

CA Common Services for z/OS

Best Practices Guide

Release 14.1.00



Second Edition

This Documentation, which includes embedded help systems and electronically distributed materials, (hereinafter referred to as the "Documentation") is for your informational purposes only and is subject to change or withdrawal by CA at any time.

This Documentation may not be copied, transferred, reproduced, disclosed, modified or duplicated, in whole or in part, without the prior written consent of CA. This Documentation is confidential and proprietary information of CA and may not be disclosed by you or used for any purpose other than as may be permitted in (i) a separate agreement between you and CA governing your use of the CA software to which the Documentation relates; or (ii) a separate confidentiality agreement between you and CA.

Notwithstanding the foregoing, if you are a licensed user of the software product(s) addressed in the Documentation, you may print or otherwise make available a reasonable number of copies of the Documentation for internal use by you and your employees in connection with that software, provided that all CA copyright notices and legends are affixed to each reproduced copy.

The right to print or otherwise make available copies of the Documentation is limited to the period during which the applicable license for such software remains in full force and effect. Should the license terminate for any reason, it is your responsibility to certify in writing to CA that all copies and partial copies of the Documentation have been returned to CA or destroyed.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CA PROVIDES THIS DOCUMENTATION "AS IS" WITHOUT WARRANTY OF ANY KIND, INCLUDING WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NONINFRINGEMENT. IN NO EVENT WILL CA BE LIABLE TO YOU OR ANY THIRD PARTY FOR ANY LOSS OR DAMAGE, DIRECT OR INDIRECT, FROM THE USE OF THIS DOCUMENTATION, INCLUDING WITHOUT LIMITATION, LOST PROFITS, LOST INVESTMENT, BUSINESS INTERRUPTION, GOODWILL, OR LOST DATA, EVEN IF CA IS EXPRESSLY ADVISED IN ADVANCE OF THE POSSIBILITY OF SUCH LOSS OR DAMAGE.

The use of any software product referenced in the Documentation is governed by the applicable license agreement and such license agreement is not modified in any way by the terms of this notice.

The manufacturer of this Documentation is CA.

Provided with "Restricted Rights." Use, duplication or disclosure by the United States Government is subject to the restrictions set forth in FAR Sections 12.212, 52.227-14, and 52.227-19(c)(1) - (2) and DFARS Section 252.227-7014(b)(3), as applicable, or their successors.

Copyright © 2012 CA. All rights reserved. All trademarks, trade names, service marks, and logos referenced herein belong to their respective companies.

CA Product References

This document references the following CA Technologies products:

- CA ACF2™
- CA Datacom®/AD
- CA Mainframe Software Manager
- CA NSM
- CA OPS/MVS®
- CA Top Secret®

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>.

Best Practices Guide Process

These best practices represent years of product experience, much of which is based on customer experience reported through interviews with development, technical support, and technical services. Therefore, many of these best practices are truly a collaborative effort stemming from customer feedback.

To continue and build on this process, we encourage users to share common themes of product use that might benefit other users. Please consider sharing your best practices with us.

To share your best practices, contact us at techpubs@ca.com and preface your email subject line with "Best Practices for *product name*" so that we can easily identify and categorize them.

Documentation Changes

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

- [Change the Characteristics of a Protocol](#) (see page 72)—Corrected the syntax such that PARM=! in both samples.

Contents

Chapter 1: Introduction	11
Purpose of this Guide	11
Audience	11
Mainframe 2.0 Overview.....	11
Mainframe 2.0 Features.....	12
Chapter 2: Installation and Configuration Best Practices	15
Mainframe 2.0 Best Practices	15
Chapter 3: CA Agent Technology Best Practices	17
System and CA Common Services Requirements	17
Configure Agent Technology	21
Mainframe Agent Installation Considerations	22
Startup Procedures	24
Installation Verification	24
Performance Considerations.....	25
The Discovery Process.....	28
Chapter 4: CAIENF Best Practices	33
Installation Considerations.....	33
Files	34
Upgrades	35
Multiple Systems.....	37
Security	37
CAIENF Database Queries	37
CAIENF Health Check.....	38
Chapter 5: Event Management Best Practices	39
Verify Your Installation.....	39
Upgrade Considerations.....	42
zSeries File System (zFS) Structure.....	42
Repository	44
Maintenance Considerations	44
Implementation Considerations.....	45
Use Store and Forward (Optional)	45

Clean up the Log Files	46
Running Without the Repository	46
Running With the Repository	47
Message Actions	48
Write Messages Using cawto	49
Reply to Messages Using cawtor.....	50
Issue Console Commands Using CAconsole	50
Route z/OS Console Messages to Event Management	51
SNMP Traps	52
Startup Procedures	53
Java GUI	54
Timeout Settings	54
Repository Maintenance	57
Required Tables and Initialization Records	57
Create a Backup	58
Clone your Current Database	58
Use a Central Database on Multiple Systems	58
Event Management Examples.....	59
Call cawto from a Batch Program	59
Call cawto in the PARM Statement	59
Call cawto from a Script	60
Integration with CA OPS/MVS.....	60
Write a Message to a Remote Node	61
Write a Message to a Remote CA NSM Machine with an Active Web Server	61
Issue a Command on a Remote CA NSM Machine.....	61
Troubleshooting	62
Check Job Output	62
Check Environment Variable Settings	62
Check Space on the zFS File	62
Check Log Files	63
Check Security Permissions.....	63
Check Tracing Variables	63
Resolve Problems Accessing the CA Datacom/AD Database	64
Protect Event Management under zFS Security	64
Web Server Requirements	64
NSMJSERV (Java Server) Requirements	65
NSMEMSTR Requirements	66
Java GUI Requirements	67
Chapter 6: CAICCI Setup Examples	69
Setup VTAM Parameters	69

Setup TCPIP GW Parameters.....	70
Setup Multiple TCP/IP Address Spaces	70
Setup Multiple CCITCP Address Spaces.....	71
Change the Characteristics of a Protocol	72
Setup XCF/XES Parameters.....	73
Setup CCISL, CCISL GW, and Certificate Deployment	73
Setup CCILGR/Assured Delivery	74
REORG the VSAM File.....	75
Activate the Assured Delivery Feature.....	76
Query the CCILGR.....	77
Allocate the VSAM File	78

Index

Chapter 1: Introduction

This section contains the following topics:

[Purpose of this Guide](#) (see page 11)

[Audience](#) (see page 11)

[Mainframe 2.0 Overview](#) (see page 11)

[Mainframe 2.0 Features](#) (see page 12)

Purpose of this Guide

The guide provides a brief introduction to the CA Technologies mainframe management strategy and features, and describes the best practices for installing and configuring CA Common Services for z/OS.

Audience

The intended audience of this guide is systems programmers and administrators who install, configure, deploy, and maintain CA Common Services for z/OS.

Mainframe 2.0 Overview

Mainframe 2.0 is our strategy for providing leadership in the mainframe operating environment. We intend to lead the mainframe marketplace for customer experience, Out-Tasking solutions, and solution innovation. After listening to customer needs and requirements to keep the mainframe operating environment viable and cost-effective, we are providing new tools to simplify usage and to energize this operating environment for years to come.

CA Mainframe Software Manager™ (CA MSM) is an important step in realizing the Mainframe 2.0 strategy. CA MSM simplifies and standardizes the delivery, installation, and maintenance of mainframe products on z/OS systems. CA MSM has a browser-based user interface (UI) with a modern look and feel for managing those solutions. As products adopt Mainframe 2.0 features and CA MSM services, you can acquire, install, and manage your software in a common way.

CA MSM provides software acquisition and installation that make it easier for you to obtain and install CA mainframe products, and apply the recommended maintenance. The services within CA MSM enable you to manage your software easily based on industry accepted best practices. The common browser-based UI makes the look and feel of the environment friendly and familiar.

We follow the IBM z/OS packaging standards using SMP/E, with some additional CA qualities of service added, to make installation simple and consistent. Additionally, through the synchronization of product releases and the use of common test environments, we will declare a yearly mainframe software stack that includes many new releases with enhanced functionality. This stack is certified for interoperability across the CA mainframe product portfolio and the base IBM z/OS product stack.

Mainframe 2.0 Features

Mainframe 2.0 has the following main features:

CA Mainframe Software Manager (CA MSM)

Delivers simplified acquisition, installation, and deployment capabilities using a common z/OS-based web application delivered through a browser-based UI. CA MSM includes the following services:

Product Acquisition Service (PAS)

Facilitates the acquisition of our mainframe products and services, including product base installation packages and program temporary fixes (PTFs). This service integrates the inventory of products available on your system with CA Support, providing a seamless environment for managing and downloading software and fixes onto your system.

Software Installation Service (SIS)

Facilitates the installation and maintenance of our mainframe products in the software inventory of the driving system. This service enables you to browse and manage the software inventory using a web interface, and automates tasks for products that use SMP/E to manage installation. You can browse downloaded software packages, and browse and manage one or more consolidated software inventories (CSIs) on the driving system.

Software Deployment Service (SDS)

Facilitates the deployment of CA Technologies mainframe products from the software inventory of the driving system. This service enables you to deploy installed products that are policy-driven with a set of appropriate transport mechanisms across a known topology. The enterprise system topology can include shared DASD environments, networked environments, and z/OS systems. Policies represent a combination of metadata input and user-supplied input. Metadata input identifies the component parts of a product. User-supplied input identifies the deployment criteria, such as where it goes and what it is named.

Electronic Software Delivery (ESD)

Enables you to get our products from an FTP server. We have improved this process so that you no longer need to build a tape to install the product.

Best Practices Management

Integrates with IBM Health Checker for z/OS to verify that deployed software follows our best practices. The health checks continually monitor the system and software to provide feedback on whether the software continues to be configured optimally.

Best Practices Guide

Provides best practices for product installation and configuration.

Note: For additional information about the CA Mainframe 2.0 initiative, see <http://ca.com//mainframe2>.

Chapter 2: Installation and Configuration Best Practices

The topics in this guide provide best practices for implementing Mainframe 2.0 strategies, and installing and configuring CA Common Services for z/OS. This release of Mainframe 2.0 includes the following features:

- CA Mainframe Software Manager
- Electronic Software Delivery
- Standardized installation using SMP/E
- Health Checker

Mainframe 2.0 Best Practices

Because of the number of components included in CA Common Services, and the extensive nature of each component, this guide has a chapter for each component for which we make recommendations.

The remaining chapters of this guide contain the recommended best practices for installation and configuration of specified components.

Chapter 3: CA Agent Technology Best Practices

This section contains specific instructions for CA Agent Technology installation and management. You should consider each of the best practices in this chapter before installing CA Agent Technology and implement them as appropriate.

Business Value

Taking these best practices into consideration will make the installation process smoother and insure that operation and maintenance of CA Agent Technology is most efficient and cost effective.

For example, proper consideration of configuration and installation requirements results in faster installation and better operation after installation, due to adequate resource allocation.

Following proper startup, verification, performance tuning, and discovery procedures results in the most efficient operation of CA Agent Technology and less down time for maintenance and adjustments for poor performance.

This section contains the following topics:

[System and CA Common Services Requirements](#) (see page 17)

[Configure Agent Technology](#) (see page 21)

[Mainframe Agent Installation Considerations](#) (see page 22)

[Startup Procedures](#) (see page 24)

[Installation Verification](#) (see page 24)

[Performance Considerations](#) (see page 25)

[The Discovery Process](#) (see page 28)

System and CA Common Services Requirements

System requirements include TCP/IP, UNIX Systems Services and SNMP.

To install CA Agent Technology, follow the instructions in the *CA Common Services for z/OS Installation Guide*. The installation relies on standard SMP/E procedures RECEIVE, APPLY and ACCEPT. It also requires UNIX System Services under z/OS, including the use of zFS (zSeries File System).

Note: For additional information, see the IBM SMP/E manuals, UNIX System Services manuals, and the z/OS Distributed File Service zSeries File System manuals.

Common Services Prerequisites

There are no CA Common Services prerequisites for Agent Technology itself. However, every agent for which Agent Technology acts as middleware requires the CAIRIM Common Services component for initialization and license checking. Examples of such agents are the z/OS Agent, the CICS Agent, the Memo Agent, and so on..

Files

Two zFS's are required for Agent Technology running on any single system:

- A Read-Only zFS which can be shared by all systems or copied to each system
- A ReadWrite zFS which is specific to a single system.

The Read-Only zFS contains executable code, and the ReadWrite zFS contains configuration files, scripts that can be tailored, and log files.

SMP/E APPLY of Agent Technology populates both the zFS and the CA Common Services .CNSMLOAD load library with the same executables.

The ENVFILE used by the mainframe Agents is delivered in the CNSMOPTV data set. It must reflect the environment settings for this installation and its limited settings must match the same environment settings in the /cai/agent/agentworks.profile on the zFS.

Upgrading

If you are upgrading from a previous release, you should do a base install into a new zFS. Many of the configuration file entries required for this release are similar to the entries required by previous releases. Therefore, it is helpful to retain the previous release's HFS for reference during the Post-Installation configuration phase.

If you are upgrading from a previous release of Agent Technology and you do not intend to re-install the mainframe Agents that you use, you must run the install_agents script found in the \$AGENTWORKS_DIR/services/tools directory in order to register the Agents with this installation of AT. This script can be used to add or remove entries from the awservices.cfg file for any Agents or services currently supported on z/OS. Any of the mainframe Agents using Agent Technology must be started using the new CNSMLOAD and the new ENVFILE found in the CNSMOPTV data set.

Multiple Systems

Detailed instructions for installing on multiple z/OS systems is contained in the Agent Technology Post-Installation chapter of the *CA Common Services Installation Guide*.

Security

One of the key steps to the successful installation of Agent Technology is the establishment of the administrator account (default name, AWADMIN). An installation job is provided to establish this account.

This account must be authorized to:

- Perform job submission
- Write to the zFS (OMVS access)
- Submit APPC transactions

This account must also specify a HOME directory equal to that of the root Agent Technology directory (cai/agent by default). It is not necessary to use the exact Group and User codes (911 and 912 respectively) as those specified in the installation process.

Additionally, if you use the CA Top Secret Security for z/OS product, you may want to amend the example TSS commands to add syntax to create a department specific to Agent Technology. For example:

```
tss cre(awdept) type(dept) name('Agent Technology Department')
```

This department name should then be referenced in the AWGROUP and AWUSER definitions.

Questions occasionally arise regarding the need to grant superuser privileges to user ID's that are used to run one or more of the z/OS-based agents. It is neither required, nor desired, for these ID's to hold this privilege. What is required however, is that these accounts must participate in the Agent Technology security group (default name, AWGROUP), as established in the AT installation process. In addition, if you intend to run your agent as a started task, the Agent Technology security group must be the default group for the task's user ID.

Access to Agent Technology applications and commands may be further tightened by means of the access.off file (which will be renamed if it is to be used) in the /cai/agent/services/config/aws_orb directory. If this extra level of security is desired, please see the comments in that file to determine how to implement the changes. As delivered, this extra security is disabled.

Services

Agent Technology for z/OS is comprised of the following services, which are a subset of the CA NSM services on the distributed platforms:

awsservices

The service controller.

aws_orb

The object request broker (that is, the communications handler).

aws_sadmin

The SNMP handler. This service is also responsible for maintaining the MIBs within the Object Store in the UNIX System Services zFS.

Port usage

All the z/OS based agents communicate with the Agent Technology z/OS services, which in turn communicate with the distributed NSM machines' Agent Technology services. The communication occurs using the TCP/IP protocol. By default, the following ports are used by these services on the mainframe:

- 6665—UDP-type port used by sadmin for SNMP requests.
- 7774—TCP-type port used for ORB->ORB communications (Release 2.2-3.0, 11.0).
- 9990—TCP-type port used by awsservices (request port).
- 9991—TCP-type port used by awsservices (control port).

Review the TCPIP.ETC.SERVICES file to determine if any other service has been registered to use any of these ports. In addition, you should review the BPXPRMnn used at IPL to determine if any of these ports is within a range of ports has been reserved in the INADDRANYPORT and INADDRANYCOUNT settings.

If you need to change the UDP port used by aws_sadmin, you must modify the servicectl command for aws_sadmin in the \$AGENTWORKS_DIR/services/tools/install_agents script for the port, that is change the following:

```
-p 6665    to -p wwww
```

where wwww is the new port number.

Then rerun the script.

If you need to change the TCP port for ORB-> ORB communications, modify the \$AGENTWORKS_DIR/services/config/aws_orb/quick.cfg file with the new port number. For example:

```
PLUGIN QIKSODLL xxxx
```

where xxxx is the new port number.

If you need to change the awsservices request or control port, you can reassign those port numbers by including the following environment variables with the new assignments to the agentworks.profile file in your \$AGENTWORKS_DIR directory:

```
export AWS_STARTER_REQUEST=yyyy  
export AWS_STARTER_CONTROL=zxxx
```

Configure Agent Technology

The *CA Common Services for z/OS Installation Guide* Post-Installation chapter for Agent Technology contains detailed instructions for all files and settings that can be configured. Follow the explicit directions in that chapter to complete the configuration.

Important! The `aws_sadmin.cfg` file must be changed to enable the sending of SNMP traps to destinations. The `ENVFILE` file must also be tailored so that mainframe agents can communicate with Agent Technology. Both these files are discussed below.

Configure the `aws_sadmin` Service

The `aws_sadmin` service is responsible for sending SNMP traps to the distributed NSM machines. To achieve this goal, the `aws_sadmin` service must know the DNS name or IP address (and listener port) for the NSM machines. This information is stored within file `aws_sadmin.cfg` in directory `$AGENTWORKS_DIR/services/config/aws_sadmin`. You must update this file with the appropriate DNS names or IP addresses of your NSM machines. Using the nodename as defined to the DNS server is the preferred method in lieu of the actual IP address which may change. Be sure to preserve the non-displayable tab characters (`x'05`) contained within this file.

These characters are used as field delimiters on the `SNMP_TRAP` lines.

The `aws_sadmin.cfg` file is also used to define the SNMP community strings for the `aws_sadmin` SNMP service. By default, these strings are set to `public` (for read access) and `admin` (for write access). If your site requires more secure SNMP community strings, these names can be changed to meet your needs.

Important! Be sure to preserve the non-displayable tab characters (x'05) contained within this file. These characters are also used as field delimiters on the SNMP_COMMUNITY lines. Also note that any change to the SNMP community strings must be reflected in the configuration of the NSM machines. Be sure to make your NSM administrator aware of these changes. Note that an agent can have its own community strings specified in a configuration set, which is unique to that agent and which overrides the default community strings specified in the `aws_admin.cfg` file.

Environment File (ENVFILE)

The Common Services CNSMOPTV library contains a member named ENVFILE. This member is used by all of the z/OS-based agents (as well as various Agent Technology utilities) to set environment variables required to communicate with the AT services. The configuration data within this member is created when the AT services are installed, and must be accurate. This data is specific to the current LPAR, TCP/IP stack, and AT zFS's. Any change to these items must be reflected in the ENVFILE member. In general, the values must match the equivalent values specified in the `agentworks.profile` script on the ReadWrite zFS.

Important! The Common Services CNSMOPTV library has special DCB attributes that must be maintained.

These attributes are particularly necessary when reading the ENVFILE member.

Communications problems will occur if the ENVFILE member is moved and read out of a standard fixed-block PDS.

Mainframe Agent Installation Considerations

Agent Technology is used by all mainframe agents. This section describes the mainframe agent installation considerations with respect to Agent Technology.

Agent MIB Files

Each agent installation provides the MIB file for the agent being installed. These files must be copied into the Agent Technology MIBLIB when the particular agent is installed.

A sample job AWADDMIB is supplied in the Common Services CNSMSJCL library. This job uses SMP/E to RECEIVE and APPLY a USERMOD containing a mainframe Agent's MIB. It will copy the MIB delivered with the specific Agent into the Agent Technology MIBLIB.

A unique USERMOD name should be used for each Agent's MIB, and the REWORK date should be accurate. Thus the SMP/E environment for Agent Technology will reflect the latest MIB for each Agent.

The MIB must be available each time the AT Object Store is cleared and rebuilt (CLEANADM and LDMIB jobs).

Establishing a Mainframe Agent with Agent Technology

There are several requirements that must be met for Agent Technology to recognize an Agent.

- First, as noted above, the Agent's MIB must be copied into the Agent Technology MIBLIB.
- Second, the MIB must be loaded into the Agent Technology Object Store. This can be accomplished either by using the individual Agent-supplied JCL, by using the `install_mibs` script found in the `$AGENTWORKS_DIR/services/tools` directory, or by using the Agent Technology LDMIB job found in the Common Services CNSMJCL library.
- Third, the Agent itself must be registered with Agent Technology. This can be accomplished either by using the individual Agent-supplied JCL or by using the `install_agents` script found in the `$AGENTWORKS_DIR/services/tools` directory.

JCL Requirements for Communicating Between Agents and AT Services

All z/OS-based agents must communicate with the AT z/OS services running under UNIX Systems Services. The agents communicate to AT Services through a TCP/IP socket connection.

To facilitate this communication, the following modifications must be made to the Agent JCL job-stream for the system being monitored:

- The Common Services CNSMLOAD library must be added to the STEPLIB Concatenation or be in the link list.
- The following DD statements must be added:

ENVFILE

Should be defined to point to the ENVFILE member within the Common Services CNSMOPTV library. This member contains configuration data that points the Agent to the appropriate TCP/IP Port and Stack, and to the AT zFS.

SYSTCPD

Should be defined to point to the TCPDATA file for the local TCP/IP Stack. This is the same file referenced in your TCP Startup proc. This file contains configuration data about the TCP/IP stack.
- Add any log or configuration files mandated by the specific agent being installed.

Startup Procedures

Agent Technology does not depend on any other CA Common Service. Once installed and configured, it can be started any time after UNIX System Services (USS) and TCP/IP have started. None of the z/OS-based agents can be started until Agent Technology for z/OS has started, and all of its services have properly initialized.

Use the Common Services CAIPROC library member AWSTART to start the Agent Technology services. It usually takes 1-2 minutes (after the AWSTART job ends) for the AT services to initialize. Also note that starting the AT Services under z/OS only starts the services and not the Agents themselves.

Each of the agent instances must be individually started and stopped in z/OS.

This action usually occurs within the application being monitored, or through an MVS console command.

Use the Common Services CAIPROC library member AWSTOP to stop the Agent Technology services.

Installation Verification

Once the AT z/OS services have successfully started:

To verify they are running properly

1. Check on the state of the AT Services within OMVS as follows:

a. Issue the following command:

```
. agentworks.profile
```

Note: Do not enclose this command in quotes and do not forget the period and space before the actual invocation of the agentworks.profile script.

b. Issue the following command:

```
awservices list
```

c. The first two lines of the report produced should read:

```
RUNNING aws_orb:aws_orb  
RUNNING aws_admin:aws_admin
```

Note: You may see several other services and agents, or both, listed in the report with a status of Stopped. This is normal.

2. Check the status of the AT services ports within OMVS as follows:
 - a. Issue the following command:

```
onetstat
```
 - b. The report produced should list all of the AT ports (see the Overview section earlier in this chapter) and show them in the appropriate status, as follows:

```
Listen – For all TCP/IP socket ports
UDP – For the SNMP listener port (normally 6665)
```
3. Ensure all of the Agent/Services MIBS have been loaded into the Object Store. To do this, within OMVS:
 - a. Issue the following command:

```
. agentworks.profile
```
 - b. Issue the following command:

```
agentctrl -m
```
 - c. The report produced must show the awsAdmin MIB, as well as at least one other Agent MIB if Agents have been registered. For example:

```
<awsAdmin> is registered
<caiDatacom> is registered
<caiDb2mvs> is registered
<caiIDMS> is registered
<caiSysAgtzOS> is registered
<caiSysAgtCics> is registered
<caiSysAgtMqs> is registered
```
4. In you encounter problems, review the log files within the AT zFS (under directory \$AGENTWORKS_DIR/services/var/log) to help diagnose the problem. These files are overwritten each time the AT services are started.

Performance Considerations

Adequate CPU

Agent Technology is a time critical application. To ensure your CA NSM workstations receive timely responses to their information requests, the necessary software components must receive adequate CPU cycles to process the workload.

These components include:

- The TCP/IP region running on the mainframe
- The Agent Technology Services (and OMVS itself)
- Any (all) z/OS-based agents

This typically means running these components in a high priority performance group, equal to that of other main systems and started tasks (for example, JES, VTAM, CA Top Secret, and so on).

The Object Store Database

The cache sizes of the object store database can be adjusted to minimize I/O EXCPs if the need arises.

There are a total of 22 files that comprise the aws_sadmin object store database.

These files are found within directory

`$AGENTWORKS_DIR/services/var/aws_sadmin`

and can be categorized into the following groups:

- Objects table—6 files prefixed with OBJECTS.
- Titles table—6 files prefixed with PROPTH.
- Properties table—6 files prefixed with PROPVH.
- Votree table—4 files prefixed with VOTREE.

The size of these files depends on the following:

- The number of MIBs loaded in the object store
- The size and number of configuration sets loaded in the store
- The number of agent instances currently running

To minimize the number of EXCPs used to access the object store, the following parameters have been added to the aws_sadmin startup command. The startup of the aws_sadmin service can be found in file awsservices.cfg, within directory services/cfg/awsservices.

- `-x <cache-size for the objects table>`
- `-y <cache-size for the properties table>`
- `-z <cache-size for the votree table>`

No parameter has been established for the Titles table, since the default value should be adequate for all sites. All cache sizes are expressed in KB. When left unspecified, the default value for all three parameters is 1024.

To minimize I/O EXCPs to the object store database, you must set the cache size parameters to the total size of all of the files within each of the different table categories (with the exception of the rollback files, which are suffixed with MMA, RBI, MM2, and RB2). Listed below are the calculations you should perform to determine the size for each of the cache parms:

- -x

Sum the size of the following files (Divide the result by 1024 to convert to KB):

```
OBJECTS.CHK
OBJECTS.FBM
OBJECTS.MDI
OBJECTS.MDS
```

- -y

Sum the size of the following files (Divide the result by 1024 to convert to KB):

```
PROPVH.CHK
PROPVH.FBM
PROPVH.MDI
PROPVH.MDS
```

- -z

Sum the size of the following files (Divide the result by 1024 to convert to KB):

```
VOTREE.CH2
VOTREE.CM2
```

Note: These calculations should be performed only after all MIBs have been loaded and all agent instances have been started. You must ensure that your AT z/OS Services have been shut down before you attempt to modify the awsservices.cfg file (otherwise your changes will not be preserved).

Statistics Collection

As noted previously, earlier, the aws_sadmin service is responsible for the processing all of SNMP requests coming in from the various CA NSM components, as well as for the generation of SNMP traps. In addition to these functions, the aws_sadmin service also maintains the awsAdmin MIB and the associated awsAdmin agent. The awsAdmin MIB contains the following categories of information:

- Configuration information
- Agent and MIB information
- Statistical information

The statistical information is collected in the following different groups:

- AwsAdminAgentGroup
- AwsAdminSnmpGroup
- AwsAdminPerfGroup
- AwsAdminSourceGroup

The maintenance of the statistical information within the awsAdmin MIB represents an appreciable amount of work for the aws_sadmin service. In fact, benchmark tests have shown that as much as 40% of the CPU time spent by the aws_sadmin process is consumed by the updating of all of these statistics.

As a performance-enhancing feature, the following environment variables have been added to control the collection of the statistical information within the awsAdmin MIB. These environment variables let you turn off the collection of any (all) statistics, resulting in the reduction in CPU (as well as increased performance) for the aws_sadmin service.

The environment variables, along with their possible values, are as follows:

```
AW_ADMIN_STAT_AGENT=ON/OFF
AW_ADMIN_STAT_SNMP=ON/OFF/NOTOTAL
AW_ADMIN_STAT_PERF=ON/OFF
AW_ADMIN_STAT_SOURCE=ON/OFF
```

The default value for each variable is ON. If you want to change this value (to improve performance and reduce CPU overhead), define the environment variable within the agentworks.profile file in the root Agent Technology directory (\$AGENTWORKS_DIR) of your zFS system.

The Discovery Process

Occasionally, you may encounter some problems with the discovery of Agents on a z/OS LPAR. The following tips may help you diagnose and resolve these problems.

- Make sure your z/OS node is properly classified.

The first-level discovery process uses your TCP/IP stack's SNMP process to classify your mainframe node. This is not the Agent Technology SNMP service, but rather, this is the SNMP service that comes with the IBM (or third-party) TCP/IP stack. This service must be running at the time the mainframe node is discovered for the node to be properly classified as an IBM3090. IBM's SNMP service (OSNMP) is usually started via an AUTOLOG command in the TCP/IP started task's PROFILE data set.

If the node is not properly classified, it can be reclassified manually from the NSM machine (using the reclass command).

- Community strings must agree between the z/OS `aws_sadmin` service and the NSM machine.

See `Configuring the aws_sadmin Service` subsection in the `Installation Considerations` section earlier in this chapter for more information.

- Is the z/OS SNMP service listener port bound?

Run `onetstat` under `OMVS` to ensure the SNMP listener port (6665 by default) is in the proper state. Check to make sure there are no firewalls preventing UDP traffic from entering your mainframe network.

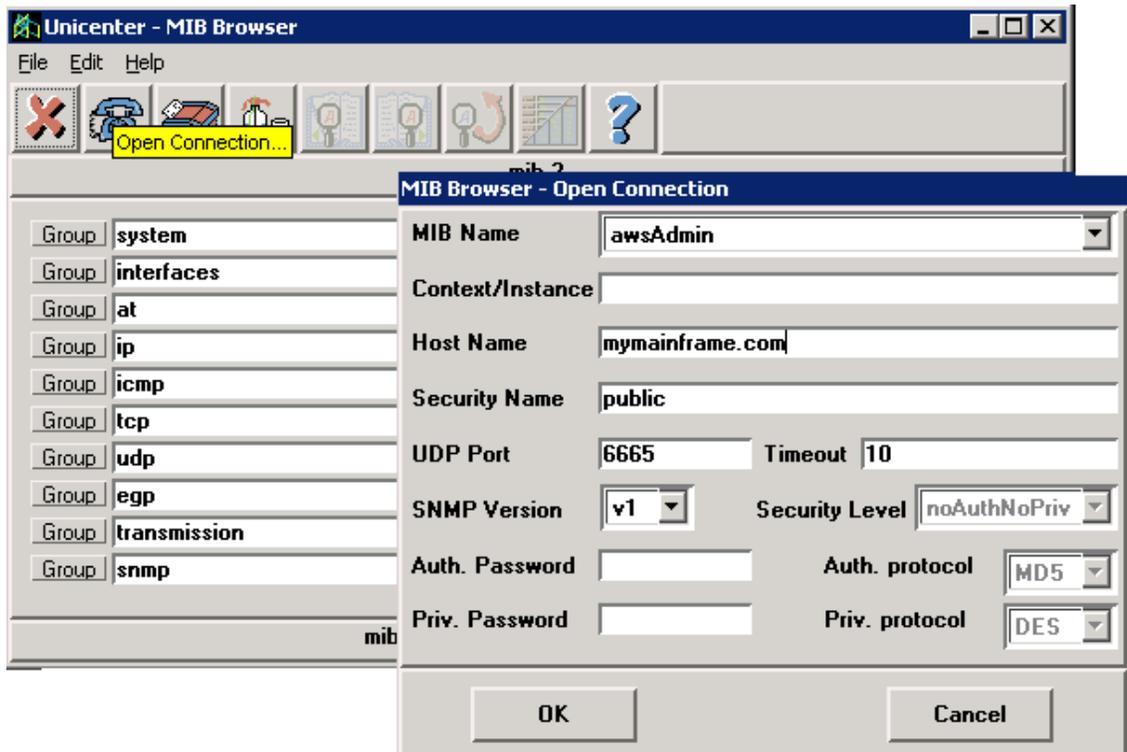
- Is the `awsAdmin` MIB loaded in the z/OS store?

- Can you Mibbrowse the awsAdmin MIB on the mainframe?

The second-level discovery process uses the awsAdmin MIB to discover agents running on z/OS.

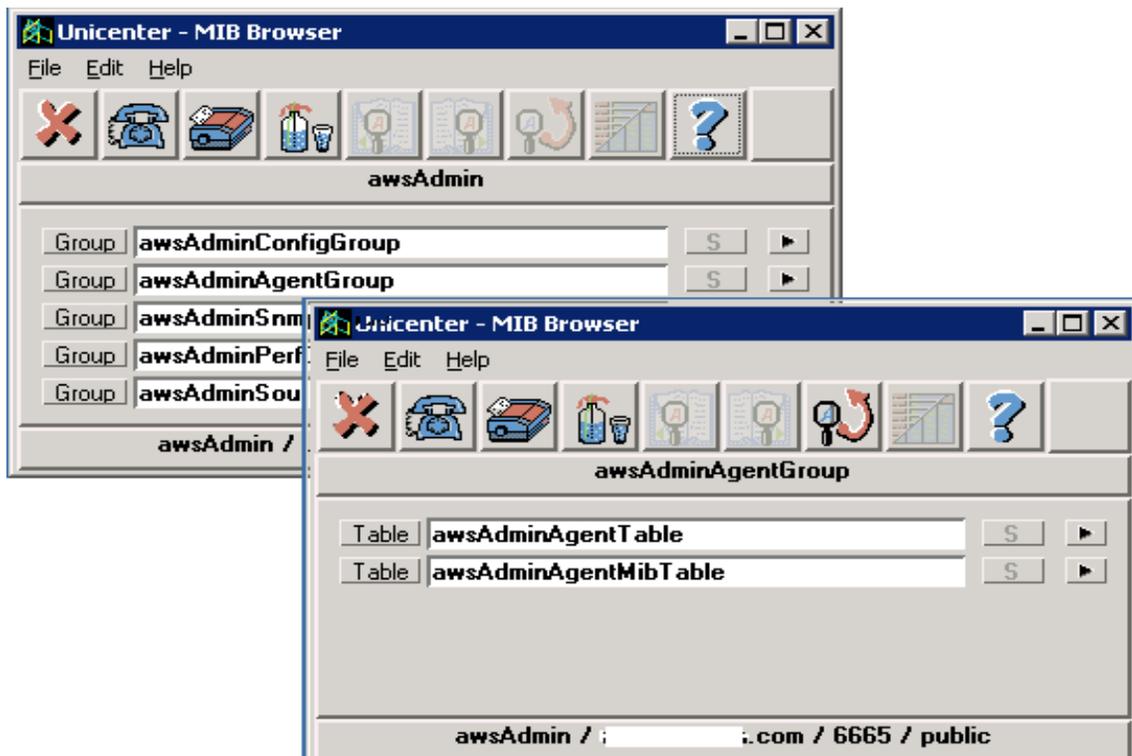
To list active agents using mibbrowse

1. Run mibbrowse from the DOS prompt at the NSM machine to verify this capability.
2. Connect to the awsAdmin MIB on the appropriate host name and UDP port (6665 by default).

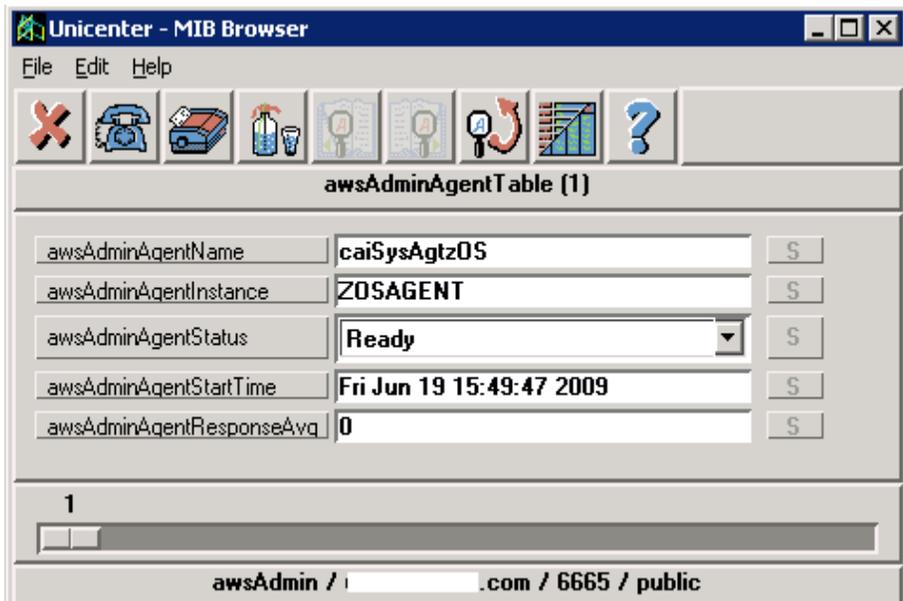


3. Attempt to drill down though
awsAdminAgentGroup->awsAdminAgentTable

Drill down to an agent group: (All active agents in this group should be listed.)



Drill down to an agent table:



Chapter 4: CAIENF Best Practices

This section contains helpful information for the CA Technologies Event Notification Facility (CAIENF). You should consider the best practices in this chapter before installing CAIENF.

Business Value

Taking these best practices into consideration will make the installation process smoother and insure that operation and maintenance of CAIENF is most efficient and cost effective.

For example, if you give the proper attention to the needed files, upgrade considerations, security, and health check, your installation will work better and CAIENF will have the critical features needed to perform best.

This section contains the following topics:

[Installation Considerations](#) (see page 33)

Installation Considerations

CAIENF runs on all IBM supported releases of z/OS.

CAIENF has the ability to record system and CA product generated events in CAIENF database event tables. There are two possible reasons to record events.

- Catch up on previous events. CA products that listen for CAIENF events may catch up on past events if the CA product was down while CAIENF was up and running. This process is called event checkpointing or event recovery. As CA products start up and initialize with CAIENF, they can request that events from a past date and time be posted to the CA product CAIENF listening task before new events are posted. CA products that take advantage of event checkpointing have installation steps to update CAIENF parameters so that specific events will be recorded.
- Capture specific events for analytical reporting purposes. You can interrogate the event database using SQL reports. Running such reports is discussed in more detail later in this guide.

CAIENF event recording requires the installation of CA Datacom/AD at a minimum level of Version 12.0. CAIENF requires that a CA Datacom MUF (Multi-User facility) be established to house the CAIENF database on each system that is running CAIENF. You can run the MUF either within the CAIENF address space or external to the CAIENF address space. We recommend you run the MUF internal to the CAIENF address space. This makes CAIENF run more self-contained and similar to the way CAIENF runs on releases prior to CAIENF r12.

The CAIENF event data is unique to each system, therefore sharing a CA Datacom/AD MUF with other CA products is not practical because other CA products generally share data across systems using a cross system MUF. Sharing a MUF may also not be practical because of maintenance and backup considerations. Do not attempt to share the MUF with other CA products unless the product documentation specifically states otherwise. The CAIENF MUF can be shared with the Event Management common service and it is expected that CAIENF and Event Management will share the same MUF if you are using Event Management message actions or calendars.

Started Task Procs ENFIMUF, which uses an internal MUF, and ENFXMUF, which uses an external MUF, are delivered in the CA Common Services for z/OS CAIPROC data set to serve as a basis for customization.

If no event recording is needed or desired, then the CAIPROC data set delivered member 'ENF' can be customized to run with no database.

Files

CAIENF requires a parameter file to startup. This file should be customized to meet the needs of your site. Some best practices concerning these parameters are:

- If you do not wish to record CAIENF events, then make sure to specify NODB AND RECORD(NO) within the CAIENF Parameters.
- If you need to record events, then make sure to have RECORD(YES) specified.
- It is fairly likely that CA products that perform event checkpointing require only batch job activity. You can cut down on the number of events that are recorded on the CAIENF database and thus cut down on system overhead by filtering events that are recorded to the database. This is done with the CAIENF parameter SELECT. Unless you have specific reasons not to, run with the following SELECT statements:

```
SELECT (JOBINIT, JOBNUM, EQ, J*)
SELECT (JOBTERM, JOBNUM, EQ, J*)
SELECT (JOBPURGE, JOBNUM, EQ, J*)
SELECT (STEPTERM, JOBNUM, EQ, J*)
```

- If you are running a CA product that is interested in data set close events and thus has you turn on recording of data set close events, you should also add the following CAIENF parameter statement:

```
SELECT(DSCLOSE, JOBNUM, EQ, J*)
```

- There can be a very large number of data set close events on a system and CAIENF processing of these events does carry some overhead. CAIENF has the SCREEN parameter that filters out events at event creation time. Screening out events is not recommended unless CA Support has determined that it is in your best interest to do so since all CAIENF product listeners will be affected, not just products that use CAIENF event checkpointing. In addition CAIENF and CAICCI listen for some events. With that caution however, there is a data set close SCREEN setting that is typically safe. Data set close events can be screened out for non-update events and thus lower overhead on the system. The following statement can be added to CAIENF parameters:

```
SCREEN(DSCLOSE, ACCESS, EQ, INPUT)
```

- You may also have some started tasks set up that have a system maintenance or utility purpose. If these started tasks or jobs are run frequently and you know that they have no relationship to any CA product, you can add SCREEN statements to keep their activity outside of CAIENF. For example, you could have the following SCREEN statements:

```
SCREEN(JOBINIT, JOBNAME, EQ, JRDR)
SCREEN(JOBTERM, JOBNAME, EQ, JRDR)
SCREEN(STEPTERM, JOBNAME, EQ, JRDR)
SCREEN(JOBPURGE, JOBNAME, EQ, JRDR)
```

Upgrades

A big change that started with CAIENF r12 is that the CAS9DB utility is no longer used. For releases of CAIENF prior to r12, the CAIENF database was mandatory and CAS9DB was the utility used to initialize and populate the database with DCM definitions. With CAIENF r12 and above, the database is now optional and only required if event recording is desired. For more information, see CAIENF [Installation Considerations](#) (see page 33) in this guide.

If event recording is turned on, the database is only used for event data. The database is no longer used to house DCM information. Rather, CAIENF now directly loads and reads DCM modules as part of CAIENF startup. CAIENF Parameters now tell CAIENF which DCMs to try to load and interrogate. A new CAIDCM DD statement points at the data sets that hold the product DCMs.

The following table provides the CA Common Services r12 and above equivalent of old CAS9DB functions:

CAS9DB DBIN Statement	CAIENF r12 and Above Equivalent
INST DB(ENFDB) DCM(xxxxxxxx)	CAIENF Parameter control statement: DCM(xxxxxxxx) and in CAIENF Procedure: //CAIDCM DD DSN=hlq.CAIDCM,... // DD DSN=hlq.product.llq
ALTER DB(ENFDB) EVENT(xxxxxxxx) OPT(A,R,P) RETPD(nn)	CAIENF Parameter control statements: EVENT(xxxxxxxx,ACT) EVENT(xxxxxxxx,REC) EVENT(xxxxxxxx,RP=n) EVENT(xxxxxxxx,PURGE=Y)
LIST DB(*) DETAIL	Use the new CAS9DCMR utility for DCM event information. Use the CASQL003 sample job in the CAW0JCL data set to obtain recorded event record counts.
QUERY DB(*) SELECT	CA Datacom/AD DBSQLPR utility. See CASQL001, 002, 003 CAW0JCL data set sample members.
INST DB(ENFDB) DCM(xxxxxxxx) REPLACE	Use CAS9DCMR utility SYSPUNCH output data set DROP TABLE statements as input to CAW0JCL member CASQL004. Run CAW0JCL member CADB001 and CASQL004 with CAIENF down, but with an external MUF active.

Some products require an updated level of their DCM to be compatible with CAIENF r12 and above. The CA Common Services for z/OS cover letter supplies CA product PTF numbers or higher product release numbers that must be in place to run with CAIENF r12 and above. If these PTFs have not been applied yet, they can be applied before upgrading to CAIENF Version 14.0.

If you don't have your previous release CAIENF CAS9DB database initialization jobs handy, a utility is available to interrogate the output of a CAS9DB LIST DB(*) DETAIL report and create the appropriate CAIENF ENFPARM statements. The CAS9DB report must be run while you are still running the previous release of CAIENF.

Use the following JCL to obtain a CA Common Services r11 detail listing:

```
//CAS9DB EXEC PGM=CAS9DB,REGION=4096K,TIME=1440
//STEPLIB DD DSN=YOUR.COMMON.SERVICES.R11.CAILIB,DISP=SHR => Update
//DBOUT DD DISP=(NEW,CATLG),
// DSN=YOUR.R11.ENFDB.LISTING, => UPDATE
// UNIT=3390,VOL=SER=??????,SPACE=(CYL,(1,1)),
// DCB=(RECFM=FBA,LRECL=133,BLKSIZE=6118)
//DBIN DD *
LIST DB(*) DETAIL
/*
```

Once you have obtained a detail listing, see the CA Common Services SAMPJCL member ENFUTIL for instructions on how to edit the ENFUTIL JCL. The output from the ENFUTIL job can be placed within the member or data set referenced by the ENFPARMS DD in your CAIENF startup procedure.

Multiple Systems

On any system where CAIENF event recording is turned on, you need a CA Datacom/AD database as described in the CAIENF Installation Considerations section of this guide.

A separate CA Datacom CAIENF database is required on each system that is recording events. The CA Datacom CAIENF database cannot be shared. Each system requires its own CA Datacom/AD MUF that runs either within the CAIENF address space (ENFIMUF), or external to the CAIENF address space (ENFXMUF). A complete CA Datacom/AD installation is not required on each system.

For instructions on creating a new CA Datacom/AD MUF and CAIENF database instance, see the *Installation Guide*.

Security

The only CAIENF security requirement is that the ACID assigned to CAIENF must have an OMVS segment.

CAIENF Database Queries

When CAIENF is setup to record events to the CAIENF database, events can be queried for the following reasons:

- An CAIENF listener seems to have a missing event problem. A good first diagnosis step is to determine if the event was recorded in the CAIENF database.
- You wish to audit what happened when a particular job ran.
- You have some site-specific reason for inspecting recorded event data.

Because CAIENF r12 and above uses a CA Datacom/AD database, standard Datacom SQL jobs can be used to look at event data in the database. CAWOJCL data set members that start with CASQL, provide samples that you can use. The following is a description of the sample members:

CASQL001

Displays the names of the available event tables on the database in case you are not familiar with the event names. You need the event table names to use the other sample CASQLnnn members.

CASQL002

Performs a standard SQL query for reporting on event data.

CASQL003

Reports on the number of recorded events of the specified event name.

CASQL004

(Not for Query purposes.) Used for DCM replacement situations only.

CASQL005

Provides archive data so the restore job can be set up properly in rare cases where a restore from archive becomes necessary.

CAIENF Health Check

CAIENF supplies 1 health check that is designed to detect a misuse of the CAIENF SCREEN parameter. CA support has discovered that if the SCREEN parm documentation is not read carefully, the opposite of the desired outcome might be accidentally setup. The CAIENF health check detects this potential misuse and reports on it if found. The CAIENF Screen parm Health Check name is:

CCS_ENF_SCREEN_VALIDITY

Chapter 5: Event Management Best Practices

This section contains helpful information for the Event Management component. You should consider each of the best practices in this chapter before installing Event Management.

Business Value

Taking these best practices into consideration will make the installation process smoother and insure that operation, maintenance, and troubleshooting of Event Management are most efficient and cost effective.

For example, proper consideration of configuration and installation requirements results in faster installation and better operation after installation, due to adequate resource allocation.

This section contains the following topics:

- [Verify Your Installation](#) (see page 39)
- [Upgrade Considerations](#) (see page 42)
- [Maintenance Considerations](#) (see page 44)
- [Implementation Considerations](#) (see page 45)
- [SNMP Traps](#) (see page 52)
- [Startup Procedures](#) (see page 53)
- [Java GUI](#) (see page 54)
- [Repository Maintenance](#) (see page 57)
- [Event Management Examples](#) (see page 59)
- [Integration with CA OPS/MVS](#) (see page 60)
- [Troubleshooting](#) (see page 62)

Verify Your Installation

Event Management consists of the following:

- Processes that make up the functional components such as message handling, calendars, and remote servers.
- GUI interface, which includes the ability to add messages to act on and the actions to take, the creation of calendars, and the viewing of the event console

Note: Not all sites will have both parts of this verification process.

To verify your Event Management installation

1. Verify that the Processes are Started.
 - Use the CAIPROC member NSMEMSTR to start the functional component.
 - After starting this job, issue the following command:

```
D  OMVS,A=ALL
```

There should be at least one caiopr processes running, ca_calendar, stardaemon, newdaylog, caidoc, and optionally, oprsafd and catrapd.

2. Verify the GUI Interface Servers are Active.

The GUI interface requires that the httpd server is active and the Java backend server be up and running. Use CAIPROC member NSMWEBSV to start the httpd server if you do not already have one started. See the Java GUI section later in this chapter for more information about setting up a new server or merging into an existing one.

3. Start the Java Server.

Use the CAIPROC member NSMJSERV to start the Java server. The following message appears on the system console to signify the startup is complete:

```
CAxx506I – TNG Root Processes Initialized
```

OMVS processes CaemRTS, logonserver.exe, EMServer.exe, and w2Tree.exe should be running at this point.

4. Connect to the GUI.

- To connect to the GUI, start your browser and use the following URL to display the welcome page:

`http://hostname:port/tngfw`

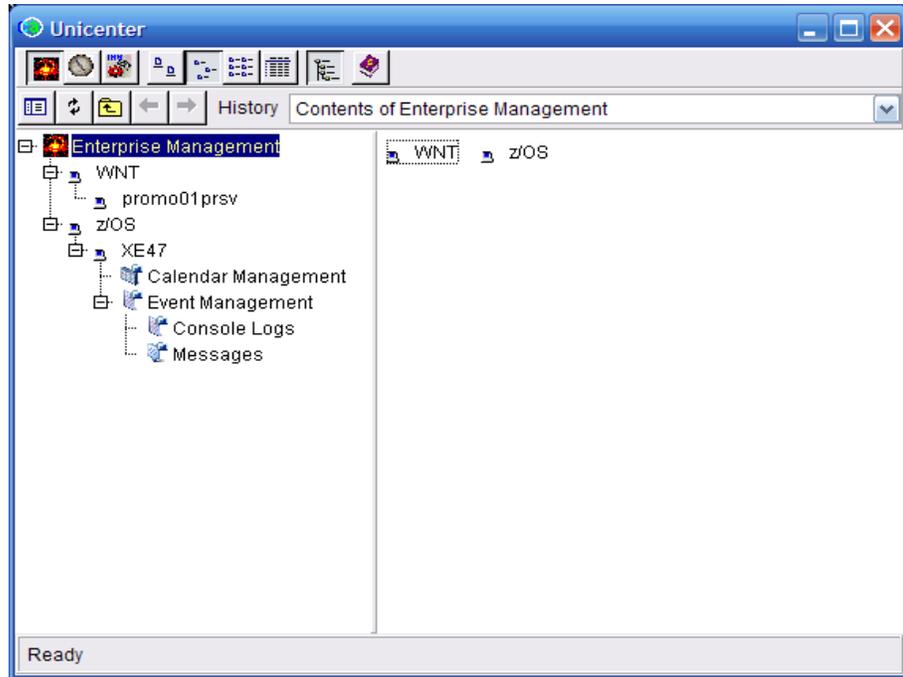
- Once you have the welcome page, select “Launch Unicenter Browser Interface” to launch the Unicenter Browser Interface.



5. Enter a mainframe user ID and password.



- Click OK to display the main CA NSM GUI.



Upgrade Considerations

This section describes upgrade considerations for Event Management.

zSeries File System (zFS) Structure

Prior to installation, decide how you want your zSeries File System (zFS) structured. Event Management r11, delivered with CA Common Services r12 and above, is designed to use a read-only zFS that can be shared across systems and a read-write zFS for each system running Event Management. For example, it is best to mount the RO zFS at /cai/nsmem and mount the RW zFS at /cai/nsmem/RW so that the RW directory is mounted under the RO directory.

Your system must support zFS aggregates. The SYS1.PARMLIB BPXPRMxx member should have an entry such as:

```
FILESYSTYPE TYPE(ZFS) ENTRYPOINT(IOEFSCM) ASNAME(MVSZFS)
```

A typical BPXPRMxx mount setup might use the following example statements:

```
MOUNT FILESYSTEM('CAI.DIR.ZFS')
MOUNTPOINT('/cai')
TYPE(ZFS) MODE(RDWR)
/* */
MOUNT FILESYSTEM('CAI.CCSR14.SP00.NSMEM.R0.CAIZFS')
MOUNTPOINT('/cai/nsmem')
TYPE(ZFS) MODE(READ)
/* */
MOUNT FILESYSTEM('CAI.CCSR14.SP00.NSMEM.Rw.CAIZFS')
MOUNTPOINT('/cai/nsmem/Rw')
TYPE(ZFS) MODE(RDWR)
/* */
```

If you want the zFS to grow automatically, add the following PARM to the MOUNT directive:

```
PARM('AGGRGROW')
```

Note the following in this example:

- The /cai directory would be added to the root directory.
- During the install, CAI.DIR.ZFS would be mounted. This is a small zFS just for directory entries.
- mkdir commands would be issued for creating the '/cai/nsmem'.
- During the install and the post-install fwsetup script execution, all zFS data sets would have to be mounted RDWR.

Repository

If you have a repository that has been used from a release of Event Management prior to r11 (EM r11 is first delivered with CCS r12), you must migrate the Datacom/TR r9 database to the CA Datacom/AD database used by Event Management r11.

SAMPJCL job D5IDBEXT extracts the Event Management r3.0 Datacom/TR r9 database records and creates files used in loading the Event Management r11 Datacom/AD database.

SAMPJCL job D5IDBBLD uses the files created by D5IDBEXT and builds the Datacom/AD database used with Event Management r11. The D5IDBBLD job must be altered to point to the high level directory for Event Management r3.0. You must ensure that the UNIX system variable CA_DBHLQ is set in /cai/nsmem/PROFILE to the value set to ADHLQ in SAMPJCL file CCSVARS.

Maintenance Considerations

Event Management r11 is installed into the following different executable code locations:

- The traditional CNSMLOAD and CNSMPLD load libraries.
- Two separate zFS files mounted read-only or read-write as follows:
 - Read-only file that contains the executables, scripts, HTML, and Java class files.
 - Read-write mounted file that contains the system specific information such as Event Management log files for that system.

When a new service pack becomes available, to perform maintenance without disrupting the running system

1. Create a copy of the production read-only zFS mounted read-write on a service directory.

Note: Ensure the SMP/E environment uses DDDEFs for the service mount point; otherwise the programs will be linked into the wrong libraries. This holds true for the non-zFS DDDEFs as well.
2. Once the DDDEFs are correct, run the pre-maintenance script (EMPREMT) to save your customizations. APPLY maintenance as usual.
3. Run the post-maintenance script (EMPOSTMT) to restore the customizations.
4. Once the EMPOSTMT script has completed, you should be able to:
 - Shut down the running product
 - Swap zFS files
 - Update your BPXPRMxx member to use the new zFS file for the next IPL

Implementation Considerations

Event Management on z/OS can be thought of as three processes:

caiopr

Creates the Unicenter console from the various message sources. It provides a consolidation of syslogd, Unicenter messages, messages generated using the EM API, and messages routed by remote applications.

ca_calendar

Determines if a specific calendar is active or inactive based on the current date and time. If a message action is assigned a calendar profile, then ca_calendar must be active for the action to be taken.

stardaemon

Transmits data from the z/OS machine to a remote NT machine. When trying to connect using the Windows GUI, this process is responsible for performing the logon.

The z/OS version does not provide components such as workload or security which are best left to existing technologies already present on the mainframe.

The remainder of this section describes some implementation considerations related to these processes.

Use Store and Forward (Optional)

The store and forward option lets you automatically forward messages to another host. If that host is unavailable, it can store the message until that host becomes available.

To enable this option after installation

1. Ensure that the `$CAIGLBL0000/opr/saf` directory exists. It should be a symbolic link to `../RW/saf`.
2. Create a file `oprsaf` in the `$CAIGLBL0000/opr/config/<node>` directory. The file can be empty. It will contain the word `<active>` when Store and Forward is activated.

These environment variables in file `/cai/nsmem/opr/scripts/envusr` will be set and exported when file `$CAIGLBL0000/opr/config/<node>/oprsaf` is available:

```
CA_OPR_SAF=Y
CA_OPR_SAF_ROOT=$CAIGLBL0000/opr/saf
```

The Store and Forward daemon is the oprsafd process. This program is located in the \$CAIGLBL0000/bin directory and is started by this command:

```
unicntrl start opr
```

When issuing a cawto command and implementing Store and Forward, the CA_OPR_SAF and CA_OPR_SAF_ROOT environment variables must be set within the issuer's environment.

Clean up the Log Files

Caiopr creates a new log file for each day it is in operation. Over time, the number of files can take up more storage than desired.

To specify the number of logs, you can set the environment variable CA_OPR_RETAIN_LOGS to the number of logs to keep. This helps keep storage use to a minimum.

To set the variable, edit the file /cai/nsmem/opr/scripts/envusr. The default setting is '0', which keeps all logs.

Running Without the Repository

Use of CA Datacom/AD is optional for Event Management r11 but certain options do require the use of the CA Datacom/AD repository. For Event Management, CA Datacom/AD is required with the Calendars or Message Actions option. If you will use Calendars or Message Actions, you must set the environment variable CA_OPR_ZOSDB to 'Y' or 'YES' in the file PROFILE found in directory /cai/nsmem. This should have been done with the D5I10050 SAMPJCL member. If it was not, the environment variable can be set at any time before starting Event Management.

You do not need to run with the repository under the following conditions:

- If you are only using the built-in integration of CA OPS/MVS® Event Management & Automation to handle any messages passed to USS or Event Management from other platforms.
- If you do not want to create message records or actions within Event Management.
- If you do not want to use the Calendar feature of Event Management.

If the repository is not being used, the `ca_calendar` process will fail to start. This is valid since calendars are only used when messages are defined in the repository. To prevent error messages related to this process or to stop any calls to the repository, set the environment variable `CA_OPR_ZOSDB` in file `PROFILE` found in directory `/cai/nsmem` to the value of 'N' or 'NO'. This will stop all access to the Datacom/AD repository.

Without CA Datacom/AD active, when Event Management starts and the system variable `CA_OPR_ZOSDB` is set to 'Y' or 'YES', you receive the following error messages:

```
.CAOP_F_403 Database OPEN failed. tmsDBrc=0804802c.
.CAOP_F_415 tmsDBrc=0804802c.
.CAI_F_DB_005 Connect Failed, DB RC = 655360
```

Despite these errors, `caiopr` will initialize.

Running With the Repository

If you are using Calendars or Message Actions, a CA Datacom/AD repository and MUF are required. If `CAIENF` is setup to record events, then the same MUF that is used by `CAIENF` should be used for Event Management. This is true regardless of whether `ENFIMUF` or `ENFXMUF` is being used. For more information about how `CAIENF` uses CA Datacom/AD, see the CA `CAIENF` Best Practices chapter in this guide.

You must set the environment variable `CA_OPR_ZOSDB` to 'Y' or 'YES' in the file `PROFILE` found in directory `/cai/nsmem`. This should have been done with the `D5I10050 SAMPJCL` member but can be set at any time before starting Event Management.

To ensure the best performance when running Event Management with the repository, two steps should be completed before starting Event Management.

- Update the MUF `DATAPPOOL` parameter. Change the CA Datacom/AD startup and tuning parameter member for the MUF by altering the `DATAPPOOL` to include a set of five hundred 12K data buffers.

Depending upon how you are running the MUF, refer to the `CAIENF` or CA Datacom/AD MUF startup procedure JCL `SYSIN DD` statement to determine the data set and member name containing the startup and tuning parameters.

```
DATAPPOOL 8K,2000,12K,500
DATA BUFFER SIZE, # OF BUFFERS
```

This generates the number of buffers with the proper buffer size to ensure the best performance of Event Management with CA Datacom/AD.

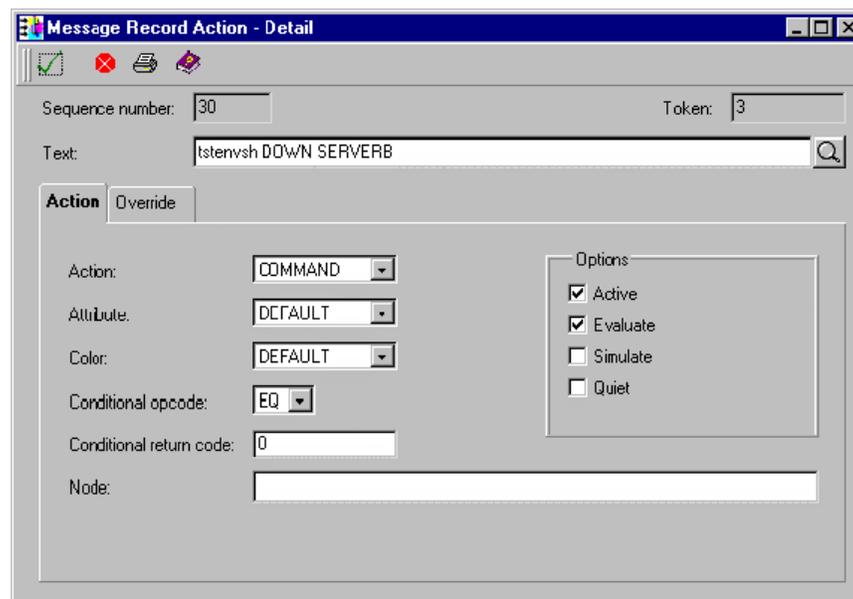
- Assemble and Link `DBMSTLST` into the CA Datacom/AD `CUSLIB`. `SAMPJCL` job `D5IDBMST` will accomplish this. This will ensure that enough memory is set aside for the largest data buffers.

Message Actions

The Message Action Detail action of COMMAND can be used to run a program or shell script. To perform this type of processing, you can execute the script `tstenvsh` as follows using the action COMMAND to test the contents of a previously set variable.

```
#
# Args: 1 - The value to be tested for
# 2 - The variable to test
#
# Results: Equal - return 0.
# False - return 1.
#
#
var='$'\`echo $2`
# Test if the variable has been defined first
if [ ! `eval echo $var` ]; then exit 1
fi
# It has been defined, test the value
if [ "$1" = `eval echo $var` ]; then
    exit 0
fi
# Not EQUAL exit false
exit 1
```

Following is an example of the Message Record Action-Detail screen defined to test if a variable `SERVERB` has the value of `DOWN`.



Write Messages Using `cawto`

You can use the `cawto` utility to write messages to the local Event Management console or to a remote machine. You can also assign different attributes to the messages to make them more meaningful. The basic syntax of the command is as follows:

```
cawto [-attribute value] <message>
```

-a

Designates one of (DEFAULT, BLINK, REVERSE)

-c

Designates the color (DEFAULT(White), Red, Orange, Yellow, Green, Blue, Pink, Purple).

-g

Designates the category (user defined).

-k

Indicates to put the message in the held message area.

-n

Designates the node name of the machine. If omitted, it defaults to the local machine.

-s

Identifies the user defined source to the message. This is useful for message matching.

For example, to call `cawto` to the local machine, specify:

```
cawto This is my message
```

To call `cawto` to a remote machine, specify:

```
cawto -n node This is my message
```

The maximum message length has been increased to 2048 bytes.

Reply to Messages Using cawtor

After you send a message, you can request a reply using the cawtor utility. This utility works similar to a WTOR message on the system console in that a reply ID is assigned to the message and the reply can be relayed back to the caller.

For example, assume you have a process that requires an action to be taken on a remote machine and you need to wait for that action to be completed before continuing. Using cawtor, the processing will hold until a response to the message is received. The response can be manual or automated on the remote machine.

Issue Console Commands Using CAconsole

You can use CAconsole to issue console commands from within the UNIX environment. This allows a message action to issue any required console commands directly without having to get automation into the process. The basic syntax of the command is as follows:

```
CAconsole [-attribute]
```

-v

Indicates to return any output.

-t

Indicates the amount of time to wait for response.

-i

Indicates the text of the command.

-x

Indicates to translate the command to upper case.

For example:

```
CAconsole -t5 -v -i"D OMVS,0"
```

In the following example, first determine if any job in a set of jobs is running. If any of the jobs is running, send an alert to a remote NSM machine and then cancel the job.

```
cd /cai/nsmem
# Set environment variables
. PROFILE
# Issue a Display Job,list command and place the output into a file display.out
CAconsole -I"D J,L" -V >display.out
# Search the output file for any job PRRP913x and place any hits into display1.out
grep PRRP913 display.out > display1.out
# If none are found send a message.
```

```

if [ $? = 1 ];then
  cawto -n NSMI Job PRRP913 is not present
else
  cawto -n NSMI Job is present, will cancel
# We need to parse the output to capture the job information.
  cat display1.out | while read v1 v2 v3 v4 v5 v6 v7 v8 v9 vx ;
  do
    echo "$v1\n $v2\n $v3\n $v4\n $v5\n $v6\n $v7\n $v8\n $v9\n" >display2.out
    job=`grep PRRP913 display2.out`
    echo $job
    job=`grep PRRP913 display2.out`
    echo $job
    CAconsole -I"CANCEL $job"
  done
fi

```

Route z/OS Console Messages to Event Management

At times, it is useful to obtain messages from the system console to be processed under Event Management or CA NSM. Release 3.0 introduced a method that incorporates a MPF exit on the mainframe to trap the messages and forward them to the Framework Event Management Services (FEMS). With CA Common Services for z/OS r12 and above, the functionality of CAS9FEMS has been expanded to handle multi-line WTO messages.

Once the message is in FEMS, you have the option of how and where to treat the message.

To trap messages on the mainframe and forward them to FEMS

1. Test the FEMS server.
 - Ensure that CAICCI is active on the system.
 - Create the DEST and CRIT DD files as documented in the Administrator Guide.
 - Update CNSMJCL member CAS9FEMS to match your installation standards.
 - Use the parms `-t -v` to run a test in verbose mode.
2. Start the FEMS server. After the server has been tested successfully it should be converted into an STC. Remember to use a user ID that has a valid OMVS segment in the security record.

3. Activate the MPF exit. Update the MPFLSTxx member to include the desired messages. The message should include the USEREXIT(CATNMPEX) operand to call the exit to process the message. For example, to trap message IEF450I you would code:

```
"IEF450I,SUP(NO),USEREXIT(CATNMPEX)"
```

4. Update the CRIT and DEST files to handle the additional messages. For example, to capture an IEF450 message, add the following to the CRIT file and a corresponding line:

```
CON11: text("^IEF450") lang(*) sev(*)
```

Add the following to the DEST file to send it to Event Management on the local box:

```
CON11: driver(/cai/nsmem/bin/casyfem1)
```

5. Test the exit. Create the messages on the console and verify that they reach the desired destination.

This process can be used to open a help desk issue based on console messages. Once this process is working to your satisfaction, you can forward the desired messages to Event Management, where a CA Service Desk issue can be opened as documented in the CA Service Desk documentation.

SNMP Traps

CA NSM has the ability to send and receive SNMP traps. Any received traps will appear on the Event Management console.

To receive traps, the daemon catrapd must be running. By default, the process listens on port 161. Your TCP/IP procedure has a PROFILE DD that contains reserved port numbers.

If you are not currently using this port, to edit your PROFILE's PORT

1. Locate the following line:

```
161 UDP SNMPQE
```

2. Change this line to:

```
; 161 UDP SNMPQE
```

If port 161 is unavailable, you must select the port you want and update file `/cai/nsmem/snmp/scripts/envset` as follows:

1. Locate the following lines:

```
# port number to listen on
CAICATD0001=161
export CAICATD0001
```

2. Change these lines as follows:

```
# port number to listen on
CAICATD0001=9161
export CAICATD0001
```

You can use the `catrap` program to send a trap. The syntax for the command is described in the *Reference Guide*.

Startup Procedures

The CAIPROC member NSMEMSTR is used to start Event Management. By default, this starts processes `caiopr`, `stardaemon`, `ca_calendar`, `newdaylog`, and `caidoc`.

Note the following startup considerations:

- All processes must be started with UID(0), and CAICCI must be active before they are started. To ensure that CAICCI is completely initialized, you can start Event Management as part of the CAIENF autocmds.
- The `stardaemon` does not require any additional services.
- The `caiopr` daemon has the option of using the repository. If the definition of message actions and calendars, or both, are desired, the repository is needed, and must be started prior to Event Management. You must ensure that the STEPLIB environment variable, in the PROFILE file, is set correctly prior to starting `caiopr`.

- If you are using CA OPS/MVS to handle the processing of messages, the repository is not required. The startup of caiopr will receive some warning messages concerning the connection to the repository and that the messages cannot be reloaded, but caiopr will continue processing normally and the messages can be ignored.
- The user ID assigned to Event Management must be UID(0). This is required to be able to switch a user from the Event Management user ID to the user ID of the client signed on. This prevents the client from being able to issue commands they would normally not be allowed to enter for the Unicenter console.
- If the SNMP trap listener, catrapd, needs to be started, you must locate the following line in the /cai/nsmem/opr/scripts/emstart file and remove the pound sign (#) from the beginning of the line:

```
#unicntrl start snmp
```

When finished, the line should appear as follows:

```
unicntrl start snmp
```

Java GUI

This section describes Java GUI considerations.

Timeout Settings

The following settings control how long the GUI waits for a response:

```
TIMEOUT=300
```

Instructs CAICCI to wait 300 seconds. This value is set near the end of the /cai/nsmem/browser/scripts/w2startup script file.

```
persistentservertimeout
```

A registry setting that is set by using the Java TIMEOUT command. The command by itself displays the number of seconds the GUI waits for a response. Passing a parameter changes the setting to whatever is passed.

If you consistently get timeout messages, you should increase these settings by updating the TIMEOUT parm to a higher value.

Security Requirements

Built-in security checks protect the defined resources from being modified by unknown sources. You must define these resources to the security system to protect them. All the resources are defined to a single class CAIUNI. The following describes each resource:

EMSRVC.MSGRECORD

Controls access to define/modify message records.

EMSRVC.MSGACTION

Controls access to define/modify message action records.

EMSRVC.CALENDAR

Controls access to define/modify calendars.

EMSRVC.CONLOG

Controls access to the Unicenter Console log.

EMSRVC.ANNOTATION

Controls access to the Unicenter Console log annotation feature.

To allow access to any of these resources, READ access is required.

Enterprise Management

The Enterprise Management icon displays all the machines that can be found running Unicenter or CCS.

- Run a CAICCI inquiry to find the receiver EMSRVC_ROUTER_U.
- For each machine that is found, an attempt to obtain the APPMAP is made. If the map cannot be loaded, the machine is left off the list.

To limit the list to only a selected few, you can create a file nodelist in the /cai/nsmem/emsvrc/data directory. A nodelist.sample file can be used as an example. To select a node to be on the list, enter the CAICCI SYSID for that machine. The local machine also needs to be in the list if desired.

Web Server Configuration

The installation process creates a configuration file containing all the information required to run the Java GUI. However, there are many options that affect the behavior of the web server; we discuss only some of them here.

For more information about web servers, see the IBM manual *IBM HTTP Server* for your operating system release.

If a web server is already running, you can support multiple applications from the one server or change the port number for one of them.

To change the port number, edit the configuration file and modify the Port operand.

To merge CCS into an existing web server, you must add the following to your configuration file:

```
ServerRoot      /cai/nsmem/browser this should contain the install path for NSM
HostNameMYHOSTSERVER the host name of your machine or
151.212.166.42 this should be the IP address of your machine
Port            80 The port to connect to: Default 80.
#
# The default EM configuration specifies no security inside the
# Web server since authentication is performed in our Java server once
# the user connects to the system. The NOSEC definition (below) allows
# Access to occur under the web server's security context (as opposed
# to under any specific individual's context).
#
Protection NOSEC {
ServerIdTNGFW_Server
  AuthType      Basic
                PasswdFile    %%SAF%%
                UserID        %%SERVER%%
                Mask          Anonymous
}
Protect        /scripts/*          NOSEC %%SERVER%%
Protect        /tngfw/scripts/*NOSEC %%SERVER%%
Protect        /tngfw/*            NOSEC %%SERVER%%
Protect        /tng/*              NOSEC %%SERVER%%
Protect        /browser/*          NOSEC %%SERVER%%
```

```
#
# The following directives specify the location of the Framework
# directories. If you are integrating Framework into an existing
# web server, these statements must be included in your existing
# HTTPD configuration file.
#
Exec      /scripts/*           /cai/nsmem/browser/scripts/*
Exec      /tngfw/scripts/* /cai/nsmem/browser/scripts/*
Pass      /tngfw/*         /cai/nsmem/browser/*
Pass      /tng/*           /cai/nsmem/browser/*
Pass      /browser/*      /cai/nsmem/browser/*
Pass      /*               /cai/nsmem/browser/*
Pass      *                /cai/nsmem/browser/*
#
Logging:
#      -- Uncomment the following lines to enable logging --
# AgentLog    logs/Agent
# AccessLog   logs/httpd-log
# CgiErrorLog logs/cgi-errors
# ErrorLog    logs/httpd-errors
# TraceLog    logs/jttrace
```

You can find these statements in file `/cai/nsmem/browser/httpd.conf`. You can copy and paste them into your existing web server configuration file.

Repository Maintenance

Event Management uses the CA Datacom/AD database. Prior to Event Management installation, you can use SMP/E to apply any required maintenance to CA Datacom/AD.

Required Tables and Initialization Records

All the required tables and initialization records are installed into the database files as part of the installation. The only records that are kept in the database are the Event Messages and Actions that you define, and the calendars you create. The actual messages on the console are kept in a separate flat file.

Create a Backup

As part of routine processing, you should take a backup of the repository periodically in the event of any disk corruption.

To create a backup of your repository, you can use SAMPJCL member D5IDBBAK. This job backs up all the relevant databases for Event Management.

In the event of any corruption or to create a new database, you can use the backup version to load in the desired entries. To reload the backup, use SAMPJCL member D5IDBRST.

Clone your Current Database

To clone your current database on another system

1. Back up the Event Management CA Datacom/AD database you wish to clone using the SAMPJCL member D5IDBBAK.

Note: This job can run anytime after the CA Datacom/AD customization for Event Management has been completed to back up the database records added after the initialization.

2. Set up the CA Datacom/AD environment on the target system by performing the procedure to customize the CA Datacom/AD for Event Management.
3. Use the SAMPJCL member D5IDBRST to restore the backup file created in step 1 into the new Event Management CA Datacom/AD database.

Use a Central Database on Multiple Systems

To use a central database on multiple systems, the environment variable CAI_OPR_REMOTEDB=ccisysid should be exported on each remote system. Add this export statement to the PROFILE file on each remote system.

This causes Event Management code to send message record queries to the database engine on the ccisysid node and receive a response from that node. The multiple systems involved can be z/OS systems running CA Datacom/AD or they can be distributed systems running CA NSM 3.0 or higher. For example, Event Management running on z/OS could use a distributed Microsoft SQL server database as its message record repository.

Event Management Examples

This section provides Event Management examples.

Call cawto from a Batch Program

You can call cawto using the BPXBATCH utility from IBM using either of the following methods:

- Directly in the PARM
- Using a script that calls cawto

The STDENV DD is used to set environment variables. The use of '\$varbname' is not supported in this type of DD so the STDENV file must contain hard coded PATH, LIBPATH, and so on.

Call cawto in the PARM Statement

Here is an example of calling cawto in the PARM statement:

```
//EXPAND EXEC PGM=BPXBATCH,REGION=0M,
// PARM='PGM /cai/nsmem/bin/cawto Hi from Batch'
//STDOUT      DD PATH='/cai/nsmem/tmp/batch.out',
//            PATHOPTS=(OwRONLY,OCREAT,OTRUNC),
//            PATHMODE=(SIRUSR,SIWUSR,SIRGRP,SIWGRP,SIROTH)
//STDERR      DD PATH='/cai/nsmem/tmp/batch.err',
//            PATHOPTS=(OwRONLY,OCREAT,OTRUNC),
//            PATHMODE=(SIRUSR,SIWUSR,SIRGRP,SIWGRP,SIROTH)
//STDENV      DD PATH='/cai/nsmem/tmp/ENVFILE',
//            PATHOPTS=(ORDONLY)
/*
/**
```

The ENVFILE could appear as follows and include any other variables that may be required:

```
PATH=/cai/nsmem/bin:/usr/bin:/usr/sbin
LIBPATH=/cai/nsmem/lib:/usr/lib
```

Call cawto from a Script

Here is an example of calling cawto from within a script:

```
//EXPAND EXEC PGM=BPXBATCH,REGION=0M,  
// PARM='PGM /cai/nsmem/bin/myscript Hi from a script file'  
//STDOUT      DD PATH='/cai/nsmem/tmp/batch.out',  
//            PATHOPTS=(OWRONLY,OCREAT,OTRUNC),  
//            PATHMODE=(SIRUSR,SIWUSR,SIRGRP,SIWGRP,SIROTH)  
//STDERR      DD PATH='/cai/nsmem/tmp/batch.err',  
//            PATHOPTS=(OWRONLY,OCREAT,OTRUNC),  
//            PATHMODE=(SIRUSR,SIWUSR,SIRGRP,SIWGRP,SIROTH)  
//STDENV      DD PATH='/cai/nsmem/tmp/ENVFILE',  
//            PATHOPTS=(ORDONLY)  
/*  
/*
```

The file myscript could do the following:

```
#Change to our directory  
cd /cai/nsmem  
# Set the environment variables  
. PROFILE  
# Take the parm passed as the message to be sent  
cawto $*  
# Send a copy of the command to the console  
logger message $* sent
```

Integration with CA OPS/MVS

This section provides examples of CA OPS/MVS rules for interfacing with Event Management.

Write a Message to a Remote Node

Following is an example of writing a message to a remote node. In this example, the message is prefixed with HELPDESK. This could be used as a keyword on the remote machine to take an action, such as opening a help desk issue based on the information passed in the message.

```
)CMD USSWT061
)PROC
Address USS
"WTO ",
"TEXT('HELPDESK Open Help Desk issue PAYROLL processing abended') ",
      " node(helpdesk) color(orange) attribute(reverse)"
return accept
```

Write a Message to a Remote CA NSM Machine with an Active Web Server

Following is an example of writing a message to a remote CA NSM machine with an active web server:

```
)USS    IMW3536I
)PROC
/* Inform a remote CA NSM machine that an IBM HTTPD server is active */
      MSGTXT = 'OPSAUTO1 IBM HTTPD server initialized at ' TIME()
      Address USS
      "WTO TEXT('"MSGTTEXT"') NODE(NSMNODE) "
end
return "Normal"
```

Issue a Command on a Remote CA NSM Machine

Following is an example of issuing a command on a remote CA NSM machine:

```
)USS    SENDCMD
)PROC
/* Execute a command on a remote machine (Unix, z/OS, or NT) */
      CMD = 'backup logfile '
      Address USS
      "CMD COMMAND('"CMD"') COLOR(blue) NODE(AIX1)"
end
return "Normal"
```

Troubleshooting

Due to the nature of running in the USS environment, the debugging of problems can be quite different than running under pure z/OS on the mainframe.

Check Job Output

The EXEC program BPXBATCH uses STDOUT and STDERR for output. These files are kept on the zFS. The script or program that is executed can be on the PARM statement or in the STDIN DD statement.

An CNSMJCL member NSMEMCHK has been provided to check some of the main sources of problems. Run the job and then verify that the output is correct.

The job checks things like the user ID that is being used to start the job, the amount of free space in /tmp, the results of certain TCP/IP functions, and so on.

Check Environment Variable Settings

A main cause of problems is the setting of environment variables. A couple of easy steps should easily determine if the settings are in place and correct.

To display current environment variable settings, issue the set command.

If the problem is seen when running a script, adding a 'set' command in the script displays the settings in STDOUT when the problem arises.

Check Space on the zFS File

Another common problem is running out of space in a zFS file. To determine how much space is left in a zFS, issue the following OMVS command to display the current amount of space left on all the file systems currently mounted:

```
df -k
```

To see the amount of space left for only the Event Management zFS file, issue the following OMVS command:

```
df -k /cai/nsmem/
```

Check Log Files

If Event Management has a problem writing to its log files, it writes a message to syslogd. Check your syslogd log file for any messages like this:

```
EDC5133I No space left on device
```

To direct these messages to the system console, add the following entry to your syslogd.conf file:

```
*.info /dev/console
```

Check Security Permissions

Inadequate security permissions can also cause problems during the installation process or during execution of the started tasks. See the product installation and maintenance documentation for security permissions required for the user ID used to install the product and for the user IDs assigned to the started tasks.

Check Tracing Variables

Tracing is usually turned on by setting environment variables before the process is started. The output is normally directed to STDOUT. The following variables are used to start tracing for caiopr. Additional environment variables for tracing other specific processing can be set at the direction of CA Support.

```
CA_CAIDDEBUG=Y  
CAI_NODENAME_DEBUG=Y
```

Some trace messages will be written based on the existence of file /cai/nsmem/emsvrc/config/traceinfo. That file contains control statements that direct the trace to a file, syslogd, or both. Send them to a file because it is easier to send the file for analysis.

The trace file that is written to is /cai/nsmem/emsvrc/config/debug.log.

Resolve Problems Accessing the CA Datacom/AD Database

CA Datacom/AD must be active before you try to connect to it. In order for the USS processes to connect to CA Datacom/AD, it uses modules that reside in a PDS or PDSE. To access those modules, environment variable STEPLIB points to these libraries. For this to work correctly, the variable must be set to the same STEPLIB that is used for the CA Datacom/AD JCL, and the user ID for the processes must have access to those data sets.

As part of the installation, you may have elected to update `/etc/profile` or to create a `tngprofile` file in your installation directory. Whichever method was used, check the file for the STEPLIB settings and ensure that they are correct. The other files to check are `/cai/nsmem/browser/httpd.envvars` and `/cai/nsmem/PROFILE`.

Note: Both methods are used to set STEPLIB for different processes.

Protect Event Management under zFS Security

For sites implementing zFS security using CA Top Secret Security or CA ACF2 Security, you might need to write rules to protect the resources. This section describes the resources that the various started tasks and the end users require to use Event Management.

Web Server Requirements

Web Server requires read access to:

- IBMFAC BPX.CAHSF.SET.RLIMIT and BPX.CAHSF.SET.PRIORITY
- HFSSEC /CAI.NSMEM.BROWSER
- HFSSEC /CAI.NSMEM.BROWSER.IMAGES
- HFSSEC /CAI.NSMEM.BROWSER.CLASSES
- HFSSEC /CAI.NSMEM.BROWSER.SCRIPTS
- HFSSEC /CAI.NSMEM.WV.HTML
- HFSSEC /CAI.NSMEM.RW.CONFIG
- HFSSEC /CAI.NSMEM.WV.SCRIPTS

Web Server requires update/alter access to:

- HFSSEC /CAI.NSMEM.RW.TMP

NSMJSERV (Java Server) Requirements

NSMJSERV (Java Server) requires read access to:

- HFSSEC /CAI.NSMEM.ATECH.SCRIPTS
- HFSSEC /CAI.NSMEM.BROWSER.SCRIPTS
- HFSSEC /CAI.NSMEM.CAL.SCRIPTS
- HFSSEC /CAI.NSMEM.EMSRVC.SCRIPTS
- HFSSEC /CAI.NSMEM.HELP.SCRIPTS
- HFSSEC /CAI.NSMEM.OPR.SCRIPTS
- HFSSEC /CAI.NSMEM.ROUTER.SCRIPTS
- HFSSEC /CAI.NSMEM.SECU.SCRIPTS
- HFSSEC /CAI.NSMEM.SNMP.SCRIPTS
- HFSSEC /CAI.NSMEM.STAR.SCRIPTS
- HFSSEC /CAI.NSMEM.COMMON.SRC
- HFSSEC /CAI.NSMEM.MESSAGES

NSMJSERV (Java server) requires exec access to:

- HFSSEC /CAI.NSMEM.BIN
- HFSSEC /CAI.NSMEM.LIB
- HFSSEC /CAI.NSMEM.EMSRVC.BIN
- HFSSEC /CAI.NSMEM.WV.SCRIPTS
- HFSSEC /CAI.NSMEM.WV.BIN

NSMJSERV (Java server) requires update/alter access to:

- HFSSEC /CAI.NSMEM.RW.TMP
- HFSSEC /CAI.NSMEM.RW.CONFIG
- HFSSEC /CAI.NSMEM.RW.BROWSER.CONFIG

NSMEMSTR Requirements

NSMEMSTR requires read access to:

- HFSSEC /CAI.NSMEM.SCRIPTS
- HFSSEC /CAI.NSMEM.ATECH.SCRIPTS
- HFSSEC /CAI.NSMEM.BROWSER.SCRIPTS
- HFSSEC /CAI.NSMEM.EMSRVC.SCRIPTS
- HFSSEC /CAI.NSMEM.HELP.SCRIPTS
- HFSSEC /CAI.NSMEM.OPR.SCRIPTS
- HFSSEC /CAI.NSMEM.ROUTER.SCRIPTS
- HFSSEC /CAI.NSMEM.SECU.SCRIPTS
- HFSSEC /CAI.NSMEM.SNMP.SCRIPTS
- HFSSEC /CAI.NSMEM.STAR.SCRIPTS
- HFSSEC /CAI.NSMEM.COMMON.SRC
- HFSSEC /CAI.NSMEM.CAL.SCRIPTS
- HFSSEC /CAI.NSMEM.MESSAGES

NSMEMSTR requires update/alter access to:

- HFSSEC /CAI.NSMEM.RW.TMP
- HFSSEC /CAI.NSMEM.RW.CONFIG
- HFSSEC /CAI.NSMEM.RW.LOGS

NSMEMSTR requires exec access to:

- HFSSEC /CAI.NSMEM.BIN
- HFSSEC /CAI.NSMEM.LIB
- HFSSEC /CAI.NSMEM.STAR.BIN

Java GUI Requirements

Use of the Java GUI requires read access to:

- HFSSEC /CAI.NSMEM.RW.CONFIG
- HFSSEC /CAI.NSMEM.RW.LOGS
- HFSSEC /CAI.NSMEM.RW.DATA
- HFSSEC /CAI.NSMEM.SECOPTS

Use of the Java GUI requires exec access to:

- HFSSEC /CAI.NSMEM.BIN
- HFSSEC /CAI.NSMEM.LIB

Use of the Java GUI requires update/alter access to:

- HFSSEC /CAI.NSMEM.RW.CONFIG.DEBUG\$LOG

Chapter 6: CAICCI Setup Examples

The following topics show examples of how to perform the CAICCI setup for specific results. You should consider the examples in this chapter before installing CAICCI.

Business Value

Applying these examples to your CAICCI installation will insure that CAICCI operates most efficiently and cost effectively.

For example, specifying which TCP/IP stack to bind to in a multiple TCP/IP environment, will insure proper routing and allocation of resources resulting in better system performance.

This section contains the following topics:

[Setup VTAM Parameters](#) (see page 69)

[Setup TCPIPGW Parameters](#) (see page 70)

[Setup Multiple TCP/IP Address Spaces](#) (see page 70)

[Setup Multiple CCITCP Address Spaces](#) (see page 71)

[Change the Characteristics of a Protocol](#) (see page 72)

[Setup XCF/XES Parameters](#) (see page 73)

[Setup CCISL, CCISL呢, and Certificate Deployment](#) (see page 73)

[Setup CCILGR/Assured Delivery](#) (see page 74)

Setup VTAM Parameters

Ensure that NODE and CONNECT are defined on both sides, and a reasonable retry time is used. Five minutes or less is typical. Use the defaults for MAXRU and START/STOP.

On SYSA:

```
SYSID(CCISYSA)
PROTOCOL(VTAM,VTAMSYSA,01,CCISYSA)
NODE(VTAM,VTAMSYSB,01,CCISYSB)
CONNECT(CCISYSB)
```

On SYSB:

```
SYSID(CCISYSB)
PROTOCOL(VTAM,VTAMSYSB,01,CCISYSB)
NODE(VTAM,VTAMSYSA,01,CCISYSA)
CONNECT(CCISYSA)
```

Setup TCPIP GW Parameters

When connecting two z/OS systems using TCP/IP, you should define the NODE and CONNECT statements on both systems.

On SYSA:

```
SYSID(CCISYSA)  
PROTOCOL(TCPIP GW, 7000, 01, CCISYSA)  
NODE(TCPIP GW, SYSB.NODE.COM: 7000, 01, CCISYSB)  
CONNECT(CCISYSB)
```

On SYSB:

```
SYSID(CCISYSB)  
PROTOCOL(TCPIP GW, 7000, 01, CCISYSB)  
NODE(TCPIP GW, SYSA.NODE.COM: 7000, 01, CCISYSA)  
CONNECT(CCISYSA)
```

When connecting a distributed (Windows/Unix/Linux) machine to z/OS, you should not define the NODE and CONNECT statements. This allows the connection to be driven only from the distributed platform.

On SYSA:

```
SYSID(CCISYSA)  
PROTOCOL(TCPIP GW, 7000, 01, CCISYSA)
```

To drive the connection from the Distributed CAICCI platform, you must define a REMOTE entry in the Distributed CCIRMTD file for the remote Mainframe CAICCI.

Setup Multiple TCP/IP Address Spaces

If your z/OS environment has multiple TCP/IP stacks and you want to ensure that CCITCP or CCITCP GW only binds to one of the stacks, you should use the PROTOCOL statement to define which TCP/IP stack to bind to. This is done by using *tcpip-job-name in the netdata parameter of the protocol. If your TCP/IP jobname is TCPIP1, your protocol statements should be coded as follows:

```
PROTOCOL(TCPIP GW, *TCPIP1: 7000, 01, CCISYSA)  
PROTOCOL(TCPIP, *TCPIP1: 1202)
```

Setup Multiple CCITCP Address Spaces

To run multiple CCITCP address spaces you should configure TCP/IP to specify the SHAREPORT option. Specifying this option configures TCP/IP to allow multiple listeners on the CAICCI port (port 1202). This parameter is set in your TCP/IP PROFILE member where you reserve PORTs. For example:

```
1202 TCP CCITCP SHAREPORT ; CAICCI servers
1202 TCP CCITCP2 SHAREPORT ; CAICCI servers
```

As client connection requests arrive for this port, TCP/IP distributes them across the members. TCP/IP selects the member with the fewest number of connections (both active and queued) at the time the request is received.

Then you can start multiple CCITCP address spaces manually as follows:

```
S CCITCP
S CCITCP.CCITCP2
```

Optionally, you may define a PROTOCOL using SPNPARMS that will start the second copy. For example:

```
TCPIP2          SERVICE  SERVER_NAME=MVS_START_SERVER,
                 MAX#_PROCESSES=1
                 PROCESS  PROCESS_TYPE=MVS_STC, PROCNAME=CCITCP,
                 JOBNAME=CCITCP2
```

By adding the following PROTOCOL statements to your CAICCI parameters, CAIENF will issue the start commands:

```
PROTOCOL (TCPIP, 1202)
PROTOCOL (TCPIP2, 1202)
```

This technique can also be used for TCPIPGW/TCPSSLGW protocols.

Note: CAICCI can support the execution of all PROTOCOLs simultaneously. However, any one system can only be connected using one of these protocols. Protocol establishment between connected systems is one to one, such as VTAM to VTAM. You cannot attempt to use a multi-protocol connection scheme between two systems. That is, SYSA cannot connect to SYSB over both the TCPIPGW and VTAM protocols simultaneously. Therefore, it is invalid to code NODE statements in SYSA for SYSB using TCPIPGW and VTAM at the same time. But you can define PROTOCOL statements for TCPIPGW and VTAM, then connect to SYSB using VTAM and SYSC using TCPIPGW.

For example, you would **not** code on SYSA:

```
SYSID(CCISYSA)
PROTOCOL(VTAM,VTAMSYSA,01,CCISYSA)
PROTOCOL(TCPIPGW,7000,01,CCISYSA)
NODE(VTAM,VTAMSYSB,01,CCISYSB)
NODE(TCPIPGW,7000,01,CCISYSB)
CONNECT(CCISYSB)
```

Rather, you **would** code on SYSA the following:

```
SYSID(CCISYSA)
PROTOCOL(VTAM,VTAMSYSA,01,CCISYSA)
PROTOCOL(TCPIPGW,7000,01,CCISYSA)
NODE(VTAM,VTAMSYSB,01,CCISYSB)
NODE(TCPIPGW,7000,01,CCISYSC)
CONNECT(CCISYSB)
CONNECT(CCISYSC)
```

Change the Characteristics of a Protocol

To change the name of the procedure started by CAICCI for a specific protocol, create an SPNPARMS member with the following parameters:

```
TCPIP  SERVICE  SERVER_NAME=MVS_START_SERVER,
          MAX#_PROCESSES=1

          PROCESS  PROCESS_TYPE=MVS_STC, PROCNAME=xxxxxxx,
          PARM=!

TCPIPGW  SERVICE  SERVER_NAME=MVS_START_SERVER,
          MAX#_PROCESSES=1

          PROCESS  PROCESS_TYPE=MVS_STC, PROCNAME=yyyyyyy,
          PARM=!
```

Where xxxxxxxx and yyyyyy are the procnames that you want to use.

Note: SPNPARMS is a DD in your CAIENF JCL. To implement a change or addition to this data set, restart CAIENF.

Setup XCF/XES Parameters

When using PROTOCOL XCF or XES, no NODE or CONNECT statements are needed for the systems you want to interconnect. The member name specified in the PROTOCOL statement must be the same for all systems involved.

On SYSA:

```
SYSID(CCISYSA)  
PROTOCOL(XCF,CCIXCF1)
```

On SYSB:

```
SYSID(CCISYSB)  
PROTOCOL(XCF,CCIXCF1)
```

Setup CCISSL, CCISSLGW, and Certificate Deployment

At a minimum, both parties in an SSL session require certificates that are signed by the same certificate authority, and installed in the z/OS security database. In the CCISSL & CCISSLGW procedures, it is required that at least the following keywords are defined:

```
CLAUTH=Y,  
CERT='certificate_name'  
KEYRING='keyring_name'
```

On distributed platforms, it is imperative that the local application be properly configured for SSL, and that an SSL path and Certificate Authority directory are available to the application.

For an initial setup of SSL, it is recommended to utilize the default certificates delivered with the CAICCI product in the CAW0OPTN data set. These would be the CCISSL client certificate named CCIP12, and the Certificate Authority certificate named CCIRTARM. For the distributed side, the downloadable executables named CCIPCS32 and CCIPCS64, also found in the CAW0OPTN data set, will extract CCI.PEM and CCIROOT.PEM which are the client and CA certificates, respectively.

Since the setup and deployment of SSL and certificates is an involved process, for more information see Tech Note TEC413258 titled 'CAICCI-SSL and External Security'. This Tech Note can be found on the CA Common Services for z/OS web page at the support.ca.com web site.

Setup CCILGR/Assured Delivery

The CCILGR/Assured Delivery component of CAICCI is required for sites that run ACF2/CPF. CCILGR is built upon the fundamental mechanisms of VSAM file access, and is a store and forward application. As such, data is received, written, and stored in the VSAM database then read before being sent out. When the data is sent it remains in the VSAM file until the local CCILGR receives confirmation from the data recipient that the data arrived intact. When confirmation is received, CCILGR deletes the record from the database. ACF2/CPF command propagation requires its own response to prevent duplicate data. Due to the constant reads, writes, and deletes, the CCILGR database is dynamic.

In order for Assured Delivery to guarantee that data is delivered and received securely, with integrity, and in the proper sequence, the product utilizes keyed sequential file access to VSAM. The keys are linked to the time the record was written, so every key is unique, and cannot be duplicated or reused. Because the access is keyed and sequential, inbound records are written to the end of the file and must be read again before being sent to their destination. Additionally, every read must begin at the top of the file and traverse all existing records until the one to be processed is encountered. Over time the database fragments with CA/CI splits, and if the amount of data being delivered within a sysplex of many systems is significantly large, there may be a performance impact.

One of the major reasons to reorganize the CCILGR VSAM file is when Control Areas become unusable. If a receiver becomes unavailable or is unable to keep up with the volume of messages, the messages will remain in the VSAM file for a long time, eventually causing Control Areas to fill up. When a control area becomes full, it is associated with a specific date and time range, and VSAM will not reuse that control area even after all the records in it have been delivered to the receiver and deleted from the file. The only way to reclaim those Control Areas is to reorganize the file.

With z/OS 1.12, a new VSAM option became available called CA Reclaim. With this option enabled, when a Control Area becomes empty, it will no longer be associated with a specific date and time range, and can be used to store newly created records. This will eliminate the need for frequent reorganizations. Even with this option enabled, the CCILGR VSAM file should still be reorganized periodically, perhaps weekly or monthly.

The CA Reclaim option is disabled by default on a system level, but it is enabled by default for all KSDSs, without having to redefine the data set. To turn on CA Reclaim at the system level, member IGDSMSxx in SYS1.PARMLIB must be updated to contain the new parameter "CA_RECLAIM(DATACLAS)". This will allow VSAM to reclaim newly emptied Control Areas in all VSAM files. See the *IBM z/OS MVS Initialization and Tuning Reference manual* for more information.

REORG the VSAM File

REORG the VSAM file on a regular basis as needed. A REORG reclaims CA/CI splits, maintains data integrity, and sustains consistent performance. A REORG every 24 hours is recommended, and can be automated. No data is lost because it is written to a backup file first, then written back to the VSAM file when the REORG is complete.

To prepare CCILGR for a manual or automated REORG of the VSAM database

1. In the CAIENF CCIPRMxx CAW0OPTN member, code the LOGGER statement as follows to specify a REORG when CCILGR is started.

```
LOGGER(020,012,012,Y)
```

When Y is specified in the 4th position, the VSAM database is REORGed when CCILGR starts up.

2. Insure the CCILGR procedure has a SYSUT1 DD statement as follows referencing the backup file.

```
//CCILGR  PROC  VSAM='CAI.AD.VSAM' , /* ASSURED DEL RESPOSITORY */
//          BACKUP='CAI.AD.BACKUP' /* ASSURED DEL BACKUP FILE */
//*
//*****
//*
//*  BACKUP FILE IS USED TO REORG THE VSAM FILE IF SPECIFIED ON THE *
//*  LOGGER STATEMENT IN CCIPARMS OR ENFPARMS AT START-UP.      *
//*
//*  I.E.  LOGGER(25,23,12,Y)                                     *
//*
//*  THIS QSAM FILE SHOULD BE ALLOCATED 20%-25% LARGER THAN THE *
//*  VSAM FILE.  FOR EXAMPLE, IF THE CLUSTER HAS BEEN ALLOCATED *
//*  WITH CYLINDERS(20,10) THEN THE BACKUP FILE SHOULD BE      *
//*  ALLOCATED WITH SPACE=(CYL,(25,10),RLSE).                  *
//*
//*  FOR EXAMPLE, AS A TEMP DATASET:                             *
//*
//*  //SYSUT1  DD SPACE=(CYL,(25,10),RLSE),                      *
//*  //          UNIT=SYSDA,                                     *
//*  //          DSN=&&TEMP,                                     *
//*  //          DISP=(,PASS)                                   *
//*
//*
//*  AS A PRE-ALLOCATED DATASET:                                 *
//*
//*  //BR14    EXEC PGM=IEFBR14                                   *
//*  //ALLOC   DD DSN=CAI.AD.BACKUP,                             *
//*  //          DISP=(NEW,CATLG,DELETE),                       *
//*  //          DCB=(RECFM=U,BLKSIZE=32760),                  *
//*  //          SPACE=(CYL,(25,10),RLSE)                      *
//*
```

```
//*
//*****
//*
//CCILGR EXEC PGM=CASNMLGR, TIME=1440, REGION=40M
//*
//STEPLIB DD DSN=CAI.CAILIB, DISP=SHR
//CAICCIAD DD DSN=&VSAM, DISP=SHR
//*
//SYSUT1 DD DSN=&BACKUP, DISP=OLD
//SYSPRINT DD SYSOUT=*
//*
//*
```

3. At a pre-determined time, issue the following command either from the console or using automation:

```
P CCILGR
```

This command stops CCILGR. The presence of the LOGGER statement causes CAIENF to automatically restart CCILGR. The Y parameter in the LOGGER statement causes the REORG to take place.

Important! Insure the SYSUT1 DD statement referencing a backup file is present in the CCILGR JCL. If there is no SYSUT1 DD statement, the REORG will not take place even if Y is specified in the logger statement.

Activate the Assured Delivery Feature

The LOGGER statement should adequately reflect that the CCILGR environment is deployed. For more information, see the *CA Common Services for z/OS Reference Guide*:

```
LOGGER(strings,buffer1,buffer2, reorg)
```

The LOGGER control option is used to activate the Assured Delivery feature of CAICCI and specify the number of VSAM strings and buffers that will be saved in the LOGGER database.

Parameters that must be entered for LOGGER are as follows:

strings

The number of VSAM strings. Calculate this number as follows: $2 \times \text{\#systems} + 5$

The default is 20 strings.

buffer1

The number of VSAM data buffers. Calculate this number as follows: $2 \times \text{\#systems} + 3$

The default is 12 data buffers.

buffer2

The number of VSAM index buffers. Calculate this number as follows: #systems but not < 12

The default is 12 index buffers.

reorg

Y or N. If Y is specified, LOGGER will reorganize the VSAM database at start-up of CCILGR.

The default is N.

Example (console)

```
CCI LOGGER(25,23,12,Y)
```

Example (ENFPARMS)

```
LOGGER(25,23,12,Y)
```

Example for a sysplex of 20 systems

```
LOGGER(45,43,20,Y)
```

Query the CCILGR

You should regularly and pro-actively query the state of CCILGR by using the MODIFY commands. For more information, see the Common Communications Interface chapter of the CA Common Services for *z/OS Administration Guide*.

The REPORT, STATUS, DBSTATUS, and DISPLAY commands are very useful and can be issued from the console at any time, or even automated.

The REPORT command writes to a dynamically allocated DD called ADREPORT. The STATUS, DBSTATUS, and DISPLAY commands write to SYSOUT.

REPORT example

The following command shows the current number of records in the VSAM file and who they are for:

F CCILGR,REPORT				
SYSID	APPLICATION ID	REC COUNT	AVG REC L	MAX REC L
MYPLEX	CA-ACF2/CPF-CMD	6499	178	178
TOTAL		6499	178	178

STATUS example

The following command shows the number of converses or sends attempted and completed from the local to the destination. The RECOVERY application ID is static, is always shown, and means the local system.

```
F CCILGR,STATUS
TASK#  APPLICATION          CATM  CCMP   SATM  SCMP 323
   02  RECOVERY
   03  #MYPLEX  CA-ACF2/CPF-CMD  5513  5513    0    0
CAS9899I: END OF COMMAND
```

DBSTATUS example

The following command provides the current statistics about the VSAM file:

```
F CCILGR,DBSTATUS
LOOK ASIDES  BUFF READS  MAX STRINGS  AVSPAC  EXTENTS
   50282     1568         2      88848K    25
CAS9899I: END OF COMMAND
```

DISPLAY example

The following command identifies the current VSAM file in use, and provides the space and extent statistics:

```
F CCILGR,DISPLAY
CAS9780I - CAICCI AD OPN(Y) DSN=CA90SMVS.CCILGR.LGR
CAS9781I - CAICCI AD AVAIL SPACE(00088864K) PERCENT FULL(001) EXTENTS(025)
CAS9899I: END OF COMMAND
```

Allocate the VSAM File

The VSAM file should be sufficiently allocated for the expected work load. For more information, see Assured Delivery in the *CA Common Services for z/OS Administration Guide*.

Insure that the PRIMARY and SECONDARY CYLINDER allocations are sufficient for the work load. The following is a sample allocation.

```
DEFINE CLUSTER -  
  NAME(xxxxxxxx.CCILGER) -  
  SHR(3,3)  
  RECORDSIZE(4096 33000) -  
  CYLINDERS(n,m) -  
  SPANNED -  
  REUSE -  
  KEYS(38 0)  
  VOL(vvvvvv,...) -  
  DATA -  
    (NAME(xxxxxxxx.CCILGER.DATA) -  
     CISZ(4096)) -  
  INDEX -  
    (NAME(xxxxxxxx.CCILGER.INDEX) -  
     CISZ(3584))
```


Index

A

- access.off file for Agent Technology security • 17
- ACID requirement for ENF • 37
- administrator account for Agent Technology • 17
- agent discovery problems • 28
- Agent Technology
 - AWADMIN • 17
 - AWGROUP, Agent Technology security group • 17
 - configuration • 21
 - performance • 25
 - services • 17
 - startup • 24
 - upgrading • 17
 - verify installation • 24
- Assured Delivery, activation • 76
- aws_orb • 17
- aws_sadmin • 17
- awservices • 17

B

- backing up the repository • 58
- business value
 - Agent Technology • 17
 - CCI • 69
 - Event Management • 39
 - Mainframe 2.0 • 11

C

- CA ACF2 security • 64
- CA NSM, remote commands • 61
- CA OPS/MVS, Event Management integration • 60
- CA/CI splits, reclaiming • 75
- CAconsole commands • 50
- calendar and message action options • 46
- CAS9DB equivalent functions for ENF • 35
- cawto
 - batch process • 59
 - PARM statement • 59
 - scripts • 60
 - writing messages • 49
- cawtor for replying to messages • 50
- CCI setup • 69
- CCI started procedures, naming • 72

- CCITCP, multiple address spaces • 71
- central database, using • 58
- cloning your database • 58
- code locations for Event Management • 44
- communication ports • 17
- console message, routing to Event Management • 51

D

- D5IDBEXT for migrating Event Management databases • 44
- database
 - access problems • 64
 - Event Management options • 46, 47
 - maintenance • 57
 - multiple systems • 58
- defined resources protection • 55
- discovering agents • 28

E

- ENF
 - database querying • 37
 - file requirements • 34
 - Health Check • 38
 - installation considerations • 33
 - multiple systems • 37
 - upgrading • 35
- environment variables • 62
- Event Management
 - implementation • 45
 - installation verification • 39
 - log files • 63
 - maintainance • 44
 - processes • 45
 - startup • 53
 - structuring zFS • 42
 - timeouts • 54
 - verify installation • 39

H

- health check ENF • 38

I

- initialization records location • 57
- interconnecting XCF or XES • 73

J

Java GUI configuration • 56

M

mainframe agent installation • 22

management nodes, finding • 55

messages

- automatically forwarding in Event Management • 45

- remote CA NSM • 61

- remote node • 61

migrating Datacom/TR to Datacom/AD • 44

N

NSMEMSTR read access • 66

NSMJSERV read access • 65

R

remote node messages • 61

REORG • 75

requirements

- CA Common Services • 17

- Java GUI • 67

- NSMEMSTR • 66

resources, locating • 55

S

screen parm to health check ENF • 38

security

- permissions • 63

- resource definition • 55

service packs for Event Management • 44

SHAREPORT option • 71

shell scripts for message actions • 48

SMP/E procedures • 17

SNMP traps for Event Management • 52

SQL jobs on the ENF database • 37

SSL certificates • 73

store and forward for Event Management, enablement • 45

system requirements • 17

T

table location • 57

TCP/IP

- binding stacks • 70

- multiple stacks • 70

- requirements • 17

- TCPIP.ETC.SERVICES file • 17

- TCPIP.GW, parameters • 70

- timeouts for Event Management • 54

- Top Secret security • 63, 64

- trace messages • 63

- tracing variables • 63

U

UNIX environment console commands • 50

UNIX System Services requirements • 17

V

VTAM, parameters • 69

W

web server configuration • 56

web server, read access requirements • 64

Z

zFS

- Agent Technology • 17

- Event Management • 42

- space • 62