HASP*,IEF*),
JOBNAME=TEST1 ANY
```

Issue the COLLECT command:

```
COLLECT CONSOLE=MSTR,
SETNAME=SAMP13 JOBNAME=TEST2
```

This results in the following new collection set:

```
CONSOLE=MSTR SETNAME=SAMP13,
SYSID=EXTERNAL MSGID=($HASP*,IEF*),
JOBNAME=(TEST1,TEST2) ANY
```

CA MIC adds the value JOBNAME=TEST2 to this collection set. As a result, messages from job TEST2 are collected for console MSTR.

How You Add Source Systems to a Collection Set

To add a new source system to a collection set, specify the ID of that system on the SYSID parameter of your COLLECT command. Also provide the local destination of the collection set so that CA MIC knows which collection set to modify.

Suppose a collection set named MASTER is collecting messages from certain external systems to console MSTR. To begin collecting messages from system 01 (in addition to messages already being collected from other systems), issue the following command:

```
COLLECT CONSOLE=MSTR SETNAME=MASTER SYSID=01
```

Example:

To add a new source system to a collection set, name that system on a COLLECT command. The following example shows you how CA MIC adds another source system to an existing collection set:

1. Start with the following collection set:

   ```
   CONSOLE=MSTR SETNAME=SAMP13,
   SYSID=01 MSGID=($HASP*,IEF*),
   JOBNAME=TEST1 ANY
   ```

2. Issue the following command:

   ```
   COLLECT CONSOLE=MSTR,
   SETNAME=SAMP13 SYSID=02
   ```

   This results in the following new collection sets:

   ```
   CONSOLE=MSTR SETNAME=SAMP13,
   SYSID=(01,02) MSGID=($HASP*,IEF*),
   JOBNAME=TEST1 ANY
   ```

CA MIC adds the value SYSID=02 to this collection set. As a result, CA MIC begins collecting the appropriate messages from system 02.

Changing the Value for the Inclusion Parameter

By changing the value for the inclusion parameter, ALL or ANY, you can change the number of requirements a message must meet to be collected. To change the value for the inclusion parameter, specify a new value on your COLLECT command. You also must provide the destination and name of the collection set.

Suppose a collection set named DEFAULT is collecting messages for console MSTR. Currently, this collection set is selecting messages from job LEDGER and action messages. That is, messages meeting either of these requirements are selected. To begin collecting only ACTION messages issued by job LEDGER (that is, messages meeting both requirements), issue the following command:

```
COLLECT CONSOLE=MSTR ALL
```

The default value SETNAME=DEFAULT is omitted from this command.

How You Replace Collection Sets

To replace an existing collection set with a new one, specify the REPLACE parameter on your COLLECT command. The new collection set created by your COLLECT command replaces the existing collection set. Also provide the destination and name of the collection set on your COLLECT command so that CA MIC knows which collection set to replace.

Suppose a collection set named DEFAULT is collecting messages for console TAPECON1. To replace this collection set with a new one that collects messages from job LEDGER, instead of the messages that are currently being collected, issue the following command:

```
COLLECT REPLACE CONSOLE=TAPECON1 JOBNAME=LEDGER
```

The default value SETNAME=DEFAULT is omitted from this command.

Example:

To replace a collection set, specify the REPLACE parameter on a COLLECT command. The following example shows you how CA MIC replaces an existing collection set with a new collection set:

1. Start with the following collection set:
   ```
   CONSOLE=MSTR SETNAME=RULE1,
   SYSID=EXTERNAL MONITOR=SESSION,
   ROUTCDE-(3,5) ANY
   ```
2. Issue the following command:

   ```
   COLLECT REPLACE SETNAME=RULE1,
   ACTION WTOR
   ```

   This results in the following new collection sets:

   ```
   CONSOLE=MSTR SETNAME=RULE1,
   SYSID=EXTERNAL ACTION WTOR ANY
   ```

Because you specified the REPLACE parameter on the COLLECT command, CA MIC replaces the existing collection set with the new one created by your COLLECT command.

This example assumes you are issuing the COLLECT command from console MSTR. Otherwise, include the value CONSOLE=MSTR on your COLLECT command.

How You Delete Collection Sets and Collection Criteria

To delete a collection set or part of a collection set, specify the DELETE parameter on the COLLECT command. You can delete message routing information in the following ways:

- Remove source systems from a collection set
- Delete collection criteria from a collection set
- Delete the entire collection set

How You Remove a Source System from a Collection Set

To remove a source system from a collection set, specify the DELETE parameter and the ID of the system on your COLLECT command. Also provide the destination and name of the collection set so that CA MIC knows which collection set you want.

Suppose a collection set named DEFAULT is collecting messages from several systems (including system S1) for console MSTR. To stop collecting messages from system S1, issue the following command:

```
COLLECT DELETE CONSOLE=MSTR SYSID=S1
```

```
The default value SETNAME=DEFAULT is omitted from this command.
```

Example:

To remove a source system from a collection set, specify the ID of the system and the DELETE parameter on a COLLECT command. The following example shows you how CA MIC removes a source system from a collection set:

1. Start with the following collection set:

   ```
   CONSOLE=MSTR SETNAME=SAMP13,
   SYSID=(01,02) MSGID=($HASP*,IEF*),
   JOBNAME=TEST1 ANY
   ```

2. Issue the following command:

   ```
   COLLECT DELETE CONSOLE=MSTR,
   SETNAME=SAMP13 SYSID=02
   ```

3. This results in the following new collection set:

   ```
   CONSOLE=MSTR SETNAME=SAMP13,
   SYSID=01 MSGID=($HASP*,IEF*),
   JOBNAME=TEST1 ANY
   ```

Because you named a source system on the COLLECT command, CA MIC deletes only that information from the collection set. CA MIC deletes the value SYSID=02 from this collection set so that console MSTR no longer receives messages from system 02.

How You Delete Collection Criteria from a Collection Set

To delete collection criteria, specify the DELETE parameter and the collection criteria on your COLLECT command. Also provide the destination and name of the collection set so that CA MIC knows which collection set you want.

Suppose a collection set named DEFAULT is collecting specific messages, including messages that have the prefix IEF, for console MSTR. To stop collecting messages that have the prefix IEF, issue the following command:

```
COLLECT DELETE CONSOLE=MSTR MSGID=IEF*
```

This example illustrates the removal of a selection criterion from a collection set. The CA MIC command set can often achieve similar results by using alternate features, and that can be confusing. If the initial COLLECT command had specified MSGID=IE*, then the previous COLLECT DELETE command would not have worked. Instead, you could have specified NOMSGID=IEF*.

The default value SETNAME=DEFAULT is omitted from this command.

Example:

To delete collection criteria from a collection set, specify that criteria and the DELETE parameter on a COLLECT command. The following example shows you how CA MIC deletes a message collection parameter from a collection set:

1. Start with the following collection set:

   ```
   CONSOLE=MSTR SETNAME=DEFAULT,
   SYSID=EXTERNAL MSGID=$HASP*,
   NOJOBNAME=TEST* ACTION ANY
   ```

2. Issue the COLLECT command:

   ```
   COLLECT DELETE CONSOLE=MSTR,
   MSGID=$HASP* NOJOBNAME=TEST*
   ```

   This results in the following new collection sets:

   ```
   CONSOLE=MSTR SETNAME=DEFAULT,
   SYSID=EXTERNAL ACTION ANY
   ```

Because you specified collection criteria on the COLLECT command, CA MIC deletes only that information from the collection set. CA MIC deletes the values MSGID=$HASP* and NOJOBNAME=TEST* from this collection set.

How You Delete a Collection Set

To delete a collection set, specify the DELETE parameter and the destination and name of the collection set on your COLLECT command. For example, to delete the collection set named DEFAULT for console MSTR, issue the following command:

```
COLLECT DELETE CONSOLE=MSTR
```

The default value SETNAME=DEFAULT is omitted from this command.

Note: It is possible to have more than one collection set for the same destination by giving each collection set a different name (using the SETNAME parameter). When this is the case, you cannot delete all collection sets for a destination by issuing a single COLLECT DELETE command. You must issue a separate COLLECT DELETE command for each collection set.

Example:

To delete a collection set, specify the DELETE parameter without any collection criteria on a COLLECT command. The following example shows you how CA MIC deletes a collection set:

1. Start with the following collection set:

   ```
   CONSOLE=MSTR SETNAME=RULE1,
   SYSID=ALL MONITOR=SESSION,
   ROUTCDE=(3,5) ANY
   ```

2. Issue the following command:

   ```
   COLLECT DELETE SETNAME=RULE1
   ```

   This results in no new collection sets. Because you did not specify any collection criteria on the COLLECT command, CA MIC deletes the entire collection set.

This example assumes you are issuing the COLLECT DELETE command from console MSTR. Otherwise, include the value CONSOLE=MSTR on your COLLECT DELETE command.

How You Direct Messages to CA Remote Console

If you are using the CA Remote Console product, you can direct messages from other systems to CA Remote Console. For example, to send all messages that are assigned routing codes to CA Remote Console, issue the following command:

```
COLLECT PRODUCT=RCS SYSID=EXT ROUTCDE=ALL
```

RCS is the name that CA Remote Console uses to identify itself to the operating system. If you specify a different value for the SSNAME parameter on the RCSINIT statement for CA Remote Console, then specify that value instead of RCS on the COLLECT command. The first three characters of the value specified on the RCSINIT SSNAME statement must be RCS to ensure that CA MIC and CA Remote Console interface properly.

You still need to use the RCOLLECT command for CA Remote Console to indicate which messages an individual CA Remote Console session should receive.

Note: Make sure the system ID assigned to a system through the DEFSYS statement is identical to the system ID assigned to that system by the DEFSYS statement for CA Remote Console.

CA OPS/MVS Considerations

This section discusses how to manage command and message routing using the interface between the CA OPS/MVS and CA MIC products.

The interface between CA OPS/MVS and CA MIC provides the following capabilities:

- The CA OPS/MVS subsystem can receive unsolicited messages from any system in the MICplex and record them in the OPSLOG.
- The CA OPS/MVS subsystem can issue cross-systems commands through the CA MIC subsystem by using the OPSCMD command processor or the ADDRESS OPER OPS/REXX host command environment, to any system in the MICplex. The solicited command response messages are returned to the issuer and may optionally be recorded in the OPSLOG.
- CA OPS/MVS subsystem AOF rules can recognize and interrogate fields from solicited and unsolicited CA MIC imported messages and take action based on the message data presented.

The MICplex can consist of up to 128 systems configured in a single sysplex, non-sysplexed systems, systems in multiple sysplexes, or z/VM systems where CA MIC is running as a service machine. Any CA OPS/MVS subsystem can interrogate messages from up to 128 systems.

Note: CA L-Serv is required for more than 32 systems.

Message Presentation to CA OPS/MVS

CA MIC-imported messages are directed from the local CA MIC subsystem to the local CA OPS/MVS subsystem using standard Subsystem Interface protocols. Certain WQE control block fields for imported messages are altered by the CA MIC subsystem before presentation to the CA OPS/MVS subsystem. These processes allow the local CA OPS/MVS subsystem to determine that a given message came from the CA MIC subsystem, and that the message originated on an external system. The CA MIC subsystem changes the following WQE control block fields:

- The WQERISS bit in the WQEFLG1 field is set to ON.
- The WQEFORN bit in the WQEFLG2 field is set to ON.
- The WQESYSNM field contains the name of the system from which the message originated.
- The WQEJOBNM field contains the job number of the task that originally issued the message. When the originating task was a z/OS subsystem or a z/VM application that did not have a job number, the OPSLOG column that displays this information shows a value of “NONE”.
- The WQEOJBID field contains the job number of the task that originally issued the message. This field contains the z/OS subsystem name or the z/VM application name when the originating task was an OS subsystem or a z/VM application, which did not have a job number.
- The WQEOJBNM contains the name of the task that originally issued the message.

CA MIC presents imported messages to CA OPS/MVS using these standards regardless of any CA MIC message editing parameter values in effect on any system. In other words, CA MIC consistently presents CA OPS/MVS with original message data regardless of the CA MIC message editing that may have taken place on a given system based on the CA MIC MIMINIT EDITMESSAGE, SYSNAME, SYSTYPE, and JOBID parameters.

When the local CA MIC subsystem is directing imported messages to the local CA OPS/MVS subsystem, it is important that CA OPS/MVS AOF rules compare the MSG.SYNA environmental variable (the origin system name) with the result of the OPSINFO("SYSNAME") function (the local system name) to identify the systems from which messages are originating. Otherwise, CA OPS/MVS rules may misinterpret CA MIC imported messages as being from the local system, which may result in unpredictable actions by the local CA OPS/MVS subsystem.

Several sample CA OPS/MVS AOF rules and message variables are distributed with the CA OPS/MVS product that may be useful to those who want to use this interface. These samples allow imported messages to be colorized and identified in other ways in OPSLOG and to AOF rules. There are also certain CA OPS/MVS operating parameter values that need to be defined properly before this interface can be operational.

Note: For more information about required CA OPS/MVS parameters settings, as well as useful CA OPS/MVS message variables and sample AOF rules that may be of value to those using this interface, see the *CA OPS/MVS AOF Rules Guide*.

Cross-system Message Routing

Using the CA MIC COLLECT command, CA MIC can import (collect) messages to a local CA OPS/MVS subsystem using any number of message selection or exclusion criteria.

Consider a MICplex consisting of SYSA, SYSB, SYSC, and VME. You want a CA OPS/MVS subsystem named “OPSS” to receive messages from SYSB, SYSC, and VME. You would issue the following command that could be placed in the MIMCMNDS member of the CA MIM parmlib:

```
IFSYS SYSA
COLLECT PROD=OPSS SYSID=EXT ROUTCDE=ALL ACTION WTOR
                            MONITOR(JOBNAME,SESSION,STATUS)
ENDIF
```

After this command is executed on SYSA, the CA OPS/MVS subsystem named OPSS on SYSA would begin receiving messages from system SYSB, SYSC, and VME that meet the message criteria defined on the COLLECT command. Note that the SYSID parameter should not be set to “ALL”, nor should it include the local system name. The reason for this is that since the local CA OPS/MVS subsystem is already receiving local messages itself, it does not need CA MIC to present it with local messages as well. Therefore the SYSID value only needs to include the external system names (or EXT) from which you want CA MIC to import messages.

You may have a need to collect messages from external systems using different message criteria based on the originating system. The SETNAME= parameter on the COLLECT command can be used to define multiple collection sets for a single CA OPS/MVS subsystem:

```
IFSYS SYSA
      COLLECT PROD=OPSS SETNAME=SETB SYSID=SYSB ROUTCDE=ALL
      COLLECT PROD=OPSS SETNAME=SETC SYSID=SYSC ROUTCDE=(3,5) WTOR
      COLLECT PROD=OPSS SETNAME=SETE SYSID=VME  MSGID=HCP*
ENDIF
```

After these COLLECT commands are executed on SYSA, the CA OPS/MVS subsystem named OPSS on SYSA would begin receiving unique messages from each external system. SYSB would send messages with any route code (1-128); SYSC would send only messages with route codes of 3 or 5, and any WTOR; and VME would send only messages starting with the text of “HCP.”

Another possible configuration is where you have not one CA OPS/MVS subsystem on SYSA, but two or more, and you want CA MIC to import messages to each local CA OPS/MVS subsystem. Since all local CA OPS/MVS subsystems must have unique subsystem names that begin with “OPS”, there are two ways to have CA MIC import messages to multiple local CA OPS/MVS subsystems. The PROD= keyword on the COLLECT command can be used to direct imported CA MIC messages to *all local CA OPS/MVS subsystems*, or to direct imported messages to only one or more individual CA OPS/MVS subsystems. When COLLECT PROD=OPSS is specified, imported messages are routed only to the CA OPS/MVS subsystem named “OPSS.” When COLLECT PROD=OPS is specified, imported messages are routed to *all locally active CA OPS/MVS subsystems*. Note that when using the COLLECT PROD=OPS method, you should not also define COLLECT commands for individual CA OPS/MVS subsystems. Otherwise, those individual CA OPS/MVS subsystems will receive duplicate imported messages.

Hardcopy-only Message Routing

By default, CA MIC does not permit hardcopy-only messages to be routed across systems. However, you can optionally activate the CA MIC hardcopy-only message routing feature by specifying the SETOPTION CAPTURELOG=YES command in the MIMCMNDS member of the CA MIM parmlib. Once this has been done, you can add the LOGONLY parameter to your COLLECT commands so that only certain destinations receive the hardcopy-only messages.

For example, if you want the OPSS subsystem on SYSA to receive hardcopy-only messages (in addition to other message types) from systems SYSB and SYSC, you would add the LOGONLY parameter to your OPSS COLLECT command:

```
IFSYS SYSA
      COLLECT PROD=OPSS SYSID=(SYSB,SYSC) ROUTCDE=ALL ACTION WTOR LOGONLY
                                          MONITOR(JOBNAME,SESSION,STATUS)
ENDIF
```

After this COLLECT command executes on SYSA, the OPSS subsystem on SYSA begins receiving hardcopy-only messages (in addition to other message types) from SYSB and SYSC.

Suppressed Message Routing

CA MIC will not route to any external system messages that have been suppressed by a local subsystem or by MPF. In other words, CA MIC honors any local message suppression and prevents locally suppressed messages from appearing on other systems in the MICplex.

Cross-system Command Routing

Using the CA MIC LINK command, CA MIC can accept commands from a local CA OPS/MVS subsystem and route them to the designated external systems. CA MIC then returns the command response messages to the local CA OPS/MVS subsystem.

Consider a MICplex consisting of SYSA, SYSB, SYSC, and VME, where you want a CA OPS/MVS subsystem named OPSS to have the ability to issue cross-system commands to SYSB, SYSC, and VME. You would issue the following command that could be placed in the MIMCMNDS member of the CA MIM parmlib:

```
IFSYS SYSA
      LINK PROD=OPSS SYSID=ALL AUTH=MASTER
ENDIF
```

This command establishes a cross-system command linkage between the OPSS subsystem on SYSA and SYSB, SYSC, and VME. Once this LINK command executes, the OPSS subsystem can issue commands to any system in the MICplex. The command response messages are returned to the command issuer, recorded in the OPSLOG, and can optionally be acted upon by the subsystem OPSSAOF rules.

Standard CA MIC command routing prefixes can be used by OPSS to direct commands to one or more systems in the MICplex. Here are a few examples using a colon (:) as the CA MIC command prefix character:

```
:ALL, D A,M*
```

CA MIC picks up this command issued by OPSS, and the D A,M* command is directed to *all* systems SYSA, SYSB, SYSC, and VME. CA MIC directs the command responses from each system to the OPSS subsystem.

```
:EXT, DISPLAY TIME
```

CA MIC picks up this command issued by OPSS and the DISPLAY TIME command is directed to external systems SYSB, SYSC, and VME. CA MIC returns the command responses from each system to the OPSS subsystem.

```
:VME, QUERY TAPE
```

CA MIC picks up this command issued by OPSS and the QUERY TAPE command is directed to system VME. CA MIC returns the command response from VME to the OPSS subsystem.

Product Activation and Termination Considerations

For information about starting and stopping CA MIC, see the CA MIM *Programming Guide*.

Intersystem Communication Facility

The Intersystem Communication Facility (ICMF) is the CA MIC facility used to transmit data (messages and commands) through the communication server of CA L-Serv. The communication server component of CA L-Serv transmits data between systems connected through VTAM. The ICMF communication method can only be used by CA MIC to transmit data.

You can use this method as your exclusive cross-system communication method, or together with other communication methods.

If the CA L-Serv route is active between two systems, then CA MIC chooses the ICMF method over other available communication methods (such as CTCDASD, CTCONLY, DASDONLY, and XCF) that may be available. If the CA L-Serv route is deactivated, then CA MIC automatically begins transmitting data using the other communication method, if available.

Note: For more information about installing and using CA L-Serv, see the CA L-Serv documentation.

How You Identify the Systems That Use ICMF

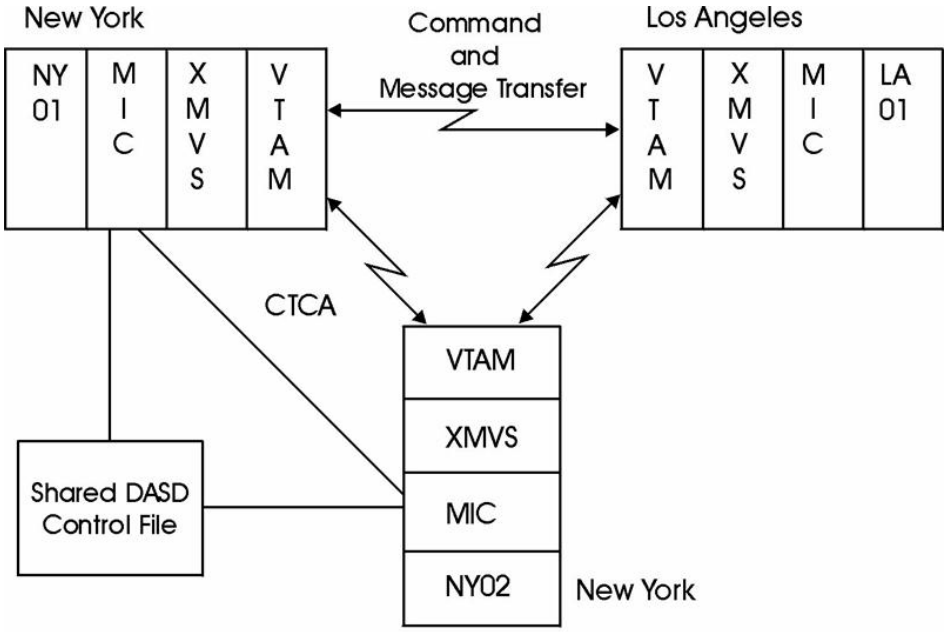
The IDEFSYS statement is used to identify the systems that use the ICMF communication method. IDEFSYS is similar to the CA MIM DEFSYS statement, except that it only identifies ICMF systems. DEFSYS is used to identify systems using VCF communication methods, DASD control file communication methods, or both. The IDEFSYS statement provides two critical items of information about a system:

- The one- to eight-character system name
- A two-character system alias

Each system in the CA MIM complex must have a unique system name and alias. However, if two systems have both an ICMF and shared DASD control file connection, then the system name and alias can be defined identically on the DEFSYS and IDEFSYS statements. For example, if two systems can communicate through a shared DASD control file and they also have an ICMF connection, and are both connected to a third system only through ICMF, then the statement definitions would look like this:

```
DEFSYS(NY01,N1,NY01),(NY02,N2,NY02)
IDEFSYS(NY01,N1),(NY02,N2),(LA01,LA)
```

The DEFSYS statement defines systems using a DASD or VCF communication method, while the IDEFSYS statement defines systems using the ICMF communication method. The following figure illustrates how the systems defined in the preceding example are configured for the different communication methods:



Note that XMVS is the z/OS cross-system communication component of CA-L-Serv.

We strongly recommend that you specify an IDEFSYS command in the MIMINIT member in order to identify the ICMF systems during CA MIM initialization.

You can route *only* console traffic (that is, cross-system commands and messages) through ICMF. CA MIM does not use ICMF to route ENQ requests or device allocation information among systems.

If you do not specify an IDEFSYS statement in the MIMINIT member, and CA MIM determines that an ICMF route is active between two systems, then ICMF builds internal definitions based on the following defaults:

- By default, ICMF uses the *system name* on the DEFSYS statement; that is, the name a system has in a complex using control files for communication. If you did not define a name for that system on a DEFSYS statement, then ICMF uses the SMF ID of the system instead.
- By default, ICMF uses the *system alias* on the DEFSYS statement; that is, the alias a system has in your control file complex. If you did not define an alias for that system on a DEFSYS statement, then ICMF uses the index number for that system as the alias.

Note: An ICMF index number is different than a control file system index number: ICMF index numbers are 33 or higher, while control file system index numbers are from 1 to 32.

Performance Considerations

CA MIC can be used to collect unsolicited messages from external systems and route them to a local console or product. These messages can be delivered to the local system through CTC, XCF, or XES coupling facility list structures, DASD, or CA L-Serv (using VTAM). The amount of control file activity for CA MIC is dependent on the message traffic on external systems, the COLLECT command, and a few other commands.

How You Filter Messages

The SET GCMF MSGFILTER command parameter can be used to determine how many copies of an identical message are routed to external systems. For example, if a message loops continuously, then CA MIM routes many copies of this message to external systems assuming a COLLECT command exists for this message. This would result in increased control file traffic and possibly storage shortages. To prevent this from occurring, you can modify the SET GCMF MSGFILTER command.

For example:

```
SET GCMF MSGFILTER=50
```

You can specify a number between 1 and 1000. To deactivate this feature, specify:

```
SET GCMF MSGFILTER=OFF
```

How You Make Messages Eligible for External Routing

Typically, CA MIC does not allow local messages destined only for the system log to be routed externally. Hardcopy-only messages are recorded to the local SYSLOG, but are not displayed at local consoles. To make these messages eligible for external routing, specify the command SETOPTION CAPTURELOG=YES on the system where the messages are generated. When you use CAPTURELOG=YES, local messages that are marked as system log messages (also known as hardcopy-only) can be routed externally. When you use CAPTURELOG=NO, no hardcopy-only messages are routed externally from this system.

When NOLOG is specified on a collection set, hardcopy-only messages are not collected, even if CAPTURELOG=YES is specified on the system where the messages originate.

You can specify that you want to collect hardcopy-only messages that do not match any other criteria, by using the LOGONLY parameter on the COLLECT command. If you specify LOGONLY, and no other COLLECT criteria, then every hardcopy-only message generated on the systems from which you are collecting messages is sent to that collection set, provided those systems have CAPTURELOG=YES set.

NOLOG and LOGONLY are collection set selection criteria, like MSGID. The inclusion parameters, ANY and ALL, affect NOLOG and LOGONLY the way they affect MSGID. NOLOG and LOGONLY are not opposites of each other, but are mutually exclusive, meaning that you cannot specify both on the same COLLECT command. The following examples show you how to specify collection sets for routing hardcopy-only messages:

- To collect hardcopy-only, ISTxxxx messages from external systems where the command SETOPTION CAPTURELOG=YES is set, to TSO user DSIAZ11, you would issue the following COLLECT command:

  ```
  COLLECT ADD TSOUSER=DSIAZ11 MSGID(IST*) LOGONLY,
     ALL SYSID=EXTERNAL
  ```

- To collect all ISTxxxx messages generated by external systems and all hardcopy-only messages from external systems where CAPTURELOG=YES is set to TSO user DSIAZ11, you would issue the following COLLECT command:

  ```
  COLLECT ADD TSOUSER=DSIAZ11 MSGID(IST*) LOGONLY,
     ANY SYSID=EXTERNAL
  ```

Note: Routing hardcopy-only messages to external systems significantly increases the number of messages processed by CA MIC. This results in an increase in overhead and storage used by the CA MIC address space.

Sysplex Considerations

Keep in mind the following when using CA MIC in a sysplex:

- Users of CA Remote Console who use CA MIC to collect messages from external systems on local CA Remote Console sessions should specify SYSPLEX=NO on the RCSINIT statement to prevent duplicate messages from being displayed at the local RCS session.
- After implementing SYSPLEX, you may see duplicate messages from external systems on local consoles. To prevent this, change the MSCOPE parameter on the local console to 'local-system-id', or '*'.

  For example, if D C,* shows that MSCOPE=*ALL, and this console is receiving duplicate messages, then enter the following z/OS command:

  ```
  VARY console,MSCOPE=*
  ```

- We recommend you use CONSOLE NAMES on the CA MIC LINK and COLLECT commands, rather than CONSOLE IDs. In a SYSPLEX, the CONSOLE IDs (on each system) may change, depending on the startup order of the systems in the SYSPLEX, while CONSOLE NAMES remain constant.
- The GCMINIT REPLYLIMIT=(0000,9999) statement limits WTOR reply ID values in a CA MIC complex. If a GCMINIT REPLYLIMIT statement is found during startup in a sysplex environment, then the statement is ignored. Additionally, the REPLYLIMIT keyword is not shown in the D GCM INIT display.

z/VM Considerations

CA MIC for z/VM is the component of the CA MIM product that provides you with cross-system command and message routing services in a multiple-system or multiple-image environment.

CA MIC provides the following:

- Provides you with cross-system command routing capability. With CA MIC, a user or product on one system can issue commands to any system in your configuration.
- Lets you collect unsolicited messages on a z/VM system and distribute them to one or more destinations on a z/OS system. This enables you to monitor activities on all systems from a single location.
- Lets you customize command and message routing.

CA MIC for z/VM uses the Global Command and Message Facility (GCMF). CA provides a stub for the CA MIC ICMF to allow compatibility with a z/OS system running the ICMF facility.

To activate GCMF, specify GCMF=ON on a MIMINIT statement in the INIT CA MIM file. If your z/OS systems are using the ICMF facility, then specify both GCMF=ON and ICMF=ON on a MIMINIT statement.

How You Issue Cross-system Commands

Command routing paths, or *linkages*, are created by issuing the CA MIC LINK command. Linkages permit a local command source (for example, a CMS user or service machine) to direct commands to a target system and to receive the response back from that command (solicited message).

To create a linkage, issue a LINK command identifying the local command source, the target systems that can receive a command, the target consoles that issues the command on the target system, and the command authority level associated with the linkage.

How You Import Cross-system Messages

Message routing definitions, or *collection sets*, are created by issuing the CA MIC COLLECT command. Collection sets permit local message destinations (consoles, products, and so on) to receive messages from external systems.

There is no COLLECT command for CA MIC for z/VM; however, when you install CA MIC for z/VM, COLLECT commands on z/OS systems can solicit messages that originate on z/VM systems.

How You Customize Command and Message Traffic

CA MIC lets you customize cross-system command traffic in the following ways:

- Define the scope (target systems) for a specific command source.
- Define the type of commands permitted, based on the command authority level associated with the command source.
- Use an optional GCMF exit routine (GCMCMDXT or GCMRCVXT) to provide site-specific control of cross-system command processing.
- Define the size and allocation method used to acquire consoles for the target console pool.
- Define the use of the command and response token (CART) field for commands.

How You Customize Cross-system Message Traffic

CA MIC lets you customize message traffic as follows:

- Edit the job ID field, SYSID field, or both of messages so that system personnel can easily identify from which system a cross-system message originated
- Highlight designated messages
- Classify action type descriptor codes used to collect action type messages
- Define the scope (target systems) to which broadcast messages are propagated
- Define the number of minutes and scope (target systems) for automatically deleting (using the DOM command) highlighted messages
- Use an optional GCMF exit routine (GCMDELXT or GCMSRCXT) to impose site-specific control of cross-system message processing

Chapter 4: Troubleshooting

This section contains the following topics:

Cross-system Command Processing Problems

This section describes solutions to some cross-system command processing problems that you may encounter.

Cross-system Command Responses Not Returned to the Local System

Try these possible solutions:

- If CA MIC is using extended MCS (EMCS) consoles, then verify that the product on the target system supports EMCS consoles. For example, earlier releases of CICS, JES2, and CA Look may not support this.

  Change GCMINIT EXTCON=NONE. This forces CA MIC to use subsystem consoles. Try the command again.

- Verify that CA MIC is synchronized on all systems. For example:

  ```
  F MIM,D SYS
  ```

  - Look in the syslog on the target system. Is the command reissued on the target system?
  - Look for any CA MIC error messages on the local or target system, for example, messages MIM3167, MIM3169, and MIM3066.

    If the MIM3066 message was issued, then this indicates that a target console was not available. You may have run out of available target consoles. Check GCMINIT CONSLIST and SETOPT MAXCONS and MINCONS values. Do a D C,L or D C,SS if you are using subsystem consoles. Check to see that the consoles specified in CONSLIST are available to CA MIC. If MAXCONS or MINCONS is used, then check to see if any subsystem consoles are available. Also, look for error messages indicating that there may be problems allocating consoles.

- Verify the source of the cross-system commands. Some sources do not receive responses to cross-system commands. For example, batch jobs and internal sources do not “receive” responses to cross-system commands. In these cases, the command executes on the target system and the response is returned to the original systembut not to the issuing source. In most cases, the response returns to the log of the originating system.
- Check for CA MIC exits.

  Issue the following command on all systems to verify whether any CA MIC exits are active:

  ```
  F MIM,D EXIT
  ```

Command Rejected-No Linkage Defined (Message MIM3039)

Try these possible solutions:

- You may need to add linkages for INTERNAL and INSTREAM. Enter the z/OS command D C,* from the problem command source. If z/OS message NO CONSOLES MEET SPECIFIED CRITERIA appears, then add linkages for INTERNAL or INSTREAM or both.

  For example:

  ```
  F MIM,LINK INSTREAM SYSID=EXT AUTH=ALL
  F MIM,LINK INTERNAL SYSID=EXT AUTH=ALL
  ```

  LINK INSTREAM may be required if a command is executed in a batch job. LINK INTERNAL may be required for some versions of SDSF.

- To issue cross-system commands from supported releases of SDSF or Netview, customers need to define LINKs for EMCS consoles. The EMCS consoles used by SDSF are named using the TSO user ID of the SDSF user issuing the command. The EMCS consoles used by Netview are named using the Netview logon ID. To determine the EMCS console name for your individual session, issue the z/OS D C,* command from the session. The EMCS console attributes for your session will be returned to you.

  You can reference console names generically through the CONSOLE= parameter on the LINK command. The CONSOLE= parameter also covers both active, and extended, MCS consoles. Through the use of wildcards in console names, you can define a single LINK command to provide multiple users cross-system command capability.

Note: Customers are not required to define a LINK for every TSO user issuing cross-system commands from SDSF. You can issue one LINK command using wildcarding to cover multiple TSO users. The same is true for Netview users and EMCS console users in general.

For example, if all TSO IDs at the site of a user begin with ABC, the user can issue the following LINK command to allow all TSO users to issue cross-system commands from SDSF:

```
LINK CONSOLE=ABC* SYSID=ALL AUTH=...
```

- Verify that the current linkage matches the current CONSOLE from which the command is entered. Review the output of the z/OS D C,* command and compare it to the output of the CA MIM D LINK DETAIL ALL command. Then, enter the appropriate CA MIM LINK command.

  For example:

```
D C,*
IEE889I 10.35.29 CONSOLE DISPLAY 878
MSG: CURR=0   LIM=9999 RPLY:CURR=0    LIM=100  SYS=XE12      PFK=00
 CONSOLE/ALT      ID  --------------- SPECIFICATIONS ---------------
 XE07900          20  COND=M      AUTH=MASTER       NBUF=N/A  UD=Y
  0900               AREA=Z          MFORM=T,J
  XE07               DEL=R    RTME=1      RNUM=39   SEG=39    CON=N
                     USE=FC   LEVEL=ALL             PFKTAB=PFKMIC
                     ROUTCDE=ALL
                     CMDSYS=XE07
                     MSCOPE=XE07
                     MONITOR=JOBNAMES,SESS
                     ALTGRP=MSTRGRPX
```

  This console requires the following LINK:

```
F MIM,LINK CONSOLE=XE07900
```

- Verify that linkage exists for the target system. For example:

```
LINK CONSOLE=XE07900 SYSID=SYSA,SYSB AUTH=SOURCE
```

  This linkage only allows commands to be sent from console XE07900 to SYSA or SYSB. This linkage will not allow a cross-system command to be sent to SYSC.

Unauthorized Use of Control Command-SYS Authority Required (Message MIM0093)

This error message is displayed when a CA MIM command is issued from MIMTSO that requires SYS (system control) authority. By default, a MIMTSO user only has INFO (information) authority. You need to have INFO authority to issue display-type CA MIM commands such as DISPLAY, EDITEST, HELP, and so on. All other CA MIM commands require SYS authority. TSO users generally do not have SYS authority.

There are two possible solutions:

- Use the MIMCMDXT to increase the authority level of the TSO user. This allows a MIMTSO user to enter CA MIM commands that require SYS authority.

  For more details on this exit, see the CA MIM *Programming Guide*.

  Use the CA MIC LINK command to increase the authority level of the TSO user. For example:

  ```
  LINK TSOUSER=KERGL01 SYSID=ALL AUTHORITY=ALL
  ```

  Then the MIMTSO session will have authority to enter all CA MIM commands, including ones that require SYS authority. You must prefix the CA MIM command with the CA MIM command character and the system alias. For example:

  @S1 @SETOPTION MIM INTERVAL=0.1

  @ is the CA MIM command character, and S1 is the CA MIM alias for the target system, as identified on the DEFSYS statement in the MIMINIT member.

  Note: For more information on routing commands to external or local systems, see the chapter “Advanced Topics.”

Command Rejected-Authority Invalid (Message IEE345I)

If this message is displayed on a target system after entering a CA MIC cross-system command, then you may need to increase the authority levels associated with the command source.

To increases authority levels

1. Find out the authority levels associated with the LINK command for the console or extended console from which the command was entered on the system of origin (not the target system). For example:

   ```
   F MIMCONS,D LINK
   ```

2. Find the AUTH value in the MIM3016 message display.

   This value represents the current command authority level for the console or extended console when a cross-system command is entered. Possible values include:

   ALL

   The target console can execute console, I/O, and system control command.

   C

   The target console can execute any cross-system console control command.

   I

   The target console can execute any cross-system I/O control command.

   INFO

   The target console can execute any cross-system informational command.

   MSTR

   The target console can execute any cross-system command requiring master console authority or any other authority level.

   NONE

   The target console cannot execute cross-system commands.

   S

   The target console can execute any cross-system system control command.

   SRCE

   The target console uses the authority level assigned to the command source when executing cross-system commands.

3. Modify the current authority level associated with the console by using the LINK REPLACE command. For example:

   ```
   F MIMCONS,LINK REPLACE CONSOLE=TESTCONS SYSID=ALL AUTH=ALL
   ```

4. Re-enter the cross-system command

   The IEE345I message should not be displayed.

For detailed information on the LINK command and command authority levels, see the chapter “Advanced Topics.”

Before You Call Technical Support

If none of the above resources fix your problem, then CA suggests that you prepare the following information before contacting CA MIM Technical Support:

- Are the commands with which you are experiencing trouble local commands (issued locally) or cross-system commands (issued to another system)?
- Is it *all* cross-system commands that you are having trouble with or certain commands in particular?
- Is it commands from *all* sources, or *one* particular source?
- Which systems in the complex are experiencing the problem?
- What commands, specifically, are involved?
- What are the LINKs on the systems involved?

Cross-system Message Processing Problems

This section discusses cross-system message processing problems.

Message Traffic from External Systems not Displayed on the Local Console

Try these possible solutions:

- Enter D C,* and F MIM,D COLLECT from the local collecting console. Compare the attributes of the local console with the information in the D COLLECT output.
- Verify that console names/IDs match and the correct target systems are specified. Verify that no exclusion criterion is specified.

  Note: Review D GCMF OPT=ALL on the WTO originating system and the local D COLLECT output.
- Verify that CA MIC is synchronized on all systems (F MIM,D SYSTEMS).

CA MIC reissues imported unsolicited WTOs from external systems as GCM/xxxxx on the local system, instead of the typical message ID (for example, IEF233A is changed to GCM/xxxxx). This is done so CA MIC can differentiate between one of its own imported messages versus a WTO that originated on the local system. CA MIC then converts it back to the original message text.

WARNINGS!

- Do not allow MPF to suppress messages that begin with GCM/. If so, then these messages will never appear on the target consoles. Also, do not allow automation packages to act on these messages because unpredictable results may occur.
- Check for CA MIC exits, for example, F MIM,D EXIT. Check the message on the syslog of the originating system. If the LOGONLY bit (00000200) is set, then CA MIC will not route this message (unless SET GCMF CAPTURELOG=YES is specified).

The following are some more possible solutions:

- Check the route codes on the message. Be sure they match the route codes that you are using on the COLLECT command.
- Check the suppression bits (00000004,00000001,00000005). If the message is suppressed on the originating system, then CA MIC will not route this message to external systems.
- If the message is issued as a command response (NR, MR in the syslog), then CA MIC will not route this message.

Receiving Multiple Copies of the Same Message

Try these possible solutions:

- After implementing SYSPLEX, you may see duplicate messages from external systems on local consoles. To prevent this, change the MSCOPE parameter to local-system-ID or *.

  For example, if D C,* shows that MSCOPE=*ALL, and this console is receiving duplicate messages, then enter the following z/OS command:

  ```
  VARY console,MSCOPE=*
  ```

- Users of CA Remote Console who use CA MIC to collect message from external systems to local CA Remote Console sessions, should specify SYSPLEX=NO on the RCSINIT statement to prevent duplicate messages from being displayed at the local RCS session.

Before You Call Technical Support

Before you call CA MIM Technical Support, we suggest that you prepare the following information:

- Do the problem messages originate on the local system or an external system?
- Which systems in the MIMplex are experiencing the problem?
- What messages, specifically, are involved?
- What are the COLLECTs on the systems involved? For example:

  ```
  F MIM,D COLLECT DETAIL ALL.
  ```

ICMF Environment Diagnostics

If command responses are not returned to the local system, or if unsolicited WTO traffic from external systems does not appear on local consoles, then you might be having ICMF communication problems.

To diagnose ICMF communication problems

1. Verify that you are using ICMF by entering the following command:

   ```
   F MIM,D FA
   ```

2. Verify that all current maintenance is applied to both CA MIC and CA L-Serv.

3. Issue the following commands:

   ```
   F MIM,D SYS
   F MIM,ICMF STATUS
   F L-SERV,D APPL
   F L-SERV,D ROUTE
   ```

   Information displayed by these commands is required to diagnose ICMF problems.

   These commands should be entered on all systems where the problem is occurring; that is, no command response due to inactive routes, or CA MIM systems showing INACTIVE.

When running CA L-Serv and the CA MIC ICMF facility, we recommend that you set the CA L-Serv parameters SENDLIMIT and HOLDBUF to 999. This provides adequate buffering for the amount of cross-system traffic generated by the ICMF facility. This prevents CA L-Serv from putting the ICMF facility in HOLD status too often.

Activating Tracing

You can activate event tracing for the GCMF and ICMF facilities using the following commands. To activate tracing, you must first issue the SETOPTION MIM TRACE command.

Note: For more information about the SETOPTION GCMF and SETOPTION ICMF commands, see the *CA MIM Statement and Command Reference Guide*.

How You Activate the TRACE Feature

Use the command SETOPTION MIM TRACE=*options* to activate and deactivate the TRACE feature. The command also lets you deallocate the existing MIM trace data set and allocate a new SYSOUT data set dynamically, and lets you limit some types of tracing by job name. By default, tracing is off.

How You Set Tracing Options

Use the command SETOPTION *facility* SETTRACE=(*options*) to activate the tracing function for the specified facility and selected options. The SETTRACE parameter enables the recording of specific program events in the CA MIM internal trace table.

Each of these commands can specify one or more individual events to be traced. For example, issuing the following command would set the three events named in the parenthesis to be traced:

```
SETOPTION GCMF SETTRACE=(DOMS,COMMANDS,WTOS)
```

Each SETOPTION SETTRACE command controls the tracing of events for a single facility, such as ECMF, MIM, GTAF, and so on. You can combine settings for a facility on a single command, but a new command must be entered for events in different facilities. The effect of all the SETOPTION SETTRACE commands you issue is cumulative, even if you issue multiple commands for the same facility.

How You Reverse Tracing Settings

Use the SETOPTION *facility* RESETTRACE command to reverse or turn off settings that were previously made using SETTRACE. After RESETTRACE is issued, the options you specify are no longer recorded in the CA MIM internal trace table.

How You Activate Printing

Use the SETOPTION *facility* SETPRINT=(*options*) command to activate the printing function for the specified facility and selected options.

The SETPRINT parameter controls whether events that are recorded in the CA MIM internal trace table are also sent to the MIM trace data set. SETPRINT is dependent upon SETTRACE, because only operands that have been specified in both SETTRACE and SETPRINT are sent to the MIM trace data set.

How You Reverse Printing Settings

Use the SETOPTION *facility* RESETPRINT command to reverse or turn off settings that were previously made using SETPRINT. After RESETPRINT is issued, the options you specify are no longer recorded to the MIM trace data set, but continue to be stored in the CA MIM internal trace table, if RESETTRACE has not been issued.

Obtaining Dumps

Dumps are used for diagnostic purposes, and should be used only by direction by CA Technical Support. Use the following commands to obtain dumps for CA MIC-related problems:

- DUMP GCMF
- DUMP ICMF
- SYSDUMP

Note: For more information, see the *Statement and Command Reference Guide*.

Contacting Technical Support

Procedures for gathering and sending diagnostic data and contacting CA Technical Support-whether by the Internet, telephone, fax, or FTP server-are included in the chapter “Troubleshooting” in the CA MIM *Programming Guide*.

Chapter 5: User Exits

This section contains the following topics:

Managing User Exits

CA MIC provides exit routines that you can use to customize its processing. The following routines are available only when you run CA MIC:

GCMCMDXT

Prevents certain local commands from being directed to external systems.

GCMDELXT

Changes the way CA MIC handles DOM (Delete Output Messages) orders.

GCMDSTXT

Prevents all local destinations from receiving collected messages or responses to cross-system commands based on routing data associated with the messages. This exit routine can also be used to alter the text and/or routing data of an imported message.

GCMRCVXT

Prevents certain commands from executing on the local system as a result of a cross-system command issued from an external system.

GCMSRCXT

Prevents certain local messages from being directed to external systems based on routing data associated with the messages.

How You Activate Exit Routines

To activate any exit routine, specify the SETOPTION EXIT command.

For example, to activate the GCMDSTXT, you would issue the command:

```
SETOPTION EXIT=GCMDSTXT
```

If you use a different module name for the exit routine, then specify the same command with the different module name indicated on the LOAD parameter.

For example, if the module name for the GCMSRCXT routine were PRLOGIC, you would specify the command:

```
SETOPTION EXIT=(GCMSRCXT, LOAD=PRLOGIC)
```

Note: For more information, see the *CA MIM Statement and Command Reference Guide*.

How You Display Exit Information

To see information about exit routines you are using, issue a DISPLAY EXIT command. CA MIM displays each logical exit routine name with its load module, address, status, and values set for protection and dump generation. The next screen shows the display for exit routine information:

```
F MIMGR,DISPLAY EXIT

MIM0264 MIM EXIT DISPLAY
EXIT        MODULE   ADDRESS   STATUS  PROT  DUMP  DISA
GCMCMDXT    USR      00006910  ACTIVE  YES   YES    NO
GCMCMDXT    USREX    00006920  INACT   YES   YES    NO
GCMSRCXT    USREXIT  00006930  INACT    NO   YES    NO
TPCRECXT    USREXIT  00006940  ACTIVE   NO   NO     YES
```

You can use the DISPLAY EXIT command to check on the status of any exit routine except MIMINIXT. Use the DISPLAY INIT command to find out whether the MIMINIT INITEXIT statement has been set.

Common Exit Interface: MIMINIXT

CA MIM provides a common exit interface that you can use to dynamically manage any CA MIM exit routine.

Note: For more information, see the *CA MIM Programming Guide*.

Coding Rules

Specific coding rules for the entry environment and register usage and a list of return codes for each routine are provided on the following pages. Follow these general rules when coding a CA MIC exit routine:

- Note that all CA MIC routines are invoked through the standard parameter list pointed to by Register 1.
- Preserve all registers except 0, 1, and 15.
- Do not use the z/OS OPEN, CLOSE, LOAD, LINK, BLDL, or WTO facility or any other z/OS facility that implicitly or explicitly calls the WAIT supervisor.
- Do not use the STIMER or TTIMER macros.
- Provide a save area of at least 72 full words if you want to issue messages from the GCMCMDXT exit routine. If you do this, then use the subroutine provided and *do not* change register 11.
- Make sure that the GCMCMDXT routine uses the appropriate command authority levels in each situation. By default, TSO users are assigned a command authority level of X'0000'. The command authority level for an MCS console is established in its MCS console control block. If you change the command authority level for a TSO user or a console through a LINK command, then CA MIC uses the authority level assigned through the LINK command.

Assemble and link-edit each completed routine as an authorized load module into the authorized load library for CA MIM. The sample member named ASMEXIT in the installed CAI.CBTDJCL data set contains JCL that you can use to assemble and link-edit an exit routine.

The CA MIC exit parameter lists support four-byte console IDs, console names, and EMCS consoles. In addition, mapping macros are provided in the installed CAI.CBTDJCL data set.

The GCMRCVXT and GCMDSTXT exit parameter lists have values that rely on external systems to pass these values to the local system.

You can change some of the routing data associated with a message through the GCMSRCXT and GCMDSTXT routines. Make note of the following information if you modify this information:

- Improper modification of the MCS flags or console ID fields can cause the message to be rejected.
- Some of the routing data associated with a message, like the message ID, may be duplicated in the message text. If you want this information to be consistent, then change both the routing data and the message text.

Sample Exit Routines

The CAI.CBTDSAMP data set contains samples of the CA MIC exit routines. You can use these sample routines, or you can create your own routines. Mapping macros for the exit-specific parameter lists are in the installed CAI.CBTDMAC data set.

EXIT Routine Programming Considerations

Use the guidelines in this section when coding your exit routines using the samples.

GCMCMDXT Exit

This exit routine lets you prevent certain local commands from being directed to external systems. You cannot use the GCMCMDXT exit routine to change the text of the commands. You must use the MIMCMDXT exit routine to change the text of the commands. However, the GCMCMDXT exit routine allows you to change the list of systems to which a command is distributed, or suppress the command entirely.

You can also use this exit routine to code a list of users authorized to issue commands from MIMTSO.

Entry Environment:

Supervisor state, problem program key, or AMODE 31.
You must use this environment at entry and restore it upon return.

Register Usage at Entry:

Common Parameter List (mapped by UXPARM macro in CAI.CBTDMAC)

R1

Contains the common exit parameter list, in the following format:

+0

Contains the address of the GCMCMDXT parameter list.

+4

Contains the address communication word of the GCMCMDXT routine. During its initialization call, the routine sets the contents of the word. The contents are passed unchanged for all subsequent calls to this routine.

+8

Contains the full word of flags used to pass status information to the routine.

+C

Contains the global communication word.

R9

This register is reserved.

R11

This register is reserved.

R13

Contains the address of the standard save area.

R14

Contains the return address for CA MIM.

R15

Contains the entry point address for the GCMCMDXT routine.

Note: All other registers are undefined.

GCMCMDXT Exit-specific Parameter List (mapped by MIMCMDXP macro in CAI.CBTDMAC)

+0

Fullword containing the address of the sixteen-byte bit-mask of systems targeted for this cross-system command. The bits, from high order to low order, represent the system indexes of the targeted systems.

+4

Fullword containing the address of the command text.

+8

Halfword containing the length of the command text.

+A

(Two bytes reserved for user exit compatibility)

+C

Fullword containing the address of the CA MIM message routine that you can use to send a message to the cross-system command issuer. If you want to use this routine, then make sure your exit provides a save area of at least 72 fullwords and that registers 9 and 11 are the same as on entry to your exit when you branch to the message routine.

+10

Fullword containing the address of the MCS authority flag bytes.

+14

(Twelve bytes reserved)

+20

Fullword containing the 4-byte console ID.

+24

One-byte unsigned integer containing the command source type defined as follows:

0-Real MCS console
4-Extended MCS (EMCS) console with migration ID
8-Extended MCS (EMCS) console
12-Subsystem console
16-Product (N/A)
20-TSO user
24-INSTREAM (JCL converter)
28-INTERNAL (no console)

+25

(Three bytes reserved)

+28

Eight-byte command source name. Can be console name, TSO user ID, INSTREAM, or INTERNAL depending on command source type.

+30

Eight-byte Command and Response Token (CART).

+38

(Sixteen bytes reserved)

+48

Eight-byte GCMCMDXT user exit parameter list eye catcher “GCMCMDXT.”

Return Code, Register 15:

0

This code indicates that this command should be processed as usual.

4

This code indicates that CA MIC should reject this command and issue message MIM3029 to notify the issuing console or TSO user that the command has been rejected.

8

This code indicates that CA MIC should reject this command but should not issue a message about the rejection.

Note: Upon return, you must restore all registers except 0, 1, and 15.

How You Issue Messages from the GCMCMDXT Exit

If you need to send a message from the GCMCMDXT routine to the console or TSO user that issued a command, then use the message subroutine provided. The following illustration shows you mandatory code for the message subroutine:

```
GCMCMDXT SAVE  (14,12)

*
         LR    R12,R15            R12 -> Base reg
         USING GCMCMDXT,R12
*
         LA    R2,SAVEAREA        R2 -> Our Save Area
         ST    R13,4(,R2)         Save -> H.S.A.
         ST    R2,8(,R13)         Save -> O.S.A. in H.S.A
         LR    R13,R2             R13 -> Our Save Area
*
         LR    R2,R1              R2 -> Common exit parm list
         USING UXPARM,R2

         ...

         LA    R1,LFORMWTO
         SLR   R0,R0               Clears register
         L     R0,CMDXSID          Loads command issuer
         L     R15,CMDXMSG@        Points to the message
         BASR  R14,R15             Calls the message subroutine

         ...

SAVEAREA DS    72F                Big Save Area so we can call
                                  ..the message routine
                                  ..pointed to by CMDXMSG@.
*
LFORMWTO WTO   'Any User Defined Text',
               MCSFLAG=(RESP),CONSID=,MF=L
```

CA MIC includes these messages in the MIM trace data set if the TRACE feature is active.

GCMDELXT Exit

This exit routine lets you change the way CA MIC selects action messages to be deleted. By using this exit routine, you can override the SETOPTION DELETEINTERVAL value for specific messages, causing them to be deleted either earlier or later than they would have been. The current value for the SETOPTION AUTODELETE controls which messages can be deleted by this exit routine.

Note: Since CA MIC does not automatically delete WTOR messages; this exit routine has no effect on them.

CA MIC typically deletes action messages according to a time period set on the SETOPTION DELETEINTERVAL command. By using the GCMDELXT exit routine, you can:

- Delete highlighted messages before they become obsolete
- Unconditionally retain critical action messages (such as tape mount messages) that should not be deleted automatically

CA MIC invokes the GCMDELXT routine approximately once a minute, which causes this exit routine to delete messages in two minutes of when a message is issued. If CA MIC invokes the routine after a message has been issued, then that message is deleted almost immediately.

You can use the GCMDELXT routine to examine (but not modify) some of the relevant data associated with a highlighted message. You can use the user data area provided in the parameter list to store information about a message between each invocation of this routine.

Entry Environment:

Supervisor state, problem program key, or AMODE 31.
You must use this environment at entry and restore it upon return.

Register Usage at Entry:

Common Parameter List (mapped by UXPARM macro in CAI.CBTDMAC)

R1

Contains the parameter list, in the following format:

+0

Contains the address of the parameter list of the GCMDELXT routine.

+4

Contains the address of the communication word of the GCMDELXT routine. During its initialization call, the routine sets the contents of the word. The contents are passed unchanged for all subsequent calls.

+8

Contains the full word of flags used to pass status information to the routine.

+C

Contains the global communication word.

R13

Contains the address of the standard save area.

R14

Contains the CA MIM return address.

R15

Contains the GCMDELXT entry point address.

Note: All other registers are undefined.

GCMDELXT Exit-specific Parameter List (mapped by GCMDELXP macro in CAI.CBTDMAC)

+0

(Fullword reserved for pre-Version 4.1 and above user exit compatibility)

+4

Fullword containing the address of the four-byte user data area for this message. The contents of the user data area are retained for the life of the action message.

+8

(Eight bytes reserved)

+10

Eight-byte job name of message issuer.

+18

(Eight bytes reserved for pre-Version 4.1 and above user exit compatibility)

+20

Eight-byte time stamp of message.

+28

(One byte reserved)

+29

Three-byte DOM ID of message. Provided for message identification purposes only. Issuing the DOM SVC from in the GCMDELXT exit causes unpredictable results. The GCMDELXT can issue a return code to instruct CA MIC to delete the message.

+2C

(Fullword reserved)

+30

Twelve-byte message name of the action message (that is, IEF233A), left justified and padded on the right with blanks.

+3C

(Twelve bytes reserved)

+48

Eight-byte GCMDELXT user exit parameter list eye catcher “GCMDELXT.”

Return Code, Register 15:

0

Indicates that CA MIC should process this message as usual.

4

Indicates that CA MIC should not delete this message now. However, CA MIC should consider the message for deletion again the next time this routine encounters the message.

8

Indicates that CA MIC should delete this message immediately.

12

Indicates that CA MIC should retain this message.

Note: Upon return, you must restore all registers except 0, 1, and 15.

GCMDSTXT Exit

This exit routine lets you prevent destinations on the local system from receiving collected messages, or responses to cross-system commands based on routing data associated with the message.

You also can use the GCMDSTXT exit routine to change the text of an imported message and the following routing data associated with that message:

- Name of the issuing job
- Message ID
- Standard MCS flags
- Descriptor codes
- Routing codes
- Extended Request flags

Entry Environment:

Supervisor state, problem program key, or AMODE 31.
You must use this environment at entry and restore it upon return.

Register Usage at Entry:

Common Parameter List (mapped by UXPARM macro in CAI.CBTDMAC)

R1

Contains the parameter list, in the following format:

+0

Contains the address of the parameter list of the GCMDSTXT routine.

+4

Contains the address of the communication word of the GCMDSTXT routine. During its initialization call, the routine sets the contents of the word. The contents are passed unchanged for all subsequent calls to this routine.

+8

Contains the full word of flags used to pass status information to the routine.

+C

Contains the global communication word.

R13

Contains the address of the standard save area.

R14

Contains the return address for CA MIM.

R15

Contains the entry point address for the GCMDSTXT routine. All other registers are undefined.

GCMDSTXT Exit-specific Parameter List (mapped by GCMWTOXP macro in CAI.CBTDMAC)

+0

(Fullword reserved for pre-Version 4.1 and above user exit compatibility)

+4

Fullword containing the address of the message text. Modifiable by the user exit.

+8

One-byte unsigned integer containing the length of the message text. The GCMDSTXT cannot change the length of the message text. If you remove text, then replace it with blanks.

+9

(Seven bytes reserved)

+10

Eight-byte job name or CMS user ID of message issuer.

+18

Twelve-byte message name (that is, IEF233A), left justified, padded on right with blanks.

+24

Two bytes containing the z/OS MCS flags for this message.

+26

Two bytes containing the z/OS descriptor codes for this message.

+28

Sixteen bytes containing the z/OS route codes for this message.

+38

(One byte reserved)

+39

(One byte reserved)

+3A

One byte containing the following flag bits:

B'10000000'-WTO imported from a z/OS system
B'01000000'-Message imported from a z/VM system
B'00111111'-(reserved)

+3B

(Five bytes reserved)

+40

Four-byte message ID of this message on the originating system. This is provided for message identification and tracking only. Do not issue the DOM SVC in the GCMDSTXT exit, as it causes unpredictable results. The GCMDSTXT can issue a return code to instruct CA MIC to DOM (delete) the message.

+44

Eight-byte system name of the originating system, if available. Otherwise, this field contains X'00's.

+4C

Eight-byte sysplex name of which the originating system is a member, if applicable. Otherwise, this field contains X'00's.

+54

One byte containing the following flag bits:

B'10000000'-four-byte console ID present at offset +X'58'
B'01111111'-(reserved)

+55

(3 bytes reserved)

Note: The fields at offsets +X'58' and +X'5C' refer to the destination console, if any, on the system where the message originated; most likely an external system. This may be defined as another console on this system, or it may be undefined. If, however, both this system and the originating system are members of the same sysplex, then these fields refer to the same console on both systems.

+58

Fullword containing the four-byte console ID.

+5C

Eight-bytes containing the console name to which this message was destined on the originating system, if message was console directed. Otherwise, this field contains X'00's.

+64

Eight-byte CART. X'00's if unavailable or not specified by the command issuer.

Note: The fields at offsets +X'6C', +X'70', and +X'78' refer to the cross-system command source, if any. These fields are non-zero only if this message is a response to a cross-system command issued from this system.

+6C

Fullword containing the four-byte console ID.

+70

Eight-byte command source name. Can be console name, TSO user ID, or INSTREAM depending on command source type.

+78

One-byte unsigned integer containing the command source type defined as:

0 Real MCS console
4 Extended MCS (EMCS) console with migration ID
8 Extended MCS (EMCS) console
12 Subsystem console
16 Product (N/A)
20 TSO user
24 INSTREAM (JCL converter)
28 INTERNAL (no console)

+79

(Seven bytes reserved)

+80

Four bytes containing Extended Request Flags for message (see GCMWTOXP mapping macro).

+84

Four bytes containing the Extended Message Modification Bytes for requesting to change message color, highlighting, and intensity (see GCMWTOXP mapping macro).

+88

(Eight bytes reserved)

+90

Eight-byte eye catcher containing GCMWTOXP.

+98

Eight-byte eye catcher containing GCMDSTXT.

Return Code, Register 15:

0

Indicates that CA MIC should accept this message (with any modifications you have made to it) and process it as usual.

4

Indicates that GCMF should reject this message.

Note: Upon return, you must restore all registers except 0, 1, and 15.

GCMRCVXT Exit

CA MIC uses the GCMRCVXT exit routine whenever a cross-system command is received on the local system. This exit routine lets you prevent certain commands from executing on the local system as a result of a cross-system command issued from an external system.

You cannot use the GCMRCVXT exit routine to change the text of commands or the information associated with a command. You must use the CA MIM MIMCMDXT exit routine to change the text of commands.

Entry Environment:

Supervisor state, problem program key, or AMODE 31.
You must use this environment at entry and restore it upon return.

Register Usage at Entry:

Common Parameter List (mapped by UXPARM macro in CAI.CBTDMAC)

R1

Contains the address of the parameter list, in the following format:

+0

Contains the parameter list of the GCMRCVXT exit routine.

+4

Contains the address of the communication word of the routine. During its initialization call, the routine sets the contents of the word. The contents are passed unchanged for all subsequent calls to this routine.

+8

Contains the full word of flags used to pass status information to the routine.

+C

Contains the global communication word.

R13

Contains the address of the standard save area.

R14

Contains the return address for CA MIM.

R15

Contains the entry point address for the GCMRCVXT routine. All other registers are undefined.

GCMRCVXT Exit-specific Parameter List (mapped by the GCMRCVXP macro in CAI.CBTDMAC)

+0

Fullword containing the address of the sixteen-byte system mask that represents the system from which this cross-system command was issued.

+4

Fullword containing the address of the command text.

+8

One-byte unsigned integer containing the length of the command text.

+9

(One byte reserved for pre-Version 4.1 and above user exit compatibility)

+A

Two bytes containing the MCS command authority of the command source defined as follows:

Byte 1:
B'00000000'-INFO authority
B'10000000'-SYS authority
B'01000000'-IO authority
B'00100000'-CONS authority
B'00000010'-MASTER authority
B'00000001'-(No authority)

Byte 2: (z/VM authority-Not applicable on CA MIC):
B'00000010'-z/VM OPERPRIV authority
B'00000001'-z/VM USERPRIV authority

+C

(Fullword reserved for pre-Version 4.1 and above user exit compatibility)

+10

(Eight bytes reserved)

+18

Fullword containing the four-byte console ID.

+1C

One-byte unsigned integer containing the command source type defined as follows:

0 Real MCS console
4 Extended MCS (EMCS) console with migration ID
8 Extended MCS (EMCS) console
12 Subsystem console
16 Product (N/A)
20 TSO user
24 INSTREAM (JCL converter)
28 INTERNAL (no console)
255 (Not available)

+1D

(Three bytes reserved)

+20

Eight-byte command source name. Can be console name, TSO user ID, INSTREAM, or INTERNAL depending on command source type.

+28

Eight-byte CART. Binary zeros if not specified or not available.

+30

(Eight bytes reserved)

+38

GCMRCVXT user exit parmlist eye catcher containing GCMRCVXT.

Return Code, Register 15:

0

This code indicates that this command should be processed as usual.

4

This code indicates that CA MIC should reject this command and issue a message to notify the issuing user (on the originating system) that the command was rejected.

Note: Upon return, you must restore all registers except 0, 1, and 15.

GCMSRCXT Exit

This exit routine lets you prevent certain local messages from being directed to external systems, based on routing data associated with the messages.

You also can use the GCMSRCXT exit routine to change the text of a message and the following routing data associated with a message:

- Name of the issuing job
- Message ID
- Standard MCS flags
- Descriptor codes
- Routing codes
- Extended Request flags

Entry Environment:

Supervisor state, problem program key, or AMODE 31.

You must use this environment at entry and restore it upon return.

Register Usage at Entry:

Common Parameter List (mapped by UXPARM macro in CAI.CBTDMAC)

R1

Contains the parameter list, in the following format:

+0

Contains the address of the parameter list of the GCMSRCXT routine.

+4

Contains the address of the communication word of the routine. During its initialization call, the routine sets the contents of the word. The contents are passed unchanged for all subsequent calls to this routine.

+8

Contains the full word of flags used to pass status information to the routine.

+C

Contains the global communication word.

R13

Contains the address of the standard save area.

R14

Contains the return address for CA MIM.

R15

Contains the entry point address for the GCMSRCXT routine. All other registers are undefined.

GCMSRCXT Parameter List (mapped by the GCMWTOXP macro in CAI.CBTDMAC)

+0

(Fullword reserved for pre-Version 4.1 and above user exit compatibility)

+4

Fullword containing the address of the message text. Modifiable by the user exit.

+8

One-byte unsigned integer containing the length of the message text. The GCMSRCXT cannot change the length of the message text. If you remove text, then replace it with blanks.

+9

(Seven bytes reserved)

+10

Eight-byte job name of WTO issuer.

+18

Twelve-byte message name (that is, IEF233A), left justified, padded on right with blanks.

+24

Two bytes containing the z/OS MCS flags for this WTO.

+26

Two bytes containing the z/OS descriptor codes for this WTO.

+28

Sixteen-bytes containing the z/OS route codes for this WTO.

+38

(One byte reserved)

+39

(One byte reserved)

+3A

One-byte field containing the following flag bits:

B'10000000' -

WTO issued on a z/OS system (always on z/OS systems)

B'01000000' -

Message issued on a z/VM system (always off z/OS systems)

B'00111111' -

(reserved)

+3B

(Five bytes reserved)

+40

Four-byte message ID of this message on the originating system. This is provided for message identification and tracking only. Do not issue the DOM SVC in the GCMSRCXT exit as it causes unpredictable results. The GCMSRCXT can issue a return code to instruct CA MIC to DOM (delete) the message.

+44

Eight-byte system name of this system, if available. Otherwise, this field contains X'00's.

+4C

Eight-byte sysplex name of which this system is a member, if any. Otherwise, this field contains X'00's.

+54

One-byte field containing the following flag bits:

B'10000000' - four-byte console ID present at offset +X'58'
B'01111111' - (reserved)

+55

(Three bytes reserved)

+58

Fullword containing the four-byte console ID to which this message is destined, if message is console directed. Otherwise, this field contains X'00's if this message is not console directed, or if bit X'80' is off at offset +X'54' in the parameter list.

+84

Four bytes containing the Extended Message Modification Bytes for requesting to change message color, highlighting, and intensity (see GCMWTOXP mapping macro).

+88

(Eight bytes reserved)

+90

Eight-byte eye catcher containing GCMWTOXP.

+98

Eight-byte eye catcher containing GCMSRCXT.

Return Code, Register 15:

0

Indicates that CA MIC should accept this message (with any modifications you have made to it) and process it as usual.

4

Indicates that CA MIC should reject this message.

Note: Upon return, you must restore all registers except 0, 1, and 15.

Chapter 6: Utilities and Other Interfaces

This section contains the following topics:

TSO Command Processing Utility

The TSO command processing interface is an optional feature you can install. This feature provides TSO users with a command interface to the CA MIM address space. Minimally, it enables TSO users to issue DISPLAY commands from their individual TSO sessions. The command responses are then directed back to the issuing TSO user.

If you are running CA MIC, then the TSO interface feature can be enhanced to execute any z/OS, CA MIM, or z/OS subsystem command. Commands can be routed to any or all systems in the CA MIC complex.

You can optionally code the MIMCMDXT and GCMCMDXT exit routines to prohibit the execution of specific commands by a TSO user.

Note: For more information, including how to install the interface, see the chapter "Utilities" in the *CA MIM Programming Guide*.

Report Generation for CA MIC

CA MIM reports provide you with an effective tool to evaluate the performance of the systems in your complex. The reports use a sampling of statistical data to yield important information about the operating activities in your complex.

Note: This functionality requires the CA Easytrieve Interface. For more CA Common Services requirements information, see the appendix "CCS for z/OS Component Requirements" in the *Installation Guide*.

The Report Generation Process

CA MIM uses a single SMF record for the record collection process. By using record *subtypes*, you can identify the statistical records to be collected for the different reports available under the CA MIM facilities (GDIF, ECMF, TPCF, and so on).

The major steps involved in the report generation process are:

1. Specify the SMF record number on a MIMINIT RECORDTYPE statement in the initialization member. Once this is set, you should not need to change it.
2. Specify the record collection criteria using the SETOPTION command and begin the record collection process.

   Note: You must allow the record collection process to run for a while before dumping the SMF records to data sets.
3. Dump the SMF records to a usable data set through the IBM IFASMFDP utility
4. Run a report or create a report output file.

Important! CA MIM provides a set of reports for general use. If, however, any one of these reports does not meet your specific requirements, then you can write your own customized reports in the language of your choice, using the SMF records supplied by CA MIM. For further information on this topic, see the MIMSTREC member in the CAI.CBTDMAC data set.

Specify an SMF Record Number

Before any statistical records can be collected for use in the CA MIM report process, you need to specify an SMF record number. To do this, you place a MIMINIT RECORDTYPE initialization statement in the initialization member.

For example, if you decide to use the default SMF record number of 189, then you would specify the following statement in the initialization member:

```
MIMINIT RECORDTYPE=189
```

Now you are ready to specify the record collection criteria for the report or reports you want to run.

Specify Record Collection Criteria

To collect statistical records for CA MIM reports, you need to specify the type of statistical records to be collected by issuing the command SETOPTION STATCOLLECT with the specific CA MIM facility (EDIF, GDIF, TPCF, and so on).

Specify the SUBTYPE operand on the SETOPTION STATCOLLECT command to select specific record subtypes for a facility, or specify ALL to collect records for all record subtypes available under that facility. These record subtypes correspond directly to the reports available for each facility. A brief description of the report follows each record subtype.

The record subtype for CA MIC is CN for GCMF. The CA MIC Command Audit report provides an audit trail of cross-system commands issued through GCMF.

The record collection process must be started for each individual CA MIM facility for which you want to run a report. This section shows how to begin the record collection process using the SETOPTION command.

For example, to create the CA MIC Command Audit report, you specify the following SETOPTION command:

```
SETOPTION GCMF STATCOLLECT(SUBTYPE=CN)
```

This command tells CA MIM to collect statistics for the report subtype CN. Next, you must specify the record collection sampling cycle and recording interval for GCMF.

Important! To activate the record collection process for all record subtypes, you must repeat this procedure for each subtype, specifying the SETOPTION STATCOLLECT command and the facility associated with that record subtype.

Set Sampling Time Cycle

The sampling cycle time is how long CA MIM waits between each statistical data sampling. You specify this time, in seconds, with the SETOPTION STATCYCLE command. The default value is 60.

Set Sample Recording Frequency

The STATINTERVAL parameter specifies how often, in minutes, statistical data samples are recorded to SMF data sets. The default value is 15.

Sample Command for Record Collection

The following example shows how to begin the record collection process for the CA MIC Command Audit report. This example indicates data sampling every 30 seconds, and a statistical record recording interval of every hour:

```
SETOPTION GCMF STATCOLLECT(SUBTYPE=CN) STATCYCLE=30, STATINTERVAL=60
```

Note: The STATCYCLE and STATINTERVAL values apply to all record collection subtypes for a given CA MIM facility (GCMF, and so on).

Once the statistical record collection process has begun, records are collected and written to the SMF data sets. You need to use your site-specific procedures for dumping the records to physical sequential data sets that can be used by CA MIM to produce reports.

Notes:

- For more information on the SETOPTION commands used in this section, see the *Statement and Command Reference Guide.*
- For detailed information on using the SMF report generator, see the chapter "Utilities" in the CA MIM *Programming Guide*.

Sample Command Audit Report (CN) Report

```
(1)                (2)                                                                                                  (3)
2010/07/13 14:06   CA MIM Console Command Audit Report                                                          PAGE      1
(4) System:     SYSA              (5) Jobname:  MIMGR       (6) Release:   11.9        (7)Service Level:  0000
(8) From:       2009/10/25 00:00  (9) Each:     RECORD
(10)To:         2010/03/12 10:52  (11)Shift:    00:00 24:00
                          (14)      (15)
       (12)      (13)  Dest/SRC  Source   (16)             (17)
       Time       S/R   System  Console  User ID          Command
 ---------------- ---- -------- -------- -------- ----------------------------------------------------------------------------------------
 2010/03/09 10:40 SENT VMA      00000017 OPSP     Q 219F
 2010/03/09 10:46 SENT VMA      00000017 OPSP     Q 219F
 2010/03/09 10:48 RCVD SYSB     0200004C OPSP     &CON
                  RCVD SYSB     0200004C OPSP     &RES
                  SENT VMA      00000017 OPSP     Q 2862
 2010/03/09 10:49 SENT VMA      00000017 OPSP     Q 2862
```

Field	Description
(1)	Date and time the report was created
(2)	Report title
(3)	Report page number

Field	Description
(4)	System name of the CA MIC image as defined by the DEFSYS statement
(5)	Name of the CA MIC started task
(6)	The CA MIC release number
(7)	The CA MIC service level
(8)	Date and timestamp of the earliest record in the reporting interval.
(9)	Summary level for the data line. In this example, each RECORD is displayed in the detail line. If each HOUR was specified, then the detail line would be a summary of the hour's activity.
(10)	Date and timestamp of the latest record in the recording interval.
(11)	Range of records included in the report.
Time (12)	Time of day that the command event occurred.
S/R (13)	Indicates whether command text is being imported from an external system (RCVD) or transported to an external system (SENT)
Dest/SRC System (14)	System name of the CA MIC image, as defined by the DEFSYS statement, from which the command text originated (if RCVD in preceding column) or target system to which the command text is being transported (SENT in preceding column)
Source Console (15)	Four-byte console number, in hexadecimal, from which the command text originated
User ID (16)	User identification or product name, if any, of entity that issued the command text
Command (17)	Command text that was sent or received

Index

L

M

O

P

R

S

T

V