CA OPS/MVS® Event Management and Automation

Integration Guide

Release 12.1

CA Technologies Product References

This document references the following CA Technologies products:

- CA 7® Workload Automation (CA 7 WA)
- CA Jobtrac™ Job Management (CA Jobtrac)
- CA MIM™ Resource Sharing (CA MIM)
- CA NSM System Status Manager CA OPS/MVS® Option (CA NSM SSM CA OPS/MVS Option)
- CA OPS/MVS® Event Management and Automation (CA OPS/MVS)
- CA PDSMAN® PDS Library Management (CA PDSMAN)
- CA Scheduler® Job Management (CA Scheduler)
- CA SYSVIEW® Performance Management (CA SYSVIEW)
- CA VM:Operator™ (CA VM:Operator)

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 Install the NetView Interface (see page 59) section.

Contents

Chapter 1: Introduction

This section contains the following topics:

Why Integrate

Through integration with other products, CA OPS/MVS provides you with a unified automation environment for your enterprise. For example, integration with CA NSM enables you to correlate events between mainframe and distributed systems, and automate your responses accordingly. On the mainframe, integration with other products enables you to automate responses to events detected by those products, (for example, performance problems detected by CA SYSVIEW).

CA OPS/MVS offers a broad set of integrated components for:

- Managing the health and availability of each aspect of your computing infrastructure
- Assessing and managing the cost of these components in the enterprise
- Managing the service to end users, ultimately sharing the costs of IT across the entities that use them

Integration with other products provides more ways to collect data to help automation applications make informed decisions. It also provides a more comprehensive set of actions for automation applications to take. The more actions that can be taken, the better a problem can be resolved. For example, not all CICS messages are issued to the z/OS console. Some are issued to the CICS transient data queue. Also, some actions related to CICS cannot be issued as operator commands. Instead, they must be issued as commands to CICS. To address these needs, the CICS Operational Facility (COF) is offered as a separate component for integration with CA OPS/MVS. It provides the level of integration necessary to solve CICS related problems.

As another example of the need for integration, CA OPS/MVS can react to events external to the job scheduling process, thereby initiating actions that control job scheduling activities.

Event Management

CA OPS/MVS reacts to events from various sources and automates responses. Facilities are available to provide enhanced management for events from the following sources through integration:

- CA Automation Point through the CA Common Services (CCS) for z/OS CAICCI service and Multi-System Facility (MSF) to provide outboard automation and automated notification, which includes paging, email, text-to-speech, voice notification, solicitation of input through Dual Tone Multi-Frequency (DTMF) tones, pre-recorded messages, and message forwarding
- CA NSM through the CA OPS/MVS UNIX System Services (USS) interface, CCS Event Management component, and CAICCI service to provide unified event management for your mainframe and distributed systems
- CA SYSVIEW through the CA OPS/MVS generic event application program interface (API) to provide automated responses to performance related problems
- CA MIM through the CA OPS/MVS generic event API and CA MIC to provide automated responses to named events and cross-system event management
- CA PDSMAN through the CA OPS/MVS generic event API to provide automated responses to partitioned resource situations
- CA 7 WA through the CCS CA Global Subsystem (GSS) and CAIENF services to enable the issuing of product-specific commands to perform CA 7 WA functions in response to messages
- CA Jobtrac and CA Scheduler through the CA GSS service to enable the issuing of product-specific commands to perform CA Jobtrac and CA Scheduler functions in response to messages
- CA NetMaster products to provide unified alert monitoring
- z/VM systems through a client/server application to enable automated responses to z/VM messages, and to monitor and manage Linux for zSeries guests
- Customer Information Control System (CICS) through the CICS Operations Facility (COF) to provide automated responses to CICS transient data queue (TDQ) messages
- Information Management System (IMS) through the IMS Operations Facility (IOF) to provide automated responses to IMS messages
- Tivoli OMEGAMON XE by writing exception messages to CA OPS/MVS through the SUBSYS DD statement to provide automated responses to those messages
- Tivoli NetView through the NetView Interface and the NetView Operator Facility (NOF) to unify network and system automation

Resource Management

CA OPS/MVS System State Manager (SSM) automates and controls the management of system resources such as started tasks, subsystems, JES initiators, and VTAM nodes. Integration with CA NSM through the CA NSM SSM CA OPS/MVS Option enables you to manage the resources from a WorldView map.

Problem Management

CA Service Desk provides service request, incident, problem, and change management that maximizes analyst productivity and enhances responsiveness. Integration enables CA OPS/MVS problems be recorded in CA Service Desk for analysis and action.

Chapter 2: Integrating with CA Automation Point

This section contains the following topics:

Overview and Value

Through MSF, CA OPS/MVS provides bidirectional integration with CA Automation Point. CA Automation Point helps you consolidate events from and automate responses to multiple platforms including mainframes managed by CA OPS/MVS.

Integration between CA OPS/MVS and CA Automation Point lets you asynchronously transmit data and commands between CA OPS/MVS and CA Automation Point. For integration, you need to configure CAICCI and MSF.

The functions you can perform from CA OPS/MVS include the following:

- Trigger automation on CA Automation Point workstation.
- Write items to a queue.
- Execute a REXX EXEC.

The functions you can perform from CA Automation Point include the following:

- Enable outboard automation for the entire z/OS shutdown and initial program load (IPL) process.
- Enable cross-system multi-platform correlation of events.
- Enable remote viewing and control of z/OS operator consoles and Hardware Management Console (HMC).
- Provide problem notification and escalation.

Environments and Rule

The following environments and rule are available to help you communicate with CA Automation Point:

- The ADDRESS AP and the ADDRESS WTO environments enable you to send commands and messages to CA Automation Point.

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

- CA Automation Point provides the ADDRESS OPS environment. The environment enables you to send commands and messages to CA OPS/MVS.

  Note: For more information, see the *CA Automation Point Reference Guide*.

- The sample APNOTIFY)MSG rule in the &hlq.CCLXRULS data set notifies CA Automation Point when IKJ574I (broadcast data set full) messages are not responded to.

Establish Communication

For CA OPS/MVS to communicate with CA Automation Point, you must configure both products.

The following components are required for the establishment of communication between CA OPS/MVS and CA Automation Point:

- CAICCI service
- MSF

Note: This topic provides an overview of how to establish communication between CA OPS/MVS and CA Automation Point. For detailed information, see the *Administration Guide*.

To configure CA OPS/MVS for communication with CA Automation Point

1. Ensure that CAICCI is installed on your system.

   The CAICCI service is available to MSF as a means of communication.

2. Define MSF sessions in your MSFINIT program:

   - Use the following statements to enable MSF to use CAICCI:

     ```
     ADDRESS OPSCTL "MSF DEFINE MSFID(msf_id_local) APPLID(local_vtam_name)"
     ADDRESS OPSCTL "MSF DEFINE MSFID(msf_id_remote) APPLID(ap_caicci_id) CCI"
     ```

   - Use the following statement to define CA Automation Point to MSF:

     ```
     ADDRESS OPSCTL "MSF DEFINE MSFID(ap_host_name) APPLID(ap_caicci_id) AP"
     ```

   These changes take effect when MSF is recycled. If you want to make these changes available to the current MSF instance, use OPSVIEW Option 4.2 to define these sessions to the current instance.

3. Ensure that the following statements are included in your OPSSPA00 program:

   ```
   var = OPSPRM("SET","INITMSF","YES")
   var = OPSPRM("SET","INITCCI","YES")
   var = OPSPRM("SET","APDEFAULTUSERID","user_id")
   var = OPSPRM("SET","OSFSECURITY","CHECKUSERID")
   var = OPSPRM("SET","MSFDELAY","10")
   ```

   These changes take effect when CA OPS/MVS is recycled.

   CA OPS/MVS is configured for communication with CA Automation Point.

To configure CA Automation Point for communication with CA OPS/MVS

1. Access CA Automation Point Configuration Manager, and then navigate to CA OPS/MVS Interface to open the CCI Configuration dialog.

2. Use the following REMOTE statement to define the MSF CAICCI node that CA Automation Point connects to:

   ```
   REMOTE=ops_host_name ops_caicci_id 32768 STARTUP PORT=ops_caicci_port
   ```

3. Open the CA OPS/MVS Event Traffic Configuration dialog, include the defined node, and recycle CA OPS/MVS Interface.

   CA Automation Point is configured for communication with CA OPS/MVS.

Automated Problem Notification

Through CA OPS/MVS Automated Operations Facility (AOF) rules, you can escalate problems on your z/OS systems and use the Notification Manager in CA Automation Point to automate problem notification using various methods (for example, paging). This ensures that someone knows about the problems, irrespective of whether they are on the systems being monitored.

Example: Notification of the Condition that the Broadcast Data Set Is Full

A common problem on z/OS systems is for the broadcast data set to fill up, resulting in an IKJ574I message. You can use the distributed sample &hlq.CCLXRULS(APNOTIFY) rule to notify appropriate staff of the problem:

1. The rule is enabled.
2. The initialization section of the rule sets up some initial values, including the list of timesharing option (TSO) users to notify and the criteria for problem escalation.
3. When an IKJ574I message is detected, the rule uses ADDRESS OPER to send the following message to the TSO users:

   ```
   BROADCAST data set is full
   ```

   It suppresses further IKJ574I messages from the console.
4. If the problem is not acknowledged after it has occurred a predefined number of times, it is escalated to a Notification Manager group in CA Automation Point (as defined in MSFID) (see page 14) using ADDRESS AP.
5. A group member acknowledges the problem and sets the GLOBAL.IKJ574I_ACKNOWLEDGED variable, suppressing further messages from the console. The rule takes no further action until the variable is reset.
6. The problem is corrected, and the variable is reset.

Chapter 3: Integrating with CA NSM

This section contains the following topics:

Overview and Value

Integration between CA OPS/MVS and CA NSM provides the following capabilities:

- Send messages to CA NSM for event consolidation and correlation through the Event Management component of CCS for z/OS and CA NSM. Integration through Event Management lets you see events from CA OPS/MVS in the CA NSM Event Console and use CA NSM to perform event correlation.
- Send information about resources defined to SSM through the CA NSM SSM CA OPS/MVS Option to CA NSM for management on a WorldView map.

Event Management

The Event Management components in CCS for z/OS enables you to route messages and commands bidirectionally between any instance of Event Management, including other CCS for z/OS, CA NSM agents, and CA OPS/MVS. Event management decisions can be made on the platform that makes the most sense.

Integration between the Event Management components in CCS for z/OS and CA NSM provides a focal point for message management throughout your heterogeneous networked environment.

The following components are required for the establishment of communications between CA OPS/MVS and CA NSM through Event Management:

- CA OPS/MVS USS interface to enable communications between CA OPS/MVS and CCS Event Management
- CAICCI services for connection between the CA OPS/MVS system and the CA NSM server to enable communications between the Event Management components

  Note: For information about how to configure CAICCI, see the *CA Common Services for z/OS Administration Guide*.

USS Interface to Event Management

The CA OPS/MVS USS interface to z/OS Event Management does the following:

- Lets Event Management console messages be available in OPSLOG
- Lets USS rules take action on the Event Management console messages
- Lets CA OPS/MVS send commands and messages to z/OS Event Management or any other CCS Event Management connected platform

Environment

The ADDRESS USS environment is available to enable you to send commands and messages to Event Management.

Set Up the USS Interface

For CA OPS/MVS to communicate with CCS Event Management, you must set up the USS interface.

Note: This topic provides an overview of how to set up the USS interface. For detailed information, see the *Installation Guide*.

To set up the USS interface for CCS Event Management

1. Ensure that Event Management is installed on your system.
2. Customize the INSTUSEX job in the CNTL data set, and submit it.

   The CA OPS/MVS message exit is copied to the Event Management file system.
3. Ensure that the following statements are included in your OPSSPA00 program:

   ```
   var = OPSPRM("SET","INITUSS","YES")
   var = OPSPRM("SET","USSRULES","YES")
   var = OPSPRM("SET","USSACTIVE","ON")
   ```

   These changes take effect when CA OPS/MVS is recycled.

Centralized Monitoring of Mainframe and Distributed Systems

By consolidating messages to the CA NSM Event Console, you can monitor the health of your mainframe and distributed systems from a central location. CA OPS/MVS can send messages to the Event Console by using the ADDRESS USS WTO command. A typical implementation can be as follows:

1. You want the Event Console to display messages that alert the operations staff to certain conditions on your mainframe systems.
2. You create rules to detect those conditions.
3. In each rule, you include the following command so that when the rule triggers, a message (*message_text*) is sent to the CA NSM server (*nsm_host_name*) for display on the Event Console:

   ```
   ADDRESS USS "WTO TEXT('"message_text"') NODE(nsm_host_name)"
   ```

   Note: The ADDRESS USS WTO command provides various keywords that let you specify attributes for the message to be sent (for example, color). For more information, see the *Command and Function Reference*.
4. When the rules are enabled, messages are sent using CAICCI to CA NSM as the monitored conditions occur.
5. CA NSM can further process these messages (for example, correlate them with other messages and take appropriate actions).

CA NSM SSM CA OPS/MVS Option

The CA NSM SSM CA OPS/MVS Option provides real-time graphical monitoring and control of z/OS resources monitored by the SSM component of CA OPS/MVS. It is based on the CA NSM distributed object repository and manager/client architecture and runs on a Windows workstation.

Resources that are monitored and managed by SSM appear as icons on the CA WorldView Map. These icons change color to show when the current state of a resource does not match the desired state. Operators can view these icons, note the color of an icon, and determine whether a problem exists. They can then drill down to further isolate the problem, and use the CA NSM SSM CA OPS/MVS Option Viewer application to determine the cause of the change in state. Also, the operator may use the Viewer to correct the problem by setting particular SSM attributes for the selected z/OS resource. You can use the CA NSM SSM CA OPS/MVS Option Viewer on a client Windows workstation and through a supported Web browser.

OPS/REXX Function

The OPSMTRAP OPS/REXX function enables you to synchronize CA NSM with the resources defined to SSM.

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

Set Up the Option

The CCS Agent Technology service is required for the CA NSM SSM CA OPS/MVS Option.

Note: This topic provides an overview of how to set up the Option. For detailed information, see the *Installation Guide*.

To set up the Option

1. Ensure that Agent Technology is installed on your system.
2. Install the agent for the Option using the INSTSMPM job in the CNTL data set.
3. Define the agent to Agent Technology:
   a. Load the SSM management information base (MIB) using the LDMIB job in the CNTL data set.

      The MIB is loaded in an Agent Technology object store.

   b. Create the configuration file for the agent, based on the CFGSSMO member in the CNTL data set. For #SNMPTRAP, specify the address and port number of the CA NSM server on which the Option is to be installed.

   c. Load the file using the LDCFG job in the CNTL data set.
4. Configure CA OPS/MVS for the agent:
   a. Ensure that the following statements are included in your OPSSPA00 program:

      ```
      var = OPSPRM("SET","CAUNICONFIGSET","agent_configuration_name")
      var = OPSPRM("SET","CAUNICONNECTWAIT","2")
      var = OPSPRM("SET","INITAWS","YES")
      ```

   b. Ensure that the following data sets are concatenated to STEPLIB or defined to LNKLST:
      - CCS CAILIB load library
      - USSLOAD load library
      - CEE.SCEERUN C run-time library

   c. Ensure that the following data sets are allocated to CA OPS/MVS:
      - IBM's Communications Server TCPIP.DATA data set using the SYSTCPD ddname
      - CCS ENVFILE member in the SRCLIB data set using the ENVFILE ddname
      - Agent log data sets using the OPSALOG and OPSBLOG ddnames
      - Agent STDERR data sets using the OPSAERR and OPSBERR ddnames
      - Agent STDOUT data sets using the OPSAOUT and OPSBOUT ddnames
      - Dump data set for C language environment using the CEEDUMP ddname

   d. Recycle CA OPS/MVS.
5. Update the SSM resource and directory tables for the resources you want to monitor through the Option:
   a. Specify **ALWAYS** in the TNGNOTIFY column of a resource table for the resources you want to monitor.

   b. Specify **YES** in the TNGELIGIBLE column of the directory table for each resource table updated in the previous step.

6. Install the Option on the CA NSM server, and set up the WorldView map for the z/OS system on which SSM is running.

   Note: For more information, see the product documentation for the Option.

After the Option is set up, the status of the resources to be monitored are forwarded to CA NSM and reflected on the WorldView map.

Chapter 4: Integrating with CA SYSVIEW

This section contains the following topics:

Overview and Value

Integration with CA SYSVIEW brings direct, automated responses to performance-related problems.

When an exception alert within CA SYSVIEW is triggered based on a defined threshold or state rule, multiple actions can be taken. One action is to send information about an event through console messages, which can be processed by)MSG rules; another action is to use the CA OPS/MVS generic event API to send an event notification to CA OPS/MVS, which can be processed by)API rules. Event notification passes information directly to CA OPS/MVS, including pertinent data in OPS/REXX variables. Using the event notification action reduces the number of console messages and the overhead associated with processing the messages.

By proactively monitoring either type of event, the rules enable CA SYSVIEW commands to be issued automatically and the responses returned to CA OPS/MVS. CA SYSVIEW commands can be issued to modify z/OS or any other monitored subsystem. Updated data collected by CA SYSVIEW is available to OPS/REXX programs, providing continuously current information for intelligent automated response. This interface enables CA OPS/MVS automation to display and control CA SYSVIEW performance metrics, data collection settings, and monitoring tools.

Environment, Programs, and Rules

The following environment, programs, and rules are available to help you monitor and respond to CA SYSVIEW events:

- The ADDRESS SYSVIEWE environment enables you to send commands to CA SYSVIEW through OPS/REXX programs and retrieve the responses.

  Note: For more information, see the *Command and Function Reference* and the *CA SYSVIEW Performance Management Administration Guide*.

- CA OPS/MVS provides the following sample OPS/REXX programs and rules in the &hlq.CCLXSAMP and &hlq.CCLXRULS data sets:

  - The SPOOLMON program monitors and responds to threshold warnings for JES2 spool space and track groups from CA SYSVIEW.
  - The SYSVALRT program, together with the SYSVALRT)TOD rule, enables you to create an application to monitor and respond to system alerts from CA SYSVIEW.
  - The SYSVECMD program, together with the SYSVE)CMD rule, sends CA SYSVIEW commands.

- CA SYSVIEW generates CAGSV* events that you can use in)API rules.

  Note: For more information, see the *CA SYSVIEW Performance Management Administration Guide*.

Enable Event Notification from CA SYSVIEW

For CA OPS/MVS to process event notifications from CA SYSVIEW, you must configure CA SYSVIEW to enable event notification.

Event notification is enabled in CA OPS/MVS by default through the APIACTIVE parameter.

To enable event notification in CA SYSVIEW, set OPSMVS-EVENT-NOTIFICATION to **YES** in either or both of the following members in its BASE.SCSYPARM data set: SYSDATA and CICSOPTS.

Note: For more information, see *CA SYSVIEW Performance Management Administration Guide*.

Chapter 5: Integrating with CA MIM

This section contains the following topics:

Overview and Value

CA MIM provides seamless integration with CA OPS/MVS in several areas, including product state management, health check status management, and individual automation event management.

You do not need to do anything for CA MIM to enable the product interface to CA OPS/MVS and you do not need to issue any CA MIM initialization statement or commands to activate the interface. If CA MIM and CA OPS/MVS are active in the same z/OS image, CA MIM automatically communicates CA MIMautomation events to CA OPS/MVS.

Before the CA MIM-to-CA OPS/MVS interface existed, CA OPS/MVS could automate CA MIM events through console messages, and CA OPS/MVS can still use CA MIM console message traffic for automation events. Console messages are processed by CA OPS/MVS)MSG rules. Depending upon the particular automation event, CA OPS/MVS rules may need to correlate console messages and, in some instances, issue commands and interrogate command responses to collect information about a given event.

On the other hand, the CA MIM-to-CA OPS/MVS interface is an integrated two-way communication mechanism. CA MIM passes automation event information directly to CA OPS/MVS, including all pertinent data related to the automation event in the form of OPS REXX variables. These events can be automated using a CA OPS/MVS)API rule. This interface is more efficient and reduces the complexity of CA OPS/MVS automation rules as the event data in readily available in REXX variables. This internal interface also eliminates the dependence on the format and content of console message text.

Also, CA MIM can take action on an automation event as directed by a CA OPS/MVS)API automation rule. CA MIM and CA OPS/MVS seamlessly work together to automate the management of shared-system resources.

Environment, Program, and Rules

The following environment, programs, and rules are available to help you monitor and respond to CA MIM events:

- The ADDRESS OPER environment enables you to send commands across systems through CA MIC and retrieve the responses.

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

- The MEDSMIM OPS/REXX program in the SAMPLES data set gathers environmental information about CA MIM.
- CA OPS/MVS provides the following sample rules:
  - The SSMCAAPI)API rule in the &hlq.CCLXRULM data set, and the APIHRTB1, APIHRTB2, and APIHRTB3)API rules in the &hlq.CCLXRULS data set monitor and respond to the status of CA MIM. These rules respond to the CASTATE and CAHEARTBT events.

    Note: For more information about the named events, see the *CA MIM Resource Sharing Programming Guide*.

  - The APIMIMGR)API rule in the &hlq.CCLXRULS data set data set monitors and responds to a VARY delay event from CA MIA. This rule responds to the MIM2211 event.

    Note: For more information about the named event, see the *CA MIA Tape Sharing Programming Guide*.

  - The GCM)API rule in the &hlq.CCLXRULS data set excludes internal CA MIC messages from OPSLOG.

Set Up Interface to CA MIC

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

- The CA OPS/MVS subsystem can issue cross-system 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 command issuer and may optionally be recorded in the OPSLOG.
- The CA OPS/MVS subsystem can receive unsolicited messages from any system in the MICplex and record them in the OPSLOG.
- 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-sysplex systems, systems in multiple sysplexes, or VM systems where CA MIC for VM is running as a service machine. Messages from up to 128 systems can now be forwarded through CA MIC to any CA OPS/MVS subsystem.

When all of your systems are in a single sysplex, you can use sysplex services to perform most of these functions. However, the CA MIC message filtering criteria are superior to those provided by sysplex. If you have licensed the Multi-System Facility (MSF), you can perform these functions by using the SYSTEM keyword of OPSCMD and ADDRESS OPER and by writing AOF rules to forward messages from one system to another.

Configure the Interface

For instructions on how to configure CA MIC to do the following, see the *CA MIC Message Sharing Systems Programmer Guide*:

- Use the LINK command to enable the cross-system command and response feature
- Use the COLLECT command to have CA MIC import unsolicited messages to local CA OPS/MVS subsystem

If you only intend to use the CA MIC cross-system command interface and do not want to automate the command responses or have them displayed in OPSLOG, then no CA OPS/MVS configuration is required.

If you intend to display CA MIC imported messages in OPSLOG, you must set the BROWSEMESSAGES parameter to MVSGLOBAL. If you intend to have CA MIC imported messages automated by AOF rules, you must set the AOFMESSAGES parameter to MVSGLOBAL.

Note: Changing this parameter may have a major impact on your automation.

In most sites that run both products, CA OPS/MVS is usually started prior to CA MIC. However, if CA MIC is started before CA OPS/MVS and the CA OPS/MVS SSIMSG parameter is set to a value of YES, you will find that the CA MIC internal encrypted messages (all of which have message IDs that start with GCM/) appear in the OPSLOG. We recommend that you always start CA OPS/MVS before CA MIC. However, if that sequence does not fit into your automation scheme, use the following sample rule (which has also been included in member GCM of the OPS/MVS sample rules library) that demonstrates how to exclude all the GCM messages from the OPSLOG.

Note: You should not attempt to suppress these GCM/ messages or you will impact the functionality of CA MIC.

```
)MSG GCM/* NOOPSLOG
)PROC
return
```

Identify Messages Received from CA MIC

When writing AOF rules you need to be aware that CA MIC imported messages have the following attributes:

- The MSG.MIC environmental variable is set to 1.
- The MSG.REISSUE environmental variable is set to 1.
- The MSG.SYNA environmental variable contains the name of the system from which the message originated.
- The MSG.JOBNM environmental variable contains the job number of the task that originally issued the message. This field contains a value of NONE when the originating task was a z/OS subsystem or a VM application, which did not have a job number.
- The MSG.JOBID environmental variable contains the job number of the task that originally issued the message. This field contains the MVS subsystem name or the VM application name when the originating task was a z/OS subsystem or a VM application, which did not have a job number.
- The MSG.JOBNAME environmental variable contains the name of the task that originally issued the message.

CA MIC presents imported messages to CA OPS/MVS using the above 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 AOF rules interrogate the MSG.SYNA, the MSG.REISSUE environmental variables, or both to identify the systems from which messages are originating. Otherwise, these rules may misinterpret CA MIC imported messages as being from the local system, which may result in unpredictable or incorrect actions.

The following sample AOF MSG rule allows imported CA MIC messages to be easily identified in OPSLOG. Filtering on the USER column with a value of MIC limits the display to CA MIC imported messages. The display can also be limited to those messages imported from a particular system by filtering on the COLOR column.

Note: This logic colorizes all imported messages from systems XE13, XE12, and XE07. If you only want to colorize the CA MIC imported messages, the select statement needs to be subject to the MSG.MIC = 1 condition. If you decide to implement this rule, we suggest that you merge the rule logic into any existing MSG * rules that you may have.

```
)MSG *
)PROC
if MSG.MIC = 1 then
  MSG.USER = "MIC"
select
  when MSG.SYNA = "XE13" then
    MSG.COLOR = OPSCOLOR("TURQ")
  when MSG.SYNA = "XE12" then
    MSG.COLOR = OPSCOLOR("YELLOW")
  when MSG.SYNA = "XE07" then
    MSG.COLOR = OPSCOLOR("PINK")
  otherwise
    nop
end
```

Chapter 6: Integrating with CA PDSMAN

This section contains the following topics:

Overview and Value

Resource state messages from CA PDSMAN can be sent as event notifications to CA OPS/MVS using the CA OPS/MVS generic event API. These messages can be processed by)API rules to provide automated responses to partitioned resource situations.

Event notification passes information directly to CA OPS/MVS, including pertinent data in OPS/REXX variables.

The integration between CA PDSMAN and CA OPS/MVS enhances the ability of your operations staff to automate or track system operations, and to monitor and manage your partitioned library environment.

Rule

CA PDSMAN generates the PDSM_RSM event that you can use in)API rules. The APIPDSMN sample)API rule in the &hlq.CCLXRULS data set monitors and responds to this event.

Note: For more information about the PDSM_RSM event, see the *CA PDSMAN PDS Library Management Partitioned Resource Monitoring User Guide*.

Enable Event Notification from CA PDSMAN

For CA OPS/MVS to process event notifications from CA PDSMAN, you must configure CA PDSMAN to enable event notification.

Event notification is enabled in CA OPS/MVS by default through the APIACTIVE parameter.

To enable event notification in CA PDSMAN, set the OPSMVS parameter to **YES** in the relevant $MONITOR control statements.

Note: For more information, see the *CA PDSMAN PDS Library Management Administration Guide*.

Chapter 7: Integrating with CA 7 WA

This section contains the following topics:

Overview and Value

Through integration, the CA OPS/MVS knowledge of events both internal and external to the job scheduling process alleviates the complexities of job scheduling through automated responses to various batch and job process conditions.

CA OPS/MVS provides tight, two-way communications with the CA 7 WA scheduling product, including built-in REXX address environments that enable CA OPS/MVS automation to display and control scheduling activities. For example, demand CA 7 WA to schedule a job in response to mainframe events. In addition, through integration with the CA 7 WA Browse Log, CA OPS/MVS gains awareness of CA 7 WA events.

For integration, you need to configure CA GSS. If you want CA OPS/MVS to access the CA 7 WA browse log, you also need to configure the CAIENF service.

Environment, OPS/REXX Function, Programs, and Rules

The following environment, function, programs, and rules are available to help you monitor and respond to CA 7 WA events:

- The ADDRESS CA7 environment enables you to send commands to CA 7 WA through OPS/REXX programs and retrieve the responses.

  CA 7 WA provides the CA7OPSRX OPS/REXX program in its CAICLS0 data set. This program uses ADDRESS CA7 to send commands.

  Note: For more information, see the *CA 7 Workload Automation Interface Reference Guide*.

- The OPSCA7 OPS/REXX function enables you to send commands to CA 7 WA through OPS/REXX programs.

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

- CA OPS/MVS provides the following sample OPS/REXX programs in the SAMPLES data set:
  - The ADDRCA7 and UCC7 programs demand CA 7 WA to schedule a job and retrieve the job information.
  - The SHUTCA7 program shuts down a CA 7 WA region.
  - The SHUTCA7I program shuts down CA 7 WA Independent Communications Manager (ICOM).
- The sample CAS9200I and CAS9300E)MSG rules in the &hlq.CCLXRULS data set start and stop the interface to CA 7 WA browse log based on the status of the CAIENF service.

Enable ADDRESS CA7

For CA OPS/MVS to use the ADDRESS CA7 environment, you must configure the CA GSS service to enable the environment.

To enable ADDRESS CA7, add the following statement in your CA GSS parameter library:

```
ADDRESS CA7 CAL2X2WR 15 TYPE 0
```

This change takes effect when CA GSS is recycled.

Access the CA 7 Browse Log

You must add a data control module (DCM) to the ENF database to access the CA 7 WA Browse log.

To access the CA 7 WA browse log

1. Install CA 7 WA Release 3.3 or higher.
2. Set the CA OPS/MVS INITCA7 parameter to YES in the OPSSPA00 member. If you want CA OPS/MVS to generate ENF-related trace messages, then you must also set the CA OPS/MVS DEBUGENF parameter to YES. Additionally, depending on the volume of browse messages that CA 7 WA produces, you may need to tailor the default values of the CAIENFMAX and CAIENFRATE parameter. For more information on these parameters, see the *Parameter Reference*.
3. Add the DCM to ENF. Verify with CA 7 WA Technical Support that their SAMPJCL contains an L232DCM1 job.

   This job installs the CA 7 WA browse event. An ENF EVENT command listing all of the DCMs that are installed should display:

   ```
   DCM module name: CAL2DCM1 Description: CA 7 BROWSE EVENT Installed
    date: 01.010 time: 11:25:20
   ```

Chapter 8: Integrating with CA Jobtrac

This section contains the following topics:

Overview and Value

Integration between CA OPS/MVS and CA Jobtrac lets you issue CA Jobtrac commands in response to messages. For integration, you need to configure CA GSS.

Environment

The ADDRESS JOBTRAC environment is available to enable you to send commands to CA Jobtrac through OPS/REXX programs and retrieve the responses.

Enable ADDRESS JOBTRAC

For CA OPS/MVS to use the ADDRESS JOBTRAC environment, you must configure the CA GSS service to enable the environment.

To enable ADDRESS JOBTRAC, add the following statement in your CA GSS parameter library:

```
ADDRESS JOBTRAC GJTRGCUU 5 TYPE 3
```

This change takes effect when CA GSS is recycled.

Chapter 9: Integrating with CA Scheduler

This section contains the following topics:

Overview and Value

Integration between CA OPS/MVS and CA Scheduler lets you issue CA Scheduler commands in response to messages. For integration, you need to configure CA GSS.

Environment

The ADDRESS CASCHD environment is available to enable you to send commands to CA Scheduler through OPS/REXX programs and retrieve the responses.

Enable ADDRESS CASCHD

For CA OPS/MVS to use the ADDRESS CASCHD environment, you must configure the CA GSS service to enable the environment.

To enable ADDRESS CASCHD, add the following statement in your CA GSS parameter library:

```
ADDRESS CASCHD CAJCADDR 15 DETACH TYPE 0
```

This change takes effect when CA GSS is recycled.

Chapter 10: Integrating with CA NetMaster Products

This section contains the following topics:

Overview and Value

CA NetMaster products enable you to manage your mainframe networks and provide an alert monitor that warns you about things that need attention. Integration between CA OPS/MVS and CA NetMaster products lets you consolidate system and network alerts on a single monitor.

Environment

The ADDRESS NETMASTR environment is available to enable you to create and maintain alerts on the CA NetMaster alert monitor.

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

Enable CA NetMaster to Handle External Alerts

For CA OPS/MVS to create and maintain alerts on the CA NetMaster alert monitor, you must configure the CA NetMaster region to enable the feature.

To enable CA NetMaster to handle external alerts, specify **YES** in the Enable External Alerts? field of the ALERTS parameter group.

Note: For more information, see your CA NetMaster documentation.

Chapter 11: Integrating with z/VM Systems

This section contains the following topics:

Overview and Value

Integration with z/VM systems extends automation control to those systems. z/VM messages can be processed by)GLV and)MSG rules.

Note: For information about the format of the z/VM messages to be processed, see the *Administration Guide*.

Integration is provided by a TCP/IP socket-based communications application. The application runs on a z/OS system as a server, and on one or more z/VM systems as clients.

Environment and Programs

The following environment and programs are available to help you monitor and respond to z/VM events:

- The ADDRESS OPER environment enables you to send commands to z/VM systems and retrieve the responses.

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

- The sample COBCMDS1 COBOL program, CPIND OPS/REXX program, and PLICMDS1 PL/1 program in the SAMPLES data set send z/VM commands and retrieve the responses.

Establish Communication

For CA OPS/MVS to communicate with a z/VM system, you must set up the OPVMSV server on the z/OS system and the OPVMCL client on the z/VM system. The z/VM system must have an application that supports REXX sockets (for example, CA VM:Operator).

Note: This topic provides an overview of how to establish communication between CA OPS/MVS and z/VM systems. For detailed information, see the *Administration Guide*.

To set up the OPVMSV server on the z/OS system

1. Set the following parameters in the OPVMSV member of the data set allocated with the SYSEXEC ddname:

   zosipaddr

   Specifies the address of the z/OS system.

   zosipport

   Defines the port of the server.

   Customize other parameters as required.

   The server is configured.

2. Customize the OPVMJCL member in the CNTL data set.

   The server started task or job is configured.

3. Execute the OPVMJCL member.

   The server starts and can communicate with the clients.

To set up the OPVMCL client on a z/VM system

1. Customize the OPSMCL member of the CA OPS/MVS REXX data set:
   - Set zosipaddr and zosipport to the same values as for the OPVMSV member.
   - Set zvmtargetid to the z/VM user ID for the application that will communicate with CA OPS/MVS.

   Customize other parameters as required.

   The client is configured.
2. Transfer the customized OPSMCL member to a read-only minidisk on the z/VM system.
3. Define a z/VM user directory entry for the client with the following requirements:
   - Set MACHINE to ESA.
   - Set storage size to at least 8 MB.
   - Ensure the user has access to the minidisk used in Step 2.
4. Execute OPSMCL.

   The client starts and can communicate with the server.

Chapter 12: Integrating with CICS

This section contains the following topics:

Overview and Value

The CICS Operations Facility (COF) is an interface between CA OPS/MVS and CICS that extends the capability for AOF rule processing to CICS messages, which are written only to CICS transient data queues. This additional message traffic expands the number of automatable events that you can use to control CICS subsystems. Events that are visible to AOF rules using the COF include terminal failures, the logon and logoff activities of the user, and journal switches.

With the COF interface installed, a single copy of CA OPS/MVS can handle an unlimited number of CICS address spaces.

The interface enables CA OPS/MVS to seamlessly integrate CICS regions with the CA OPS/MVS operation of z/OS and JES. It enables you to issue various operator-oriented CICS transactions (CEMT) through the z/OS MODIFY console command if security and console definitions are properly defined to CICS. You have the capability to use the write-to-operator (WTO) event on all transient data messages so they can be displayed on a specific console, the system log, and OPSLOG, and periodically produce a special CICS heartbeat message to ensure that transaction activity is proceeding as usual in CICS.

Install the XTDOUT COF Interface for CICS/TS

The following list pertains to the CICS/TS interface:

- It uses the CICS global exit (XTDOUT) to intercept all transient data write requests. CA OPS/MVS matches a transient data queue name against a list of designated queue names for AOF processing.
- Messages sent to the matched queue names are forwarded to the AOF for rules processing, which also allows for message suppression and rewording. Messages sent to unmatched queue names are ignored by the exit.
- You build and maintain the designated queue name list with the ADDRESS OPSCTL COF command.
- No changes to the standard CICS DCT are required to intercept transient data messages and the selection of specific destinations can be dynamically altered.
- You can build a distinct queue name list for each CICS region, and a general default list for undefined CICS regions.

To install the XTDOUT COF interface

1. Copy load module OPCITDCN from SYS1.OPS.CCLXLOAD to a library in the CICS DFHRPL concatenation.

   The module is linked AMODE=31 and RMODE=ANY.

2. Define the transaction and program to CICS using the CICS RDO facility:

```
DEFINE GROUP(OPXTDOUT) PROGRAM(OPCITDCN)
   DATALOCATION(ANY) EXECKEY(CICS)
   LANGUAGE(ASSEMBLER) RESIDENT(YES)
   DESCRIPTION(OPS/MVS XTDOUT GLOBAL EXIT)
DEFINE GROUP(OPXTDOUT) TRANSID(OPTD) PROGRAM(OPCITDCN)
   TASKDATAKEY(CICS) TASKDATALOC(ANY)
   DESCRIPTION(OPS/MVS XTDOUT EXIT CONTROL)
INSTALL GROUP(OPXTDOUT)
ADD GROUP(OPXTDOUT) LIST(DFHLIST)
```

   The XTDOUT exit code is contained in the OPCITDCN program, and it is enabled as an entry point address in this module using the name OPCITDEX. The exit program does not need to be defined to CICS.

3. Enable the XTDOUT exit by invoking OPTD from a CICS terminal or with a MODIFY command from a z/OS console. OPCITDCN may be added to the CICS PLTPI stage 3 for automatic exit enablement at CICS initialization when desired.

4. Activate the AOF processing of CICS messages by setting the INITCOF and CICSAOF parameters to YES and define, at the least, the default transient data queue name list.

```
X = OPSPRM('SET','INITCOF','YES')
X = OPSPRM('SET','CICSAOF','YES')
ADDRESS OPSCTL "COF DEFINE JOBNAME(DEFAULT)",
     "DESTIDS(CSMT,CSSL,CADL,...)"
```

   The XTDOUT COF interface for CICS/TS is installed.

For information on permitting the suppression of transient data queue messages by AOF rules, see the description of the CICSDELETE parameter in the *Parameter Reference*.

The OPTD transaction may be used to disable and re-enable the exit at any time by invoking OPTD with a single character command code as follows:

- OPTD E-Enable the XTDOUT exit (default command)
- OPTD D-Disable the XTDOUT exit
- OPTD S-Display the status of the XTDOUT exit
- OPTD T-Issue a test message to the transient data queue
- OPTD H-Issue the periodic CICS status message, OPS3420O

Chapter 13: Integrating with IMS

This section contains the following topics:

Overview and Value

The IMS Operations Facility (IOF) is an interface between CA OPS/MVS and IMS that extends the CA OPS/MVS facilities to IMS. For example, you can write AOF rules that process IMS messages, and you can use OPSVIEW to operate IMS.

A single copy of CA OPS/MVS can handle up to 32 copies of IMS. If you run multiple copies of IMS under the control of one copy of CA OPS/MVS, the copies of IMS may be any combination of IMS levels that CA OPS/MVS supports.

The IOF interface enables CA OPS/MVS to seamlessly integrate IMS control regions with the CA OPS/MVS operation of z/OS and JES. IMS requires its own master terminal, which receives most IMS related messages. The IOF interface dynamically inserts an IMS Automated Operator (AO) exit to capture the IMS command and message traffic, and also identifies and captures z/OS messages that are associated with the IMS. It provides integrated support for IMS DB/DC, data communication control (DCCTL), and database control (DBCTL) environments.

To issue IMS commands and retrieve command responses, the IOF interface can use a batch message processing (BMP) region, if it is available, or it can use the IMS write-to-operator-with-reply (WTOR) method when it needs to issue an IMS command.

Configure IOF

Before you use IOF, you need to set up its parameters. If you do not have AO exits installed, you must also install the supplied sample exits.

Note: This topic provides an overview of how to configure IOF. For special considerations, see the *Installation Guide*.

To configure IOF

1. Ensure that the following statements are included in your OPSSPA00 program:
   - Include the following statement to activate IOF:

     ```
     var = OPSPRM("SET","INITIMS","YES")
     ```
   - Include the following statement for each IMS control region with which IOF communicates:

     ```
     var = OPSPRM("SET","IMSnID","ims_id")
     ```

   Include statements for other parameters as required.

   These changes take effect when CA OPS/MVS is recycled.

   Note: For more information about IOF-related parameters, see the *Parameter Reference*.
2. Install the supplied sample exits if you do not have AO exits installed:
   - Customize the OPSAOUE0 job in the CNTL data set, and submit it.

     A type 1 AO exit is installed.
   - Customize the OPSAOE00 job in the CNTL data set, and submit it.

     A type 2 AO exit is installed.
3. (Optional) Set up a BMP region.

   CA OPS/MVS can issue IMS commands through this region instead of WTOR messages.

   Note: For information about how to set up the BMP region, see the *Installation Guide*.

Chapter 14: Integrating with Tivoli OMEGAMON XE

This section contains the following topics:

Overview and Value

The CA OPS/MVS AOF component can respond to exceptions detected by any or all of the Tivoli OMEGAMON XE on z/OS products. Currently, this means that CA OPS/MVS can interact with the following products:

- Tivoli OMEGAMON XE on z/OS
- Tivoli OMEGAMON for IMS on z/OS
- Tivoli OMEGAMON XE for CICS on z/OS

Tivoli OMEGAMON XE for DB2 Performance Monitor/Expert on z/OS.Exceptions can be processed by)OMG rules.

Interface to Exception Analysis Process

The AOF cannot directly automate OMEGAMON exception messages because they are not routed through z/OS console support. Fortunately, all Tivoli OMEGAMON XE on z/OS products support a log file onto which they write a copy of their logical exception screen at the end of each analysis interval. The size of the OMEGAMON logical screen is one of its startup parameters (LROWS), and users typically set it to a size much greater than the number of lines on the physical screen. Thus, while important exception messages may not appear on the physical screen of an OMEGAMON for lack of room, they will fit on the OMEGAMON logical screen and therefore are written to the log file.

Interface to OMEGAMON

To establish an interface between CA OPS/MVS and OMEGAMON, insert an O*x*REPORT DD statement in the OMEGAMON JCL procedure that uses the SUBSYS keyword to identify CA OPS/MVS as the target of that file.

```
SEND OMEGAMON MVS EXCEPTIONS TO CA OPS/MVS
//OMREPORT     DD   SUBSYS=(OPSS,OMEGAMON,MVS),
//            DCB=(RECFM=FBA,LRECL=81,BLKSIZE=81)
              SEND OMEGAMON CICS EXCEPTIONS TO CA OPS/MVS
//OCREPORT     DD   SUBSYS=(OPSS,OMEGAMON,CICS),
//            DCB=(RECFM=FBA,LRECL=81,BLKSIZE=81)
              SEND OMEGAMON CICS EXCEPTIONS TO CA OPS/MVS;
              IDENTIFY SOURCE CICS SYSTEM
//OCREPORT     DD   SUBSYS=(OPSS,OMEGAMON,CICS,CICSTEST),
//            DCB=(RECFM=FBA,LRECL=81,BLKSIZE=81)
              SEND OMEGAMON IMS EXCEPTIONS TO CA OPS/MVS
//OIREPORT     DD   SUBSYS=(OPSS,OMEGAMON,IMS),
//            DCB=(RECFM=FBA,LRECL=81,BLKSIZE=81)
              SEND OMEGAMON DB2 EXCEPTIONS TO CA OPS/MVS
//ODREPORT     DD   SUBSYS=(OPSS,OMEGAMON,DB2),
//            DCB=(RECFM=FBA,LRECL=81,BLKSIZE=81)
```

The format for the JCL examples above is as follows:

```
//ddname DD SUBSYS =(ssid,OMEGAMON,type{,reportid})
```

ddname

Specifies the ddname associated with the file.

ssid

Specifies the four-character CA OPS/MVS subsystem ID to which these messages are routed (usually OPSS).

type

Identifies the specific Tivoli OMEGAMON XE on z/OS product. The *type* must be MVS, CICS, IMS, or DB2.

reportid

(Optional) Specifies a unique report ID you can use in a rule to identify the source of the message.

Provide OMEGAMON Exceptions

Ensuring that the correct output is being written to the *O*xREPORT file requires some OMEGAMON customization. Customization includes choosing thresholds and options to create and define a profile. Use the OMEGAMON User Profile Facility to customize these parameters.

When using the AOF to automate OMEGAMON exceptions, note the following:

1. CA OPS/MVS must be started before OMEGAMON.
2. An OMEGAMON session must be active to feed the exception event process. If you have a dedicated mode terminal next to the console that is always left on the exception analysis screen, use that terminal to provide the exception data. If that terminal often displays other screens, then you risk missing important exceptions when the operators use it for other functions. CA OPS/MVS can monitor exceptions only while the exception analysis screen is active.

   The simplest way to configure the interface is to have a dedicated session with exception analysis always active. However, this solution has two drawbacks, the first of which is mentioned in the previous paragraph. The second drawback is that it requires a locally attached 3270 device.

   An alternative solution is to use the OMEGAMON VTAM interface with an EPI logical terminal. This solution is more complicated to configure, but it eliminates both of the problems associated with a dedicated 3270 terminal. The EPI session is hidden, so no one can walk up and change the screen. No real 3270 terminal is required, since the EPI is used as a virtual 3270.
3. Check the LROWS parameter of the OMEGAMON started task JCL to ensure that all exceptions fit on the logical screen that is written to the *O*xREPORT file. The default value for the LROWS parameter is two times the physical screen minus one; the maximum value is 999.
4. All exceptions must be unboxed, either by setting the BOX parameters to NO for all exceptions or by turning boxes off in the installation or user profile. You cannot alter the default profile. You can set some control options with the .SET command and you can set some exception thresholds using the XACB command, the XSET command, or both. Use these commands on the actual exception analysis screen for testing, but for production usage, place them in the installation or user profile so that they execute at OMEGAMON startup.

   Note: OMEGAMON installation procedures and actual commands can vary from one platform to another. The commands referenced above may be specific to OMEGAMON for MVS. Consult the appropriate installation guide for the IMS, CICS, and DB2 releases.
5. Set the page limit for the OMEGAMON *O*xREPORT file to a high number. To do so, either specify .PLM 999999999 in a screen space or preferably use the PAGELIMIT option in the user profile.

6. The OMEGAMON logging facility must be turned on. You need to issue the LOGON command to OMEGAMON to tell it to write screens to the *O*xREPORT file.

   OMEGAMON 7.1.0 and OMEGAMON II Configuration for dedicated terminals:

   Create an initial screen space and enter the following commands on separate lines following the rules for creating OMEGAMON screen spaces (commands should start in column 2):

   ```
   OUTP REPORT
   DDNM OPREPORT      (or whatever DDNAME is used in proc)
   .LOGOUT
   .LOGON
   .FGO exscrn
   ```

   exscrn

   Specifies the name of the screen space containing the exception analysis command.

   OMEGAMON II Configuration for OMVTAM:

   a. Create an initial screen space and enter the following commands on separate lines following the rules for creating OMEGAMON screen spaces (commands should start in column 2):

      ```
      OUTP REPORT
      DDNM OPREPORT       (cannot be OMREPORT)
      .LOGOUT
      .LOGON
      .FGO exscrn
      ```

   b. Logon to OMVTAM:

      ```
      LOGON APPLID(OMVTAM) DATA('FSCR=yyyy')
      ```

      yyyy

      Specifies the name of the screen space containing the commands described above. The purpose of the initial screen space (in either dedicated or VTAM mode) is to configure the logging facility when OMEGAMON starts. The .FGO command then transfers control to the exception analysis screen space, which then remains on the screen and drives the exception analysis process on a regular interval (the OMEGAMON session must be in auto-update mode).

7. Invoke exception analysis through one of these commands: LEXSY (for OM), LXIMS (for OI), or LCXSY (for OC). Place the command in column 1 and be sure to prefix it with an L. The L tells OMEGAMON to label the exception by putting its four-character name on the screen in addition to the message. These exception names are the message IDs that CA OPS/MVS uses to invoke its OMEGAMON rules.

At this point, you should see OMEGAMON messages appearing in OPSLOG, and you can enable rules to execute in response to them. Each exception generates a message each time the screen is refreshed, so you may want to review your exception thresholds and your refresh time to ensure that you do not flood OPSLOG with unimportant messages. Use the CA OPS/MVS BROWSEOMG parameter to keep OMEGAMON messages from appearing in OPSLOG. If you set the BROWSEOMG value to OFF, you can audit the occurrence of OMEGAMON messages that execute OMEGAMON rules by including a SAY statement or an ADDRESS WTO host command that reports the text of the exception message processed in the rules.

If you are licensed to use the OMEGAMON Exception Logging Facility (XLF), then you may want to consider using the XLFLOG DD as an alternative to the OMEGAMON report file. The XLFLOG has the advantage that it does not repeatedly generate exception events to CA OPS/MVS every OMEGAMON cycle. If you choose to use XLF, then you must customize the OMEGAMON exception analysis values for persist and limit.

Chapter 15: Integrating with Tivoli NetView

This section contains the following topics:

Overview and Value

Integration between CA OPS/MVS and Tivoli NetView is provided by the NetView interface and NOF. It enables you to combine the system automation capabilities of CA OPS/MVS with the network automation capabilities of Tivoli NetView. Automation is done where it logically belongs, with both CA OPS/MVS and Tivoli NetView aware of the activities of each other.

The NOF interface includes the following benefits:

- Two-way management for NetView alerts
- VTAM message handling
- Interface to the NetView STATMON (status monitoring) feature

Note: For information about how to use NOF, see the *User Guide*.

Install the NetView Interface

Perform these steps to install the CA OPS/MVS NetView interface:

1. To gain access to the NetView unsolicited message stream, module OPNVEX11 must be relinked with a NetView exit alias name of DSIEX11. To add the DSIEX11 alias using SMP/E, apply usermod OPUM003 contained in library SYS1.OPS.CCLXCNTL(USEREX11). If necessary, copy exit module OPNVEX11 and alias DSIEX11 from SYS1.OPS.CCLXLOAD to a library in your NetView STEPLIB concatenation. If you already have a DSIEX11 exit in NetView, then modify it to include the logic in the CA OPS/MVS-supplied exit. Copy exit DSIEX11 from SYS1.OPS.CCLXLOAD to a library in your NetView STEPLIB concatenation.

   DSIEX11 resides, in source format, in SYS1.OPS.CCLXASM. The exit sends unsolicited messages to the master console, which in turn routes them through the subsystem interface where CA OPS/MVS can access and automate them.

   Important! If you decide to no longer use the DSIEX11 exit, then you can safely delete the alias name from the load library. The DSIEX11 exit is only an alias name of the OPNVEX11 module.

2. Copy SYS1.OPS.CCLXEXEC(OPSALERT) to a data set in your NetView DSICLD concatenation. This program is a NetView REXX EXEC.

3. Establish a connection between NetView and your MCS master console. The interface to issue NetView commands from CA OPS/MVS rules or REXX programs is the same interface that IBM provides to enable NetView commands to be issued from z/OS consoles.

   Use a NetView AUTOTASK command to create an association between the MCS master console and a NetView user ID. You can issue this command at a NetView terminal, or in the NetView initial CLIST member.

   The command has this syntax:

   ```
   AUTOTASK CONSOLE=consolenumber,OPID=operatorid
   ```

   consolenumber

   The MCS console number.

   operatorid

   The NetView operator ID to be associated with the MCS console. NetView operator IDs are defined in the DSIOPF member of the NetView parameter data set.

   The CA OPS/MVS NetView interface assumes that you have established an association between a NetView operator ID and your MCS master console. Consider modifying your NetView start up CLIST to issue the AUTOTASK command for you when NetView starts.

4. Modify your NetView startup CLIST to issue the following command:

   ```
   OPSALERT NOTIFY
   ```

   This command issues a series of NPDA set recording filter (SRF) commands to give CA OPS/MVS access to NetView alert information. You can issue this command without restarting NetView after you complete step 2 of this NetView installation process.

5. Copy the message table entry in SYS1.OPS.CCLXCNTL(OPSAUTO) to your NetView automation message table. If you do not have a NetView automation message table, copy the OPSAUTO member to a data set in the NetView DSIPARM concatenation and issue the following command:

   ```
   AUTOMSG MEMBER=OPSAUTO
   ```

   To have this command invoked automatically at NetView startup, place it in the NetView start up CLIST.

   If you already have a NetView message table, copy the message table entry to the bottom of your existing message table and reactivate it with the AUTOMSG command.

6. Copy SYS1.OPS.CCLXEXEC(ALERT) to a library that CA OPS/MVS rules can access (that is a library in the SYSEXEC concatenation). Doing this enables CA OPS/MVS rules and programs to use the ALERT function.

7. CA OPS/MVS message rules can set or reset a bit in the MSG.AFLAGS variable.

   For more information, see the *AOF Rules User Guide*.

8. To use the subset of POI command processors that can run as NetView command processors, define each command processor in the DSICMD member of the NetView parmlib.

   For example:

```
*-----------------------------------------------*
*     CA OPS/MVS NETVIEW CAPABLE POI COMMANDS  *
*-----------------------------------------------*
OPSGETV  CMDMDL  MOD=OPSGETV,TYPE=R,RES=Y,PARSE=N
           CMDCLASS=1,2,3
OPSGETVL CMDMDL  MOD=OPSGETVL,TYPE=R,RES=Y,PARSE=N
           CMDCLASS=1,2,3
OPSSETV  CMDMDL  MOD=OPSSETV,TYPE=R,RES=Y,PARSE=N
           CMDCLASS=1,2,3
OPSDELV  CMDMDL  MOD=OPSDELV,TYPE=R,RES=Y,PARSE=N
           CMDCLASS=1,2,3
OPSQL    CMDMDL  MOD=OPSQL,TYPE=R,RES=Y,PARSE=N
           CMDCLASS=1,2,3
*-----------------------------------------------*
*        AUTOMATE/MVS COMMAND ALIASES OF OPSMODE  *
*-----------------------------------------------*
GETVAR   CMDMDL  MOD=GETVAR,TYPE=R,RES=Y,PARSE=N
           CMDCLASS=1,2,3
GETVARL  CMDMDL  MOD=GETVARL,TYPE=R,RES=Y,PARSE=N
           CMDCLASS=1,2,3
SETVAR   CMDMDL  MOD=SETVAR,TYPE=R,RES=Y,PARSE=N
           CMDCLASS=1,2,3
DELVAR   CMDMDL  MOD=DELVAR,TYPE=R,RES=Y,PARSE=N
           CMDCLASS=1,2,3
SQL      CMDMDL  MOD=SQL,TYPE=R,RES=Y,PARSE=N
           CMDCLASS=1,2,3
OPSMODE  CMDMDL  MOD=OPSMODE,TYPE=R,RES=Y,PARSE=N
           CMDCLASS=1,2,3
```

If the load modules defined in the example above are not available in the system linklist or LPA, you must add a STEPLIB for the CA OPS/MVS load library to the NetView procedure JCL and the library must be APF-authorized.

Since multiple CA OPS/MVS subsystems may be active on one system, default routing of all command requests to a desired subsystem name can be accomplished by allocating a dummy data set with a ddname of OPS$*xxxx*, where *xxxx* is the subsystem name; you may use JCL or the NetView ALLOCATE command. OPSS is the default subsystem name. For example:

```
ALLOC FILE(OPS$OPST) DUMMY
//OPS$OPST DD DUMMY
```

The CA OPS/MVS security rules do not currently have access to the NetView user ID for security checking of global variable access. To permit global variable access to NetView command processors, you must enable a generic security rule for the NetView address space. For example:

```
)SEC OPSGLOBAL*
)PROC
  If Opsinfo('JOBNAME') = 'netview job name' Then
    Return 'ACCEPT'
  Else
    Return 'NOACTION'
)END
```

Install the NetView Operator Facility

Installing the NetView Operator Facility (NOF) requires you to make changes to both your CA OPS/MVS and NetView environments. You may want to consult the NetView systems programmer at your site for help with installing the NOF.

The NOF resides on the CA OPS/MVS distribution media. It uses the following libraries:

- The CA OPS/MVS sample rules library, OPS.CCLXRULB
- The CA OPS/MVS NetView CLIST library, OPS.CCLXCLS0

  Note: We also provide this library in variable block format.
- The CA OPS/MVS load library, OPS.CCLXLOAD
- The CA OPS/MVS control library, OPS.CCLXCNTL

To install the NOF

1. Create a rule set to house the sample rules supplied with the NOF by performing one of the following steps:
   - Create a new rule set.
   - Copy all of the members from OPS.CCLXRULB into an existing rule set.

   If your site uses the CA OPS/MVS SECURITYRULESET parameter, copy OPNFSEC into your security rule set. OPNFSEC is a security rule that gives NetView access to CA OPS/MVS global variables. You can use the OPNFCPYR job in the OPS.CCLXCNTL data set to copy OPNFSEC and other NOF rules.
2. Copy the NetView REXX programs from OPS.CCLXCLS0 to a library in the DSICLD concatenation in NetView. You can use the OPNFCPYE job in the OPS.CCLXCNTL data set to do this.

   Note: If you only concatenate the CA OPS/MVS load library to the NetView STEPLIB concatenation, then the CA OPS/MVS DSIEX11 module (part of the former CA OPS/MVS NetView interface) gets control. See Installing the NetView Interface in this chapter.

3. Make the OPS.CCLXLOAD library available to NetView. If your CA OPS/MVS load library is in the z/OS LNKLST, NetView already has access to it. Otherwise, you need to copy the following load modules named from OPS.CCLXLOAD to your NetView STEPLIB library. You can use the OPNFCPYL job in the OPS.CCLXCNTL data set to accomplish this.

   OPNFSGLV enables NetView to set CA OPS/MVS global variables.

4. Include the entries from the OPNFATBL member of the OPS.CCLXCNTL data library in your NetView message automation table. These entries trap events that CA OPS/MVS is interested in. We recommend that you use the NetView %INCLUDE feature to include the OPNFATBL entries, because this method enables you to maintain the CA OPS/MVS table entries separately.

5. Configure a user ID called OPSMAIN on NetView so that OPSMAIN is a task that starts automatically when NetView starts. You can use an existing autotask if you change the OPNFATBL member to route messages to it.

   Note: Using a new autotask is preferable, because doing so enables you to use the NetView TASKUTIL command to track NOF resource consumption. The easiest way to create the autotask is to copy the autotask definition for AUTO1, which is a standard NetView autotask.

6. (Optional) If you want to use the NetView STATMON interface, modify the DSICMN member of the NetView parameter library (typically, DSIPARM) by removing comments from the statements that begin with the text SENDMSG. Activating these statements will cause the NetView status monitor to issue CNM094I messages whenever a managed resource changes state. You can control the volume of CNM094I messages by determining which types of resources should generate these messages.

7. Make sure that the NetView subsystem address space is active. This is required to generate NetView alerts. You can use the CA OPS/MVS System State Manager feature to manage this address space.

   Note: You can use the OPSNETV function of OPS/REXX to determine the status of the NetView subsystem address space. For more information about OPSNETV, see the *User Guide*.

8. Use NetView LOADCL commands to load the NOF REXX programs into storage. This enables NetView to use the in-storage copy of the program instead of having to get it from disk for every message and alert.

9. Modify the NetView startup procedure to issue the appropriate alert filtering commands. These commands are:

   NPDA SRF (set recording filter)

   Specifies which alerts you want to keep and filters out alerts you do not want. To enable all alerts to flow to NPDA, to be displayed on the NPDA screen, and to be automated by CA OPS/MVS, you must issue the following command in your NetView startup CLIST or after NetView is active:

   ```
   NPDA SRF AREC PASS DEFAULT
   ```

   NPDA SVF (set viewing filter)

   Specifies which alerts you want to see. To enable all alerts that CA OPS/MVS generates to appear on the NPDA display, issue the following command in your NetView start up CLIST or after NetView is active:

   ```
   NPDA SVF PASS DEFAULT
   ```

After you have completed the steps listed above, the NOF is ready to operate. When you activate your new NetView message automation table, the NOF will behave like your existing DSIEX11 module (that is, if your DSIEX11 module echoes unsolicited VTAM messages to the console, so will the NOF). At this point, you may want to set up your NOF parameters using the OPNOF command.

Chapter 16: Integrating with CA Service Desk

This section contains the following topics:

Overview and Value

CA OPS/MVS can automatically open CA Service Desk requests for the following types of problems:

- Recoverable product abends
- Shortages of process blocks, which are necessary for automation
- Failure to respond to internal MSF ping requests
- Operator Server Facility (OSF) TSO server transactions that exceed their elapsed time or output line limits
- Automated Operations Facility (AOF) rules that fail to complete due to errors

This provides your organization with an immediately recorded notification of a problem so that it can be addressed before causing more serious problems. For integration, you need to configure the CAICCI and CAISDI services.

Enable CA Service Desk Requests

For CA OPS/MVS to open CA Service Desk requests, you must load the CA OPS/MVS data files in CA Service Desk, configure the CAICCI and CAISDI services, and enable CA Service Desk requests in CA OPS/MVS.

To enable CA Service Desk requests

1. Load the CA OPS/MVS data files in CA Service Desk using the following command. The data files are in the $NX_ROOT\data\integrations\ directory.

   `pdm_userload -f data_file_name`

   Note: For more information about the pdm_userload command, see the *CA Service Desk Administration Guide*.

   a. Load the integOPMVS.dat data file.

   b. (Required for ITIL-compliant CA Service Desk only) Load the itil_integOPSMVS.dat data file.

   CA Service Desk is configured to accept CA Service Desk requests from CA OPS/MVS.

2. Ensure that the CAICCI and CAISDI services are configured. Add a PRODUCT and an EVENT control statement to the CAISDI/med parameter member, MEDPARMS, in your parmlib.

   The CAISDI service is configured to pass CA OPS/MVS CA Service Desk requests to CA Service Desk using CAICCI.

   Note: For detailed information, see the *CA Common Services for z/OS CA Service Desk Integration Guide*.

3. Include the following statement in your OPSSPA00 program:

   `var = OPSPRM("SET","INITSD","YES")`

   CA OPS/MVS is configured to be able to open CA Service Desk requests. This change takes effect when CA OPS/MVS is recycled. If you want to make this change available to the current CA OPS/MVS region, use OPSVIEW Option 4.1.1 to set the parameter in the current region and recycle the CA OPS/MVS CA Service Desk Integration component, SERVDESK.

   Note: For more information, see the *Parameter Reference*.

Create CA Service Desk Requests

The address SERVDESK CREATE request environment lets you create a request on CA Service Desk.

To create a CA Service Desk request, use the following syntax:

```
address SERVDESK "REQ(CR) SUMM('Summary text.') DESC('Description text.') WAIT(25)"
```

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

Index