

z/OS



Common Information Model User's Guide

Version 1 Release 10

z/OS



Common Information Model User's Guide

Version 1 Release 10

Note

Before using this information and the product it supports, be sure to read the general information under “Notices” on page 209.

This edition applies to Version 1 Release 10 of z/OS (5694-A01) and to all subsequent releases and modifications until otherwise indicated in new editions.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address below.

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

IBM Deutschland Research & Development GmbH
Department 3248
Schönaicher Str. 220
D-71032 Böblingen
Federal Republic of Germany

Or by FAX:

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

Or electronically:

Internet e-mail: s390id@de.ibm.com

If you would like a reply, be sure to include your name, address, telephone number, or FAX number.

Make sure to include the following in your comment or note:

- Title and order number of this book
- Page number or topic related to your comment

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

© Copyright International Business Machines Corporation 2005, 2009.

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

Contents

Figures	v
--------------------------	----------

Tables	vii
-------------------------	------------

About this publication	ix
---	-----------

Who should use this document	ix
Where to find more information	ix
Legend for graphics showing class structures	x
How to read syntax diagrams	x

Summary of changes	xiii
-------------------------------------	-------------

What's new in z/OS V1R10.	xiii
Support of CIM Schema version 2.13	xiii
Running providers in a designated user context	xiii
Audit logging with SMF records	xiii
MODIFY console command.	xiii
CIM client for Java Version 2	xiii
Application ID CFZAPPL for authentication	xiii
New and changed z/OS-specific messages.	xiv

Chapter 1. Introduction and concepts	1
---	----------

Support of CIM indications	4
How indications work	5
CIM indications	6
CIM indication class hierarchy	7
CIM subscription mechanism.	8
CIM server security.	11

Chapter 2. Installation and migration	15
--	-----------

Software and hardware prerequisites	15
Installation	15
Installation directories	15
Customize CFZRCUST	16
Migration from z/OS 1.8 or z/OS 1.9 to z/OS 1.10	18

Chapter 3. First-time CIM server set up	21
--	-----------

Customizing the security for the CIM server	21
Defining a RACF class and profile for the CIM server	22
Defining a user ID that runs the CIM server	23
Configuring the CIM server's resource authorization model	23
Granting clients and administrators access to the CIM server	26
Switching identity (surrogate)	27
Configuring the CIM server HTTPS connection using AT-TLS.	27
Defining the CFZAPPL profile for the APPL class	33
Defining an application ID for PassTicket validation	33
Additional setup for specific providers	33
Customizing the CIM server startup	34
Customizing the started task procedure CFZCIM	34

Customizing the UNIX System Services shell for CIM server startup	34
Customizing environment variables	35
Selecting a WLM service class for z/OS CIM priority	36
Provider-based authorization model	36
Running providers in a designated user context	37

Chapter 4. How to work with the CIM server	39
---	-----------

Controlling the CIM server	39
Starting and stopping the CIM server.	39
Running providers in their own address space	40
CIM server configuration.	41
Changing current configuration properties	41
Changing planned configuration properties.	41
CIM server configuration properties	42
Configuring CIM server tracing	44
Configure logging	45
Audit logging with SMF record 86.	47
Backing up the CIM server configuration	48
Automatic restart of the CIM server	48
CIM server command-line utilities and commands	52
cimmof/cimmofl	53
cimconfig	55
cimprovider	57
cimcli	59
cimsub	69
MODIFY console command	72
Backing up the CIM server repository	73
Sample client applications	74
Troubleshooting	74
Problem determination	74
ASCII-EBCDIC conversion	75
Error messages	75

Chapter 5. z/OS Management Instrumentation for CIM	79
---	-----------

Supported CIM operations	82
OS management Base classes	83
CIM_ComputerSystem.	84
CIM_OperatingSystem.	85
CIM_RunningOS	86
CIM_Process	86
CIM_OSProcess	88
OS management BaseBoard class	89
IBM_BaseBoard	89
CIM_ComputerSystemPackage association	91
IBMzOS_CSBaseBoard.	91
OS management Processor class	91
CIM_SystemDevice association	92
CIM_Processor	92
IBMzOS_Processor	92
OS management Logical Disk class	93
CIM_LogicalDisk	94

OS management File System classes	96
CIM_HostedFileSystem	96
CIM_LocalFileSystem	96
IBMzOS_UnixLocalFileSystem	97
CIM_RemoteFileSystem	97
IBMzOS_NFS	97
OS management Network classes	98
CIM_EthernetPort	99
IBMzOS_EthernetPort	100
CIM_PortImplementsEndpoint	100
CIM_IPProtocolEndpoint	100
IBMzOS_IPProtocolEndpoint	100
CIM_SystemDevice association	101
OS management Job classes	101
IBMzOS_Job	101
IBMzOS_SysoutDataset	120
IBMzOS_Subsystem	123
IBMzOS_JobsManagementSettings	124
OS management Cluster classes	125
IBMzOS_Sysplex	125
IBMzOS_CouplingFacility	131
IBMzOS_CFStructure	134
IBMzOS_CFStructureConnector	141
IBMzOS_CouplingFunction	145
IBMzOS_CoupleDataset	148
IBMzOS_CFRMPolicy	151
IBMzOS_CFStrDependsOn association	152
IBMzOS_CollectionOfCFs association	153
IBMzOS_CollectionOfSysplexNodes association	153
IBMzOS_HostedCFStructures association	153
IBMzOS_HostedCFStructureConnectors association	154
IBMzOS_UsesCF association	154
IBMzOS_UsesCFRMCoupleDatasets association	154
IBMzOS_UsesCFRMPolicies association	154
IBMzOS_UsesCouplingFunctions association	155
IBMzOS_UsesSysplexCoupleDatasets association	155

Chapter 6. Messages and logs	157
Logging	157
CIM server messages	157
z/OS-specific messages	158

Appendix A. Sample CIM request and response	175
--	------------

Appendix B. Impacts of the Out-of-Process support for providers . 179	
Provider management and registration	179

Tracing providers running Out-of-Process	179
Performance implications	180

Appendix C. Developing CMPI providers for z/OS	181
Obtaining the required header files	182
General aspects for developing a provider	183
Provider initialization and function signatures	184
Instance provider functions	184
Method provider functions	185
Association provider functions	185
Indication provider functions	185
Security aspects	185
Data conversion ASCII / EBCDIC	186
Registering a provider with the CIM server	186
PG_Provider	187
PG_ProviderModule	188
PG_ProviderCapabilities	190
Provider registration processing	191
Samples	191

Appendix D. Cluster / CoupleDataset / JES2-JES3Jobs Provider Setup	193
Required PARMLIB updates	193
Required RACF setup	193
Sysplex couple dataset formatting	196

Appendix E. WLM classes	199
IBMzOS_WLM	199
Methods of IBMzOS_WLM	201
Indications for IBMzOS_WLM	201
IBMzOS_WLMOS association	202

Appendix F. Reason codes	203
---	------------

Accessibility	207
Using assistive technologies	207
Keyboard navigation of the user interface	207
z/OS information	207

Notices	209
Programming Interface Information	210
Trademarks	210
Terms and conditions for downloading and printing publications	211
Policy for unsupported hardware	211

Index	213
------------------------	------------

Figures

	1.	Sample network environment managed with	
		CIM management applications	1
	2.	Exemplary tasks of the CIM server in a z/OS	
		environment	3
	3.	CIM indication flow and processing.	5
	4.	CIM indication hierarchy	7
	5.	Indication subscription class diagram	9
	6.	Indication Handler	10
	7.	Security components	12
	8.	CIM Base classes extended by z/OS-specific	
		classes (1)	83
	9.	OS management BaseBoard Class	89
	10.	OS management Processor classes	91
	11.	CIM Base classes extended by z/OS-specific	
		classes (2)	94
	12.	OS management FileSystem classes	96
	13.	OS management Network classes	99
	14.	CMPI provider interfaces	181
	15.	OpenPegasus CVS Repository	183
	16.	CIM classes from the provider registration	
		schema	187

Tables

1. UML syntax.	x	8. Properties of class PG_Provider	188
2. Default installation directories for z/OS CIM	15	9. Properties of class PG_ProviderModule	188
3. Access types required for CIM operations	26	10. Properties of class PG_ProviderModule	190
4. CIM server basic configuration properties	42	11. Sample TSO CLIST	194
5. CIM server advanced configuration properties	43	12. Sample sysplex couple dataset formatting JCL	197
6. Log and syslog levels	46	13. Jobs providers' reason codes	203
7. UCB control block information	95		

About this publication

This document describes the implementation of the Common Information Model (CIM) and Web Base Enterprise Management (WBEM) standards for z/OS. It explains how to set up and use the CIM server and CIM resource instrumentation provided together with the z/OS operating system. CIM is a standard data model for describing and accessing systems management data in heterogeneous environments. It allows system administrators and vendors to write applications that measure system resources in a network with different operating systems and hardware.

The focus of this document is on the z/OS-specific implementation of CIM. For more detailed information on the CIM and WBEM standards please review the information provided by the Distributed Management Task Force (DMTF), which is found in the internet on the *DMTF* web site. (For explicit link addresses, see “Where to find more information.”)

Following are the major topics of this document:

- Chapter 3, “First-time CIM server set up,” on page 21 describes how to set up the CIM server for z/OS.
- Chapter 4, “How to work with the CIM server,” on page 39 describes how to customize and use the CIM server for z/OS.
- Chapter 5, “z/OS Management Instrumentation for CIM,” on page 79 describes the z/OS system resources that are exposed through CIM and z/OS-specific extensions of the CIM Schema. The focus of this chapter are the z/OS-specific supplements and differences as well as how to use this information on z/OS.

Who should use this document

This document is intended for the system programmer and performance analyst responsible for measuring and improving system performance. The CIM data model described in this document can be used in writing applications that monitor system resources, for example to discover bottlenecks or to control thresholds and to provide information on which tuning actions should be applied.

Administrators of the z/OS CIM runtime environment should read Chapter 3, “First-time CIM server set up,” on page 21, which contains information on how to set up security for the CIM server.

Developers of CIM providers for z/OS should read “Appendix C. Developing CMPI providers for z/OS” on page 181.

Because the CIM data model provided for z/OS is used in applications that measure z/OS system performance, this document assumes that the reader has extensive knowledge of the z/OS system.

Where to find more information

OpenPegasus web site
<http://www.openpegasus.org>

DMTF web site

<http://www.dmtf.org>

WBEM standards

<http://www.dmtf.org/standards/wbem>

Common Information Model (CIM) Standards

<http://www.dmtf.org/standards/cim>

CIM Event Model White Paper

<http://www.dmtf.org/standards/documents/CIM/DSP0107.pdf>

Specification for CIM Operations over HTTP

http://www.dmtf.org/standards/published_documents/DSP200.html

CIM Query Language Specification

<http://www.dmtf.org/standards/documents/WBEM/DSP0202.pdf>

@server Common Information Model

http://publib.boulder.ibm.com/infocenter/eserver/v1r1/en_US/info/ciminfo/eicah.pdf

z/OS information updates on the web

http://publibz.boulder.ibm.com/cgi-bin/bookmgr_OS390/BOOKS/ZIDOCMST/CCONTENTS

LookAt web site for online message explanations

<http://www.ibm.com/eserver/zseries/zos/bkserv/lookat/>

Legend for graphics showing class structures

The graphics in this book showing class structures illustrate the CIM object modeling using the UML syntax:

Table 1. UML syntax

Construct	Description	Syntax
association	A relationship between two or more classifiers that involves connections among their instances.	
aggregation	A special form of association that specifies a whole-part relationship between the aggregate (whole) and the component part.	
inheritance	A relationship among classes where one class shares the structure and/or behavior defined for one or more other classes. Inheritance is the mechanism that makes generalization, subclasses, and superclasses possible.	
class	Denotes the representation of a CIM class in UML notation with title, properties, and methods.	

How to read syntax diagrams

The following rules apply to the syntax diagrams used in this book:

Arrow symbols

Read the syntax diagrams from left to right, from top to bottom, following the path of the line.

The arrow symbols indicate the beginning and the end of a statement.



Conventions

- Variables appear in italics, for example, *column-name*. They represent user-defined parameters or suboptions.
- When entering commands, separate parameters and keywords by at least one blank if there is no intervening punctuation.
- Enter punctuation marks (slashes, commas, periods, parentheses, quotation marks, equal signs) and numbers exactly as given.

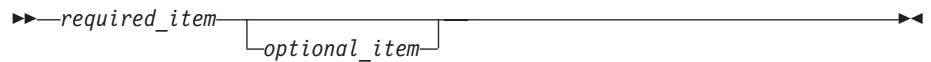
Required items

Required items appear on the horizontal line (the main path).



Optional items

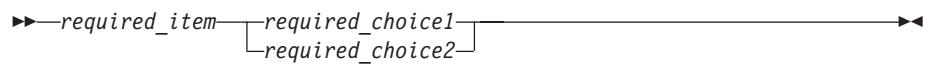
Optional items normally appear below the main path.



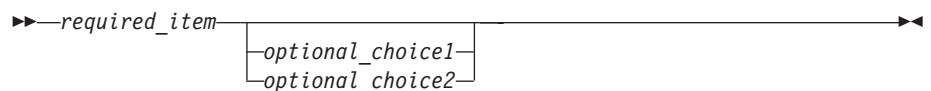
Multiple required or optional items

If you can choose from two or more items, they appear vertically in a stack.

If you *must* choose one of the items, one item of the stack appears on the main path.

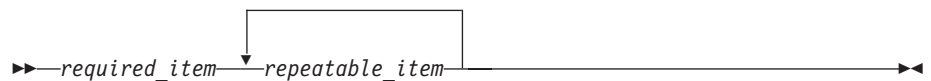


If choosing one of the items is optional, the entire stack appears below the main path.



Repeatable items

An arrow returning to the left, above the main line, indicates that an item can be repeated.



A repeat arrow above a stack indicates that you can repeat the items in the stack.

Keywords

- Keywords appear in uppercase (for example, FROM). They specify field names in the field table and must be substituted by the field ID when coded.
- Variables appear in all lowercase letters (for example, *column name*). They represent user-supplied names or values. Variables enclosed in quotes must be coded as character strings.
- If punctuation marks, parentheses, arithmetic operators, or other such symbols are shown, you must enter them as part of the syntax.

Summary of changes

This document contains terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations in the present release are indicated by a vertical line to the left of the change.

The following sections list the changes that have been made to CIM with various operating system releases.

What's new in z/OS V1R10

The following information describes the enhancements for the CIM server in z/OS Version 1 Release 10.

Support of CIM Schema version 2.13

The CIM server now supports CIM Schema version 2.13.

Running providers in a designated user context

Until now, providers were always processed in the context of the user ID of a request. The provider ran under the identity of the requestor's user ID, and resource access authorization occurred against this user ID. So the requestor was required to be authorized for all resources that a provider was accessing when it processed a request.

Starting with z/OS 1.10, you can process a provider alternatively in the context of a **designated user ID**. The designated user ID specifies a separate security context which is used to process the provider.

The properties *UserContext* and *DesignatedUserContext* of CIM class *PG_ProviderModule* specify the provider's processing context.

Audit logging with SMF records

The CIM server can file audit log records to SMF record 86. These records contain information about authentication, configuration, provider status, and CIM operations.

MODIFY console command

Starting with z/OS 1.10 the CIM server configuration can be changed from the z/OS system console using the MODIFY command.

CIM client for Java™ Version 2

Starting with z/OS 1.10, version 2 of the CIM client for Java is supported in addition to the version 1.3 CIM client for Java from the SBLIM project. One of the features provided by the CIM client for Java Version 2 is the support for embedded instances.

Application ID CFZAPPL for authentication

The CIM server is using the application ID CFZAPPL for authentication. This applies in using PassTickets and authorization check of the SAF APPL class.

New and changed z/OS-specific messages

The following z/OS-specific messages have been added or changed:

CEZ10000E	CFZ06201W	CFZ06205E	CFZ06206I	CFZ06207E	CFZ06208I
CFZ06209I	CFZ06210I	CFZ06211E	CFZ06212E	CFZ07806E	CFZ07807E
CFZ10037E	CFZ12526E	CFZ12527E	CFZ12528I	CFZ12529E	CFZ12568E
CFZ12569E	CFZ12570I	CFZ12571E	CFZ12572W	CFZ12573W	CFZ12574W
CFZ12575E	CFZ12576F	CFZ12577I	CFZ12578W	CFZ13007W	CFZ17200W
CFZ17805I	CFZ17806I	CFZ18202E	CFZ18204I	IWMCP001E	IWMCP002E
IWMCP003E	IWMCP004E	IWMCP005E	IWMCP006E		

The following z/OS-specific messages have been deleted:

CFZ06201I CFZ14208W

The following z/OS-specific messages have changed the message number:

from CFZ12547E to CFZ12547F	from CFZ12559E to CFZ12559F
from CFZ12567W to CFZ13006W	

Chapter 1. Introduction and concepts

The Common Information Model (CIM) is a standard data model developed by a consortium of major hardware and software vendors (including IBM) called the Distributed Management Task Force (DMTF) as part of the Web Based Enterprise Management (WBEM) initiative. WBEM includes a set of standards and technologies that provide management solutions for a distributed network environment. Interoperability is a major focus of WBEM, and using WBEM technologies can help you develop a single set of management applications for a diverse set of resources.

Figure 1 shows a sample environment in which management applications can run that use the DMTF CIM standard data model.

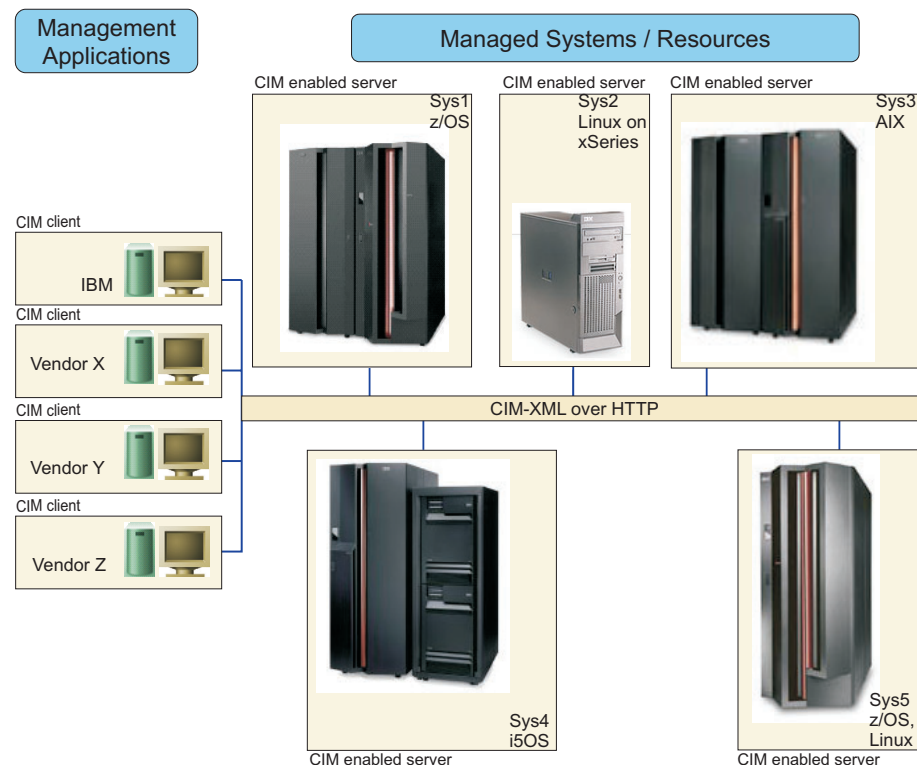


Figure 1. Sample network environment managed with CIM management applications

CIM is a major component of the WBEM initiative, providing a model for describing and accessing data across an enterprise. CIM consists of both a specification and a schema. The specification defines the details for integration with other management models, while the schema provides the actual model descriptions.

With support for the CIM server on systems running z/OS, users have the ability to access z/OS resources through an extendible industry standard model. The remainder of this document contains information on how to use the CIM server for z/OS for this purpose.

CIM for z/OS includes:

CIM server

An open source implementation of the CIM server manages communication between clients and providers. The CIM server also provides several management functions, including security, and a set of commands that provide configuration and management functions to administrators.

The CIM server implementation on z/OS is based on the **OpenPegasus CIM server 2.7** from **The Open Group**. See the *OpenPegasus web site* for more information (Copyright 2005, 2007. The Open Group. All Rights Reserved).

CIM-XML over HTTP protocol

The "CIM-XML over HTTP" protocol is an implementation of the standardized formats for communication between clients and the CIM server *Representation of CIM in XML* and *CIM Operations over HTTP*. For more information about these standards, see the *WBEM web site*.

CIM operations over HTTP

The CIM server for z/OS supports most of the CIM operations defined in the *CIM Operations over HTTP* specification by the DMTF.

DMTF CIM Schema

A CIM Schema defines an information model for representing systems management functions. For z/OS 1.10, CIM Schema version 2.13 is supported by the CIM server.

Instrumentation for server resources

Instrumentation for server resources on the system are called **providers**. The providers, which are based on a subset of the standardized CIM classes, gather data on a system. CIM clients can work with these data by accessing the providers through the CIM server. For more information about what is supported in z/OS, refer to Chapter 5, "z/OS Management Instrumentation for CIM," on page 79.

CIM client for Java

Since z/OS 1.9, the CIM client for Java library from the SBLIM project is included with z/OS CIM. For z/OS 1.10, version 2 of the CIM client for Java is included. The CIM client for Java is a programming API that enables z/OS applications written in Java for local and remote access of CIM instrumentation through the CIM-XML over HTTP access protocol. It consists of a Java library and associated online Java documentation.

Figure 2 on page 3 illustrates how the CIM server works in the z/OS environment: A CIM client application requests the CIM server to return information about z/OS resources, in this case about basic operating system (OS) data as well as RMF metrics. The CIM server invokes the according CIM providers which retrieve the requested data associated to z/OS system resources. The z/OS RMF monitoring provider invokes the RMF Distributed Data Server (DDS) which in turn collects RMF Monitor III performance data. The CIM server consolidates the data from the providers and returns them back to the calling client through the CIM-XML over HTTP protocol.

Figure 2 on page 3 shows two types of CIM providers: RMF monitoring providers that use the RMF DDS to access the z/OS system data, and z/OS OS management providers that access the z/OS system data directly.

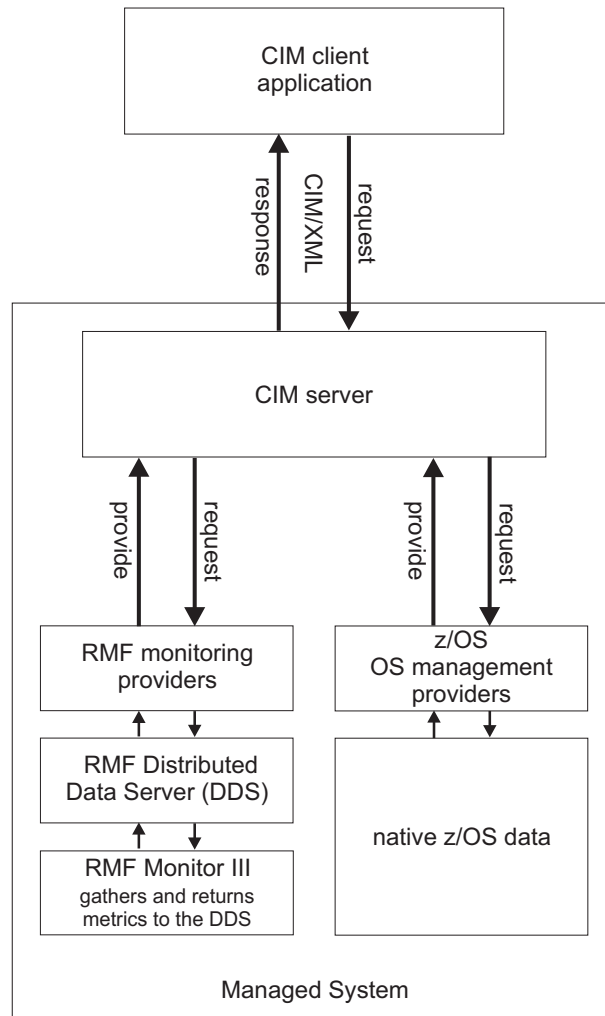


Figure 2. Exemplary tasks of the CIM server in a z/OS environment

For more information about the CIM standard, see the *CIM Specification* on the DMTF web site.

Important Note:

Each IBM® eServer™ operating system is supporting a specific open source implementation of a CIM server. The @server *Common Information Model* document contains overall information about how to use CIM for systems management on IBM @servers. Users of CIM for z/OS need to know this information. The present z/OS Common Information Model User's Guide contains the z/OS-specific supplements and deviations from the common @server CIM and from OpenPegasus.

Starting with z/OS 1.8, CIM supports the concept of indications as described in the following chapter.

Support of CIM indications

Copyright attribution:

The introduction to CIM indications provided in this section is based on the information in the *CIM Event Model White Paper, DSP0107*, Document Version 2.1 June 10,2003, provided by the Distributed Management Task Force (DMTF).

In CIM terminology, an indication is the representation of the occurrence of an event. For example, an event can be the unexpected termination of a program, or the modification of a property value of a CIM instance. There is not necessarily a one-to-one correspondence between events and indications. In particular, multiple indications can be generated for the same underlying event if multiple CIM client applications had subscribed for the event. An event can also occur without causing a related indication to be raised, for example if no subscription was made for the event.

Indications also may correlate. Two indications are correlated, if they do not represent the same underlying event, but there is a relationship between the events they represent.

With z/OS 1.8, support for indications has been added to the CIM infrastructure. As an example, RMF can generate indications for monitoring data, this way enabling CIM clients to support event-based monitoring, which eliminates expensive polling cycles by the CIM client. A CIM client can subscribe for certain conditions, for example whether a certain performance metric value is above a threshold. While the subscription is active, the RMF indication provider checks the condition independently and notifies the CIM client whenever the condition becomes true.

The CIM indication support comprises the following steps:

- define an indication filter condition: this describes the event that you might want to be notified about, that is, when to send an indication,
- define an indication listener: this describes how and where to send an indication
- activate the subscription by associating a filter with a listener,
- consume the indication once it is raised: the indication is sent to the indication listener, which decides how to react to the event.

The CIM Event Model defines the CIM classes used for indication support. It defines the CIM indication class hierarchy that is used to model various types of events, and the CIM subscription mechanism.

The CIM Event Model is described in the *CIM Event Model White Paper, DSP0107*, Document Version 2.1 June 10,2003, provided by the Distributed Management Task Force (DMTF).

The *Specification for CIM Operations over HTTP* describes how the CIM server transmits CIM indications to the CIM listener.

How indications work

Indications are generated and processed as shown in Figure 3 and described in the subsequent list:

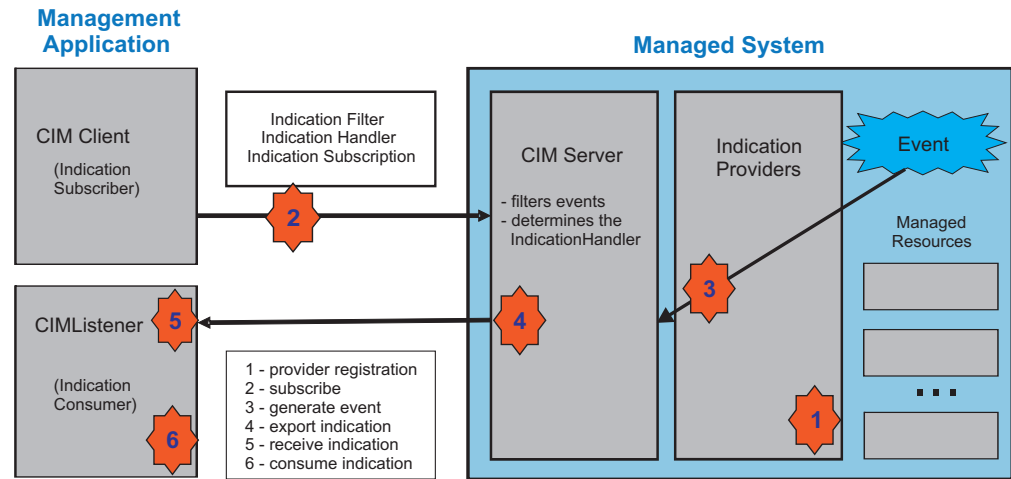


Figure 3. CIM indication flow and processing

1. Indication providers are registered:
 - An indication provider is registered with a CIM server just as any other provider is registered (using *PG_ProviderCapabilities*). An indication provider is a CIM provider that recognizes when a particular type of event occurs on the managed system. The indication provider turns that event into a type of *CIM_Indication* and gives it to the CIM server.
2. A CIM client defines an indication filter condition in order to specify the event to be notified about.
 - The CIM client issues CIM operation requests to the CIM server to create an instance of the *CIM_IndicationFilter* class.
 - The *CIM_IndicationFilter* instance defines the event to be notified about, in the form of a query string in a query language like CQL (CIM Query Language, see DMTF document DSP0202: *CIM Query Language Specification*) or deprecated WQL (WBEM query language).
3. A CIM client defines an indication listener in order to specify how to handle and where to send an indication.
 - The CIM client issues CIM operation requests to the CIM server to create an instance of the *CIM_ListenerDestinationCIMXML* class.
 - A *CIM_ListenerDestinationCIMXML* is an abstract superclass that specifies how to handle and where to send the indication. This may define a destination and protocol for delivering indications, or it may define a process to invoke. z/OS will support the subclass *CIM_ListenerDestinationCIMXML* as a vehicle to describe, via CIMXML, the destination for indications to be delivered. The destination is a URL.
4. A CIM client activates the subscription.
 - The CIM client issues CIM operation requests to the CIM server to create an instance of the *CIM_IndicationSubscription* class.
 - A *CIM_IndicationSubscription* is an association between a *CIM_IndicationFilter* and a *CIM_ListenerDestinationCIMXML* (see Figure 5 on page 9).

5. The three CIM instances mentioned above are created by the CIM client using the **createInstance** CIM operation. The instances must be created in the root/PG_InterOp namespace of the CIM server.
6. At some future time an event occurs on the managed system and is detected by the CIM indication provider.
 - The CIM indication provider turns that event into a specific indication. At this stage, the indication is a local representation of an instance of a subclass of class **CIM_Indication**. It delivers that indication to the CIM server for further processing and delivery.
 - Typically the indication is an instance of a subclass of class **CIM_ProcessIndication** or class **CIM_InstIndication**.
7. The CIM server filters the indications delivered by the indication provider according to the filter conditions of the active subscriptions.
8. The remaining indications are delivered to the CIM listeners specified in each of the **CIM_ListenerDestinationCIMXML** instances associated to the matching filter conditions via active subscriptions.
 - The CIM server generates a CIM export message to transmit the **CIM_Indication** instance to the CIM listener URL in the format and protocol specified in the **CIM_ListenerDestinationCIMXML** instance.
9. The **CIM_Indication** instance is received by the CIM listener.
The CIM listener or CIM server coordinates the distribution of the indication to one or more registered indication consumers and sends CIM export responses.
10. The **CIM_Indication** is delivered to one or more indication consumers.

CIM indications

Indications in CIM are represented as instances of class **CIM_Indication**. This abstract class serves as the base class for all indication classes.

Indications are transient instances used to distribute information from an indication generator to an arbitrary number of indication consumers. Therefore, they are typically very short-living. Indications have a source namespace, this is the value of the **SourceNamespace** property of the **CIM_IndicationFilter** instance that produced the indication. Although indications are instances of CIM classes, they are unique in that they cannot be addressed, but can only be received by subscription. Hence, indication instances cannot be enumerated, created, deleted, retrieved or modified by client operations.

Note that z/OS does not ship generic providers, that is, an indication subscription is only processed if the required indication provider exists and is registered with the CIM server.

The CIM Schema version provided with z/OS supports three types of indications (representing different types of events) which are modeled as **CIM_Indication** subclasses. These subclasses include:

CIM_InstIndication

used to report life cycle events for CIM instances. Types of events include: Instance creation, deletion, modification, method invocation and read access. For each of these types, a specific subclass of **CIM_InstIndication** is defined in the CIM Schema: **CIM_InstCreation**, **CIM_InstDeletion**, **CIM_InstModification**, **CIM_InstMethodCall** and **CIM_InstRead**.

CIM_ClassIndication

used to report lifecycle events for CIM classes. Types of events include: Class creation, deletion, modification, and read access. This type of indication is not implemented by z/OS.

CIM_ProcessIndication

used to report the occurrence of any other event, typically alert type events. See “CIM_ProcessIndication” on page 8.

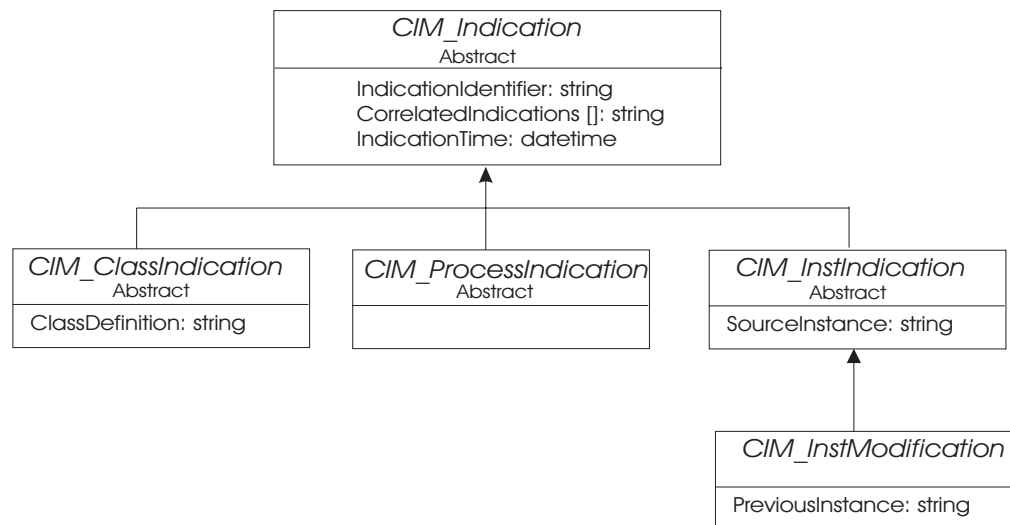


Figure 4. CIM indication hierarchy

CIM indication class hierarchy

The CIM indication class hierarchy models the types of events that can be detected. An instance of **CIM_Indication** represents the occurrence of an event in general. Indication instances cannot be addressed, but they have a source namespace. Although indications are modeled using CIM classes, indications are unique in that they cannot be manipulated or retrieved, but they can only be received by subscription. The **CIM_Indication** class is the base class for all other indication classes. It includes the following properties:

IndicationIdentifier

identifies indication instances uniquely within their source namespace.

IndicationTime

describes, to the extent possible, the time and date of the creation of the underlying event for the indication.

CorrelatedIndications

specifies a list of other indications, referenced by their **IndicationIdentifier** property values, that are related to this indication. These **IndicationIdentifier** property values are interpreted to have the same source namespace as this indication.

While the **CorrelatedIndications** property values are to be interpreted in the context of a single CIM namespace, any instances of other classes of the CIM Event Model do not need to be located in the same namespace.

CIM_ProcessIndication

CIM_ProcessIndication models any events other than lifecycle events. In the CIM Schema version supported for z/OS, the following two subclasses of **CIM_ProcessIndication** are defined:

- **CIM_AlertIndication** – signals the occurrence of an alert type of event. Properties of this subclass include **PerceivedSeverity**, **ProbableCause**, **RecommendedAction** and **Trending**, describing an alerting situation.
- **CIM_SNMPTrapIndication** – used to map SNMP traps to CIM indications. This is currently not supported by the z/OS CIM server.

CIM_InstIndication

An instance of **CIM_InstIndication** denotes the occurrence of a lifecycle event on a CIM instance. The possible lifecycle events are: creating an instance, deleting an instance, modifying an instance, reading an instance or invoking a CIM method on an instance. An instance of **CIM_InstIndication** includes an embedded copy (that is, a current snapshot) of the instance, **SourceInstance**, on which the lifecycle event occurred.

CIM_InstModification

Instances of **CIM_InstModification** include an embedded copy of the instance, **PreviousInstance**, before the modification occurred.

Lifecycle events on CIM instances include both, changes caused by a CIM client, and changes that happen spontaneous from a CIM client perspective due to volatile behavior of the CIM provider.

CIM subscription mechanism

The CIM Event Model defines how CIM clients subscribe to receive indications. A **CIM_IndicationFilter** instance describes the set of indications of interest, a **CIM_ListenerDestinationCIMXML** instance defines the **CIM listener** and the communication protocol, that is, it describes the method and targets for distributing the indications. Finally, a **CIM_IndicationSubscription** association instance between the **CIM_IndicationFilter** instance and the **CIM_ListenerDestinationCIMXML** instance is used to subscribe for receiving these indications. The creation of this association instance activates the subscription.

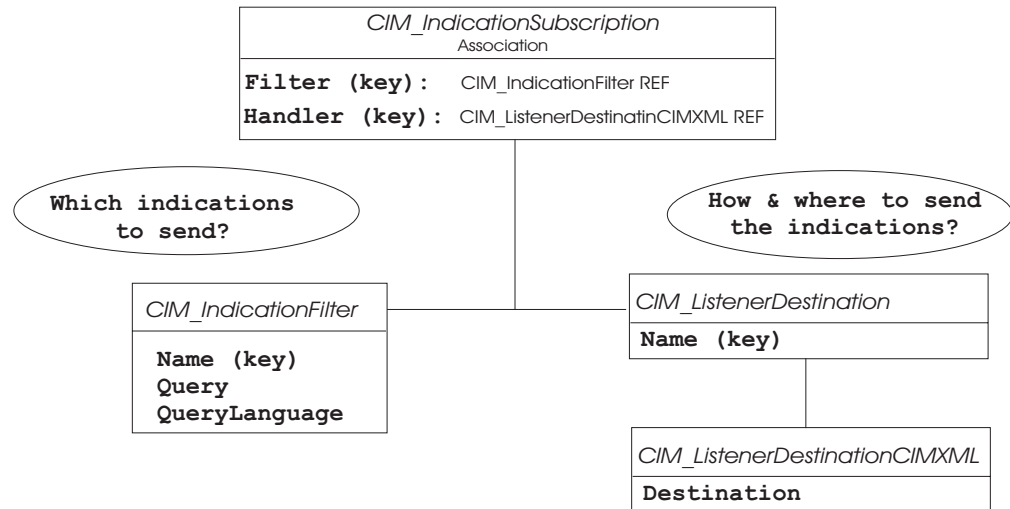


Figure 5. Indication subscription class diagram

CIM_IndicationFilter

An instance of **CIM_IndicationFilter** describes the set of indications of interest by means of a query expression. This is also called the desired indication stream. The most relevant properties of **CIM_IndicationFilter** are:

- **Name, CreationClassName, SystemName, SystemCreationClassName** – key properties.
- **SourceNamespace** – defines the source namespace for the indications resulting from this indication stream.
- **Query** – query string, like “select * from CIM_InstModification where ...”; defines the indication class, filter condition and property list of the indication stream.
- **QueryLanguage** – defines the query language used in the **Query** property. The z/OS CIM server supports the query languages “DMTF:CQL” (CIM Query Language) and “WQL” (WBEM Query Language). RMF providers only support query language “DMTF:CQL”. For more information, see the *CIM Query Language Specification, DSP0202*, on <http://www.dmtf.org/standards/documents/WBEM/DSP0202.pdf>.

For information about the complete set of properties of a **CIM_IndicationFilter**, refer to the *CIM Event Model White Paper* or to the definition of this class in the CIM Schema. The white paper also contains an example of a **CIM_IndicationFilter** instance.

CIM_ListenerDestinationCIMXML

An instance of **CIM_ListenerDestinationCIMXML** defines “how and where” to send an indication. In particular, the **CIM_ListenerDestinationCIMXML** instance defines the desired indication destination, encoding and protocol for delivery of the indication stream. **CIM_ListenerDestinationCIMXML** specializes **CIM_ListenerDestination** and is used for indication consumers that support the CIM Operations over HTTP protocol (see Specification for CIM Operations over HTTP, DSP0200, on <http://www.dmtf.org/standards/documents/WBEM/DSP200.html>).

The **CIM_ListenerDestination** class hierarchy can be extended to allow the definition of additional indication handling mechanisms.

The most relevant properties of **CIM_ListenerDestinationCIMXML** are:

- **Name**, **CreationClassName**, **SystemName**, **SystemCreationClassName** – key properties
- **Destination** – URL to which the indications are to be delivered

For information about the complete set of properties of **CIM_ListenerDestinationCIMXML**, refer to the *CIM Event Model White Paper* or to the definition of this class in the CIM Schema.

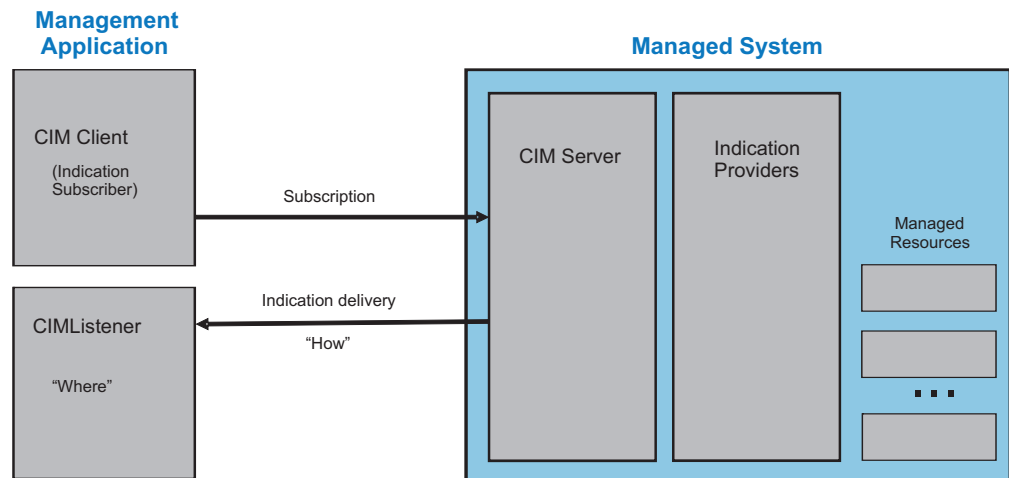


Figure 6. Indication Handler

CIM_IndicationSubscription

Primarily, an instance of **CIM_IndicationSubscription** defines the association between a **CIM_IndicationFilter** instance and a **CIM_ListenerDestinationCIMXML** instance. In addition, it includes a set of properties that further specify the behavior of a subscription. The most relevant properties of **CIM_IndicationSubscription** are:

- The **Repeat Notification** properties (those having "RepeatNotification" contained in their property name) define the behavior for handling indications that report the occurrence of the same underlying event (that is, the disk is still generating I/O errors and has not yet been repaired).
- The **Subscription State** properties (those having "SubscriptionState" contained in their property name) allow a CIM client to monitor and control the state of the subscription.
- The **Subscription Failure Handling** properties (**OnFatalErrorPolicy**, **OtherOnFatalErrorPolicy**, **FailureTriggerTimeInterval**) define the desired behavior when a fatal error occurs during subscription processing.
- The **Subscription Duration** properties (**SubscriptionDuration**, **SubscriptionStartTime**, **SubscriptionTimeRemaining**) allow to expire a subscription automatically, based upon elapsed time since its creation, and to monitor the elapsed times since creation and until expiration.

You can find more detailed information about these properties as well as the complete set of properties of **CIM_IndicationSubscription** in the *CIM Event Model White Paper* or in the definition of this class in the CIM Schema.

CIM server security

The CIM server security consists of two major areas. Protection of resources on the managed system through *authentication* and *authorization*, and protection of communicated information through *network security*.

The AT-TLS feature of z/OS is used to encrypt data using SSL for data security on the network. It is recommended to utilize this support.

To protect resources on the managed system from unauthorized access, first of all users have to be authenticated to ensure the CIM server is really communicating with a specific identity (user). Authentication of users can be done by either using a user identity and a password, a user identity and a PassTicket, or a user certificate. In both cases after successful authentication the user who wants to access the system is well known and now authorization checks against that specific user identity are executed.

Three types of authorization checks are being done by the CIM server:

1. General access to CIM is being checked for a user. To get general access to CIM, a user needs at least READ access to profile CIMSERV in class WBEM.
2. The access to the provider is checked. Access to a provider can be explicitly restricted by defining a provider-specific profile in RACF class WBEM and registering the provider with that security profile. This access restriction is optional and depends on whether a provider was registered with a security profile or not.
3. The last checks of authorization are being done based on the z/OS system resources a user tries to access, what effectively means that a user can only access the resources for which he was entitled before.

Figure 7 on page 12 shows the CIM server runtime environment security.

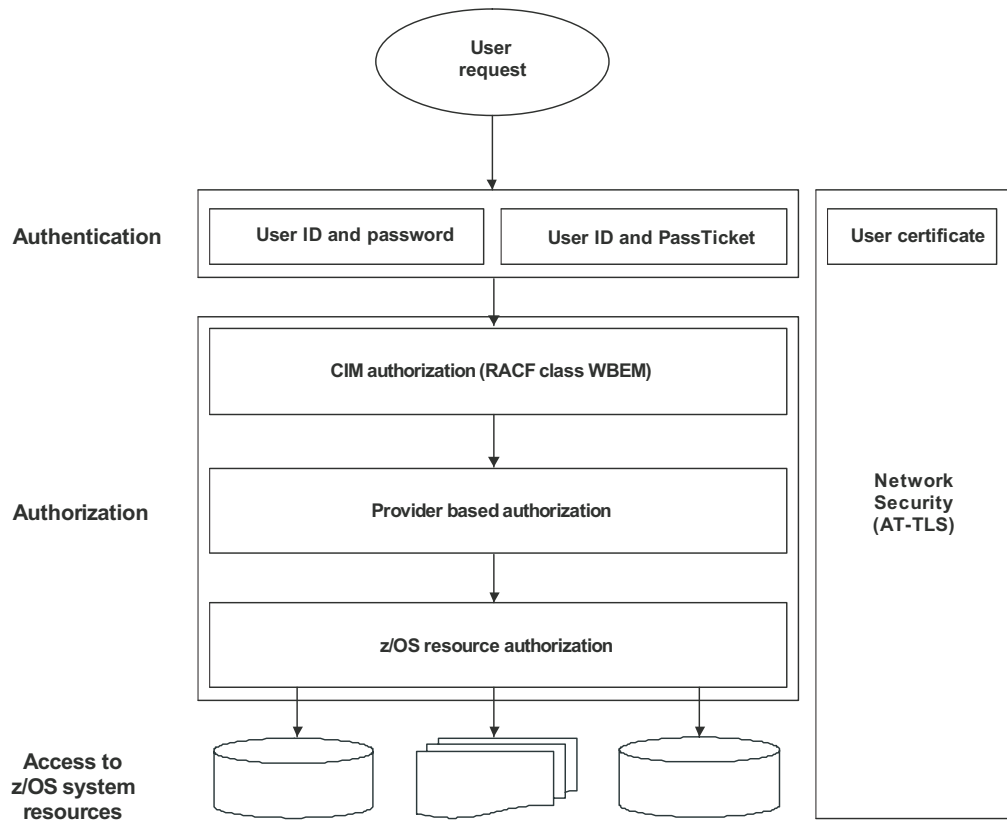


Figure 7. Security components

Network security

AT-TLS provides network security. It is recommended to utilize this feature.

Authentication

Authentication is always enabled for the CIM server. The CIM server checks whether the requestor is entitled to use the CIM server. A requestor authenticates with a user ID and a password, with a user ID and a PassTicket, or with a user certificate.

Authorization

CIM authorization (RACF class WBEM)

The CIM server controls whether the user ID is authorized to access the CIM server using the RACF class WBEM. The profile CIMSERV restricts access to the CIM server.

Provider based authorization

Optionally, a provider can be registered with a specific security profile. In this case, the user ID has to be authorized before it can invoke the provider. A provider-specific profile in RACF class WBEM restricts the access to the provider.

These checks are strongly recommended for providers which use a designated user ID.

z/OS resource authorization

The z/OS system resource access authorization is verified against the requesting user ID.

| For authorization purposes to specific z/OS system resources, the
| CIM server processes requests either under the user ID from which
| the request was generated or under a designated user ID which
| was registered for the provider. To do this, the CIM server uses
| thread-level security, which is provided by the UNIX System
| Services.

| Additionally, the CIM server is enabled for the *Enhanced Security*
| *model*. Under the Enhanced Security model, the CIM server does
| not load any dynamic link library that is not program controlled,
| in particular it does not load any such provider DLL.

| See “Customizing the security for the CIM server” on page 21 for setting up the
| security for the CIM server.

Chapter 2. Installation and migration

This chapter lists the software and hardware prerequisites and describes how to install and how to migrate the CIM server to the current release.

Software and hardware prerequisites

z/OS® CIM is delivered with the z/OS operating system and runs on any hardware that can exploit the current z/OS version as described in the *z/OS Program Directory*.

Installation

You install z/OS CIM using SMP/E as described in the *z/OS Program Directory*. For additional setup steps, refer to Chapter 3, “First-time CIM server set up,” on page 21. Note that other z/OS components that provide CIM instrumentation, like for example RMF or DFSMSrmm, may require additional configuration steps which are described in the respective product documentation.

Installation directories

After a successful SMP/E installation, the components of z/OS CIM are located in the following hierarchical file system directory:

Table 2. Default installation directories for z/OS CIM

Directory	Description
/usr/lpp/wbem	Base hierarchical file system directory
/usr/lpp/wbem/bin	CIM server executables
/usr/lpp/wbem/lib	CIM server libraries (dlls)
/usr/lpp/wbem/install	Sample profile
/usr/lpp/wbem/provider	CIM provider libraries provided with z/OS
/usr/lpp/provider/schemas	IBM z/OS instrumentation MOF files
/usr/lpp/wbem/msg	CIM server message files for NLS
/usr/lpp/wbem/schemas	CIM MOF schema files (CIM 2.13)
/usr/lpp/wbem/repository	CIM schema version 2.13 master repository
/usr/lpp/wbem/jclient	CIM client for Java version 1 and 2
/usr/lpp/wbem/IBM	SMP/E target library path

The configuration files and the repository are installed or migrated during separate steps. Refer to the appropriate migration information: “Migration from z/OS 1.8 or z/OS 1.9 to z/OS 1.10” on page 18.

For a better maintainability, it is recommended to mount a separate file system on */var/wbem* for the CIM server's data repository. The recommended size is 100MB. You can do so by adding a mount statement in your **BPXPRMxx PARMLIB** member:

Mount statement:

```
MOUNT FILESYSTEM(OMVS.HPG7750.VARWBEM.ZFS)
      TYPE(ZFS)
      MOUNTPPOINT('/var/wbem')
      MODE(RDWR)
```

Customize CFZRCUST

The job **CFZRCUST** is for installing and migrating the z/OS CIM server configuration and repository on each target machine. A sample of **CFZRCUST** is shipped with the default SAMPLIB.

Prerequisites:

- ___ 1. The target system is running with configured UNIX System Services.
- ___ 2. The CIM server must be stopped.
- ___ 3. The user running this job
 - must either have UNIX user ID 0
 - or must be able to copy files and set the program control bit on files.
- ___ 4. If you intend to mount the data on a separate file system - which is recommended - this user must be entitled to allocate a 100MB hierarchical file system or zFS data set (if not yet allocated).

Adjust the sample job **CFZRCUST**, which is located in the SAMPLIB, to fit your environment:

If you want to place the repository and the log files on a separate file system:

- ___ 1. Adjust the job card.
- ___ 2. Adjust STEP 1a or STEP 1b of the JCL to create a file system data set. Choose one of these steps to create a data set and delete the unwanted step. You must provide the name and file system type in the JCL for further processing the selected sample job.
As an alternative, you can also create the file system outside of this JCL.

STEP 1a is a sample to allocate a zFS file system dataset:

JCL - sample STEP 1a

```
/*STEP 1a - Create zFS DataSet for /var/wbem */
//*****
//DEFZFS EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//DASD0 DD DISP=(NEW,CATLG),UNIT=unit,VOL=SER=volser
//SYSIN DD *
//DEFINE CLUSTER( -
//NAME(%CFZVARWBEMDS%) -
//VOLUMES(volser) -
//STORAGECLASS(OMVS) -
//LINEAR -
//CYLINDER(150 15) -
//SHAREOPTIONS(3) -
//)
//FRMZF EXEC PGM=IOEAGFMT,REGION=OM,
// PARM=(' -aggregate %CFZVARWBEMDS% -compat ')
//STEPLIB DD DISP=SHR,DSN=SYS1.SIOELMOD
//SYSPRINT DD SYSOUT=*
//STDOUT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
```

STEP 1b is a sample to allocate an hierarchical file system dataset.

JCL - sample STEP 1b

```
/*STEP 1b - Create HFS DataSet for /var/wbem */
//*****
//ALLOCHFS EXEC PGM=IEFBR14
//*
//HFS2 DD DSN=%CFZVARWBEMDS%,
// DISP=(NEW,CATLG),
// UNIT=unit,
// VOL=SER=volser,
// SPACE=(CYL,(150,15,1)),
// DSNTYPE=HFS,
// STORCLAS=OMVS
```

3. If you are using an extensible file system, you can suppress the check for enough free space by specifying the parameter `-noSpaceCheck` in the installation/migration utility at STEP 2 of the JCL. The system administrator is responsible to ensure that there is enough free space (60 MB) available for installation/migration, otherwise the job will fail. This would not suppress the check if a separate file system data set is used.

The beginning of STEP 2 will then look like:

JCL - sample STEP 2

```
/*STEP 2 - Run customization/migration utility */
//*****
//CFZRCUST EXEC PGM=BPXBATCH,TIME=NOLIMIT,REGION=OM,
// PARM='PGM /usr/lpp/wbem/install/CFZRCUST.sh -noSpaceCheck'
//*
```

- ___ 4. Replace the place holder %CFZVARWBEMDS% in the JCL with the name of the file system data set, for example:
OMVS.HPG7750.VARWBEM.ZFS.
- ___ 5. Replace the place holder %CFZVARWBEMDSTYPE% in the JCL with the file system type of the data set:
 - If you use STEP 1a to create the file system data set, replace with 'ZFS'.
 - If you use STEP 1b to create the file system data set, replace with 'HFS'.
- ___ 6. Submit the job. The return code (MAXACC) is 0 or 4 and indicates a successful installation or migration. If the return code is 12, look at the job output, correct the error and submit the job again.

If you do not intend to use a separate data set for the CIM server repository:

- ___ 1. Adjust the job card.
- ___ 2. Omit STEP 1a and STEP 1b of the sample job and specify the parameter -noDS in the installation/migration utility at STEP 2 of the JCL:

JCL - sample STEP 2

```

//*****
//* STEP 2 - Run customization/migration utility          */
//*****
//CFZRCUST EXEC PGM=BPXBATCH,TIME=NOLIMIT,REGION=0M,
//      PARM='PGM /usr/lpp/wbem/install/CFZRCUST.sh -noDS'
//*
```

- ___ 3. Submit the job. The return code (MAXACC) is 0 or 4 and indicates a successful installation or migration. If the return code is 12, look at the job output, correct the error and submit the job again.

After performing the actions described in Chapter 3, "First-time CIM server set up," on page 21, you can run the sample application **CIMIVP** delivered with the product as an installation verification program (see "Sample client applications" on page 74).

Migration from z/OS 1.8 or z/OS 1.9 to z/OS 1.10

You can install the z/OS 1.10 CIM server as a replacement of a previous z/OS CIM server version without affecting any external programs interfacing with the CIM server, such as management applications.

Installation

The SMP/E installation of the z/OS 1.10 CIM server installs a new master repository in */usr/lpp/wbem/repository*. (Existing versions of the repository are located in */var/wbem/repository*.)

- ___ 1. On each target system, mount a system specific data set with at least 100Mb size at */var/wbem* which either holds the z/OS 1.8 or z/OS 1.9 repository or is empty.
- ___ 2. In file */etc/wbem/cimserver_planned.conf*, ensure that the *logdir*, *enableNamespaceAuthorization* and the *httpAuthType* properties do not exist. If they exist, delete them.
- ___ 3. It is recommended to replace the environment file *cimserver.env* located in */etc/wbem* with the new sample installed in directory:

| */usr/lpp/wbem*

- | — 4. If you do not intend to replace the environment file `cimserver.env`
| with the new sample, make sure that the following directories are
| included in the `LIBPATH` defined in `cimserver.env`:

| */usr/lpp/wbem/lib:/usr/lpp/wbem/provider:/usr/lib*

| **CIM server startup**

| During startup, the z/OS 1.10 CIM server automatically corrects eventually
| missing file tags. In addition, it detects if an existing repository is up to
| date.

| If back-level, the CIM server automatically upgrades the repository:

- | 1. The CIM server backs up the current repository in
- | • `repository_old19_<timestamp>` for z/OS 1.9 repositories
 - | • `repository_old18_<timestamp>` for z/OS 1.8 repositories
 - | • `repository_old_<timestamp>` for other z/OS repositories
- | 2. The CIM server copies the master repository from */usr/lpp/wbem/*
| *repository* to */var/wbem/repository*.
- | 3. The CIM server migrates the previous repository content to the current
| repository.

| If there are any quotes in the `cimserver.env` file in */etc/wbem*, the CIM
| server removes the quotes, stops the startup procedure and requires you to
| restart the CIM server.

Chapter 3. First-time CIM server set up

This chapter describes the actions to be completed before the first invocation of the CIM server:

- “Customizing the security for the CIM server”
- “Customizing the CIM server startup” on page 34
- “Customizing environment variables” on page 35
- “Selecting a WLM service class for z/OS CIM priority” on page 36
- “Provider-based authorization model” on page 36

Customizing the security for the CIM server

The security design of the CIM server on z/OS is based on the open source implementation, but has been extended to meet z/OS specific needs.

The z/OS implementation of the CIM server requires each requestor to have a real z/OS user ID. Only users who have been successfully authenticated with the z/OS security product and who have been granted access to the CIM server, will be able to execute requests against the CIM server. This chapter describes the details on how to set up these features.

Setting up security for the CIM server includes the following steps:

- 1. Define a RACF class and profile for the CIM server (see “Defining a RACF class and profile for the CIM server” on page 22).
- 2. Define a user ID for the CIM server and grant it access to the CIM server’s RACF profile (see “Defining a user ID that runs the CIM server” on page 23).
- 3. Configure the CIM server’s resource authorization model (see “Configuring the CIM server’s resource authorization model” on page 23).
- 4. Grant client users and administrators access to the CIM server (see “Granting clients and administrators access to the CIM server” on page 26).
- 5. Allow the CIM server to surrogate for a client ID (see “Switching identity (surrogate)” on page 27).
- 6. Optionally configure secure connections (HTTPS) for the CIM server (see “Configuring the CIM server HTTPS connection using AT-TLS” on page 27).
- 7. For PassTicket validation define an application ID (see “Defining an application ID for PassTicket validation” on page 33)
- 8. Specific providers need an additional setup
Some providers as the network, job and cluster providers have additional security and setup requirements. Please see Chapter 5, “z/OS Management Instrumentation for CIM,” on page 79 for these providers. See also “Appendix D. Cluster / CoupleDataset / JES2-JES3Jobs Provider Setup” on page 193 for additional RACF setup.

The following chapters describe how each step of security can be enabled and configured.

Note: This document describes how to set up security using Resource Access Control Facility (RACF®) as security product. However, you can use any other suitable security product for this purpose.

Defining a RACF class and profile for the CIM server

Access to the CIM server is controlled through RACF class WBEM. How to define a new class in RACF through the dynamic CDT feature of the *z/OS Security Server* is described in the following:

- ___ 1. To be able to build the dynamic class WBEM, you must first build the class descriptor table (CDT) using the following RACF command:

Example:

```
SETOPTS CLASSACT(CDT) RACLIST(CDT)
```

- ___ 2. By adding a profile to the IBM class named CDT, you can create a new class definition. This profile then represents a dynamic class. The segment CDTINFO is used to define the class attributes. Definition of dynamic class WBEM can be done with the following RACF commands:

Example:

```
RDEFINE CDT WBEM UACC(NONE) CDTINFO(  
  CASE(UPPER)  
  FIRST(ALPHA)  
  OTHER(ALPHA,NUMERIC)  
  MAXLENGTH(246)  
  MAXLENX(246)  
  KEYQUALIFIERS(0)  
  PROFILESALLOWED(YES)  
  POSIT(200)  
  DEFAULTRC=8  
  DEFAULTUACC=NONE  
  RACLIST(REQUIRED)  
)  
SETOPTS RACLIST(CDT) REFRESH
```

The default values shown above (except POSIT(200)) are expected by the CIM server; do not use different values as this can yield unpredictable results. For a more detailed description of how to create a new class within RACF dynamic CDT, see *z/OS Security Server RACF Security Administrator's Guide*.

- ___ 3. To activate the new class, issue:

Example:

```
SETOPTS CLASSACT(WBEM) RACLIST(WBEM)
```

- ___ 4. After creating and activating the WBEM class, you must create the CIMSERV profile within this class through which users then can be granted access to the CIM server.

The following example illustrates the flow of RACF commands that are required to define a profile named CIMSERV in this class:

Example:

```
SETROPTS CLASSACT(WBEM)
SETROPTS RACLIST(WBEM)
RDEFINE WBEM CIMSERV
SETROPTS CLASSACT(WBEM) RACLIST(WBEM) REFRESH
```

Defining a user ID that runs the CIM server

Either select an existing user ID or create a new user ID that runs the CIM server. Depending on the security model under which the CIM server runs (see “Configuring the CIM server’s resource authorization model”), a privileged user (UID=0) may be required.

In addition, the CIM server’s user ID must be given CONTROL access to profile CIMSERV in class WBEM. The following example shows the required RACF commands to achieve this, where the user ID CFZADM was chosen for the CIM server:

Example:

```
PERMIT CIMSERV CL(WBEM) ACCESS(CONTROL) ID(CFZADM)
SETROPTS CLASSACT(WBEM) RACLIST(WBEM) REFRESH
```

Furthermore, a surrogate to itself must be defined for the CIM server’s own user ID (see “Switching identity (surrogate)” on page 27).

For running the CIM server started task, see “Customizing the started task procedure CFZCIM” on page 34 for how to associate the CIM server with the started task.

Configuring the CIM server’s resource authorization model

The CIM server can be run with two different authorization models, depending on whether the profile BPX.SERVER is defined in the FACILITY class or not. In any case, the CIM server follows a resource-based authorization model, which means that user requests are processed in separate threads, for which the security context is switched to the user ID of the requestor or to a designated user ID. So any z/OS system resource that is accessed directly through a CIM provider executing a user request in such a thread, will be accessed under the requestor’s or a designated user ID and thus, authorization checks occur against this user ID.

These checks will occur in addition to the general access check for the CIM server through the CIMSERV profile in class WBEM.

To let the resource based authorization security work properly, consider the following when setting up the user ID that runs the CIM server:

If the Enhanced Security model is disabled:

When the Enhanced Security model is disabled, no profile BPX.SERVER is active in the FACILITY class.

- Set up the user ID running the CIM server as a privileged user (UID=0).

If the Enhanced Security model is enabled:

When the Enhanced Security model is enabled, profile BPX.SERVER exists in the FACILITY class.

The definition of BPX.SERVER is not specific for the CIM server, but has system wide implications for all programs running on the z/OS system. Refer to *Setting up the BPX.* FACILITY class profiles* in *z/OS UNIX System Services Planning* for additional information.

- 1. Set up the user ID running the CIM server with READ or UPDATE access to BPX.SERVER.

UPDATE access means:

z/OS resource access authority is based **solely** on the requesting client's authority.

READ access means:

z/OS resource access authority is based on the server program's **and** the requesting client's authority. For example, even if the client user has a wider resource access authority than the server, the result is the least common denominator type of access authority.

- 2. Flag all CIM server libraries, as well as all installed providers, as "program controlled" (see "Program control" on page 25).
- 3. If the user ID that runs the CIM server is not privileged (UID ≠ 0), ensure that
 - either CHOWN.UNRESTRICTED is defined in RACF,
 - or the user ID that runs the CIM server has READ access to the SUPERUSER.FILESYS.CHOWN resource in the UNIXPRIV RACF class.

Refer to *Using UNIXPRIV class profiles* in *z/OS UNIX System Services Planning* for additional information.

- 4. If the user ID that runs the CIM server is not privileged (UID ≠ 0), ensure that the files and directory of the CIM installation on the target system are owned by this user ID.

The following example shows how to change ownership:

Example:

```
chown -R <Server UserID>:<Server GroupID> /etc/wbem
chown -R <Server UserID>:<Server GroupID> /var/wbem
```

If any of these requirements is not met, the CIM server will not start, but issue an according error message in the logs.

Enabling the must-stay-clean feature

Provider libraries are loaded dynamically during runtime by the CIM server. To prevent the CIM server from loading non-program-controlled providers, the z/OS must-stay-clean feature must be enabled.

This feature is optional. You only must enable this feature if you want to add further system integrity to the CIM server.

The second reason for switching must-stay-clean on is that due to the *Out-of-Process* feature, providers can be managed in separate address spaces rather than loading and calling provider libraries directly within the CIM server process.

This converts the CIM server process into a daemon process that starts off several server processes (Provider Agent processes). Providers are then run in threads by the Provider Agent. The trust base between both address spaces can be secured by switching on the z/OS must-stay-clean feature.

The CIM server uses the must-stay-clean feature only if the BPX.DAEMON FACILITY class is defined in your security product. Defining BPX.DAEMON enforces program control. The following sample shows the according RACF commands:

Example:

```
RDEFINE FACILITY BPX.DAEMON UACC(NONE)
SETROPTS CLASSACT(FACILITY)
SETROPTS RACLIST (FACILITY)
SETROPTS RACLIST(FACILITY) REFRESH
```

Note: The definition of BPX.DAEMON is not specific for the CIM server, but has system wide implications for all programs running on the z/OS system. Refer to *Setting up the BPX.* FACILITY class profiles* and *Setting up security procedures for daemons* in *z/OS UNIX System Services Planning* for additional information.

Program control

Program control means that all programs running in the address space have been loaded from a library that is controlled by a security product. A library identified to RACF program control is an example. Refer to *z/OS UNIX System Services Planning* for additional information about program control.

If the CIM server runs with authority to BPX.SERVER or with the must-stay-clean feature, the server must run in a clean program controlled environment.

By default, all libraries shipped with the CIM server are flagged as program controlled. If additional provider libraries are installed, it may be required to set the program control flag manually using the extattr +p <libname> command.

In addition to the UNIX System Services files, you must mark several MVS libraries as program controlled. The following sample shows the according RACF commands. This list contains more libraries than you need to mark as program controlled for the CIM server, but they are needed to keep the system accessible via 3270telnet, telnet and ftp.

Example:

```
RALT PROGRAM * ADDMEM('SYS1.SCEERUN'/'*****'/NOPADCHK) UACC(READ)
RALT PROGRAM * ADDMEM('SYS1.SCEERUN2'/'*****'/NOPADCHK) UACC(READ)
RALT PROGRAM * ADDMEM('SYS1.SCLBDLL'/'*****'/NOPADCHK) UACC(READ)
RALT PROGRAM * ADDMEM('SYS1.SCLBDLL2'/'*****'/NOPADCHK) UACC(READ)
RALT PROGRAM * ADDMEM('SYS1.SCLBCPP'/'*****'/NOPADCHK) UACC(READ)
RALT PROGRAM * ADDMEM('SYS1.SIEALNK'/'*****'/NOPADCHK) UACC(READ)
RALT PROGRAM * ADDMEM('SYS1.SIEALNKE'/'*****'/NOPADCHK) UACC(READ)
SETR WHEN(PROGRAM) REFRESH
```

You also must ensure that the CIM server runtime environment runs in its own address space. To run the CIM server runtime environment in its own address space, either start it using the provided started task procedure or set the

environment variable `_BPX_SHAREAS=NO` in your z/OS UNIX System Services shell before starting it with the `cimserver` command.

Granting clients and administrators access to the CIM server

The CIM server authenticates users with the *z/OS Security Server* to determine which users can log into it. Authentication is performed for every new connection (local or remote) before a user is granted access to the CIM server.

For the CIM server for z/OS, users log in over HTTP or HTTPS using basic authentication or certificate authentication. When logging in, users are authenticated using their z/OS user ID and password as defined, for example, in RACF.

A CIM user must have at least READ access to the RACF profile CIMSERV in class WBEM to access the CIM server. In order to use any of the administrative command-line tools of the CIM server, as described in “CIM server command-line utilities and commands” on page 52, a user instead requires CONTROL access to the CIMSERV profile.

For detailed information on the required access rights, see the following table.

Table 3. Access types required for CIM operations

CIM Operation type	CIM Operations	RACF access
Basic read	GetClass, EnumerateClasses, EnumerateClassNames, GetInstance, EnumerateInstance, EnumerateInstanceNames, GetProperty, GetQualifier, EnumerateQualifier	READ
Basic write	SetProperty	UPDATE
"Method"	ExecuteMethod	UPDATE
Schema Manipulation	CreateClass, ModifyClass, DeleteClass	CONTROL
Instance Manipulation	CreateInstance, ModifyInstance, DeleteInstance	UPDATE
Association Traversal	Associators, AssociatorNames, References, ReferenceNames	READ
Query	ExecQuery	READ
Qualifier Declaration	SetQualifier, DeleteQualifier	CONTROL

Note: For indications subscriptions, CONTROL access is required.

The following example shows how to define READ access for a client user ID called CFZUSR:

Example:

```
PERMIT CIMSERV CL(WBEM) ACCESS(READ) ID(CFZUSR)
SETROPTS CLASSACT(WBEM) RACLIST(WBEM) REFRESH
```

In addition, the CIM server's user ID must be defined as a surrogate of the client user ID (see “Switching identity (surrogate)” on page 27).

For a user to be able to use the command line tools, this UNIX System Services environment has to be set up as described in “Customizing the UNIX System Services shell for CIM server startup” on page 34

Switching identity (surrogate)

The CIM server uses services which can be run in client or server security context. Therefore, the CIM server’s user ID must be able to switch the user ID to itself and to the client’s user ID. To enable the CIM server’s user ID to switch user ID, BPX.SRV profiles for the SURROGAT class have to be defined within your SAF facility.

There are two alternate ways to do this:

- You can specify a general profile to allow the CIM server user ID to switch to any other z/OS user ID with a UNIX System Services segment defined. The following sample shows the required RACF commands to create the generic profile, where the CIM server’s user ID is CFZADM:

Example:

```
SETROPTS CLASSACT(SURROGAT) RACLIST(SURROGAT)
RDEFINE SURROGAT BPX.SRV.** UACC(NONE)
PERMIT BPX.SRV.** CLASS(SURROGAT) ACCESS(READ) ID(CFZADM)
SETROPTS GENERIC(SURROGAT) RACLIST(SURROGAT) REFRESH
```

- Alternatively, you can define a separate surrogate profile for each user ID intending to use the CIM server, including the CIM server user ID. The following sample shows the required RACF commands to create the CIM server user ID profile, where the CIM server’s user ID is CFZADM:

Example:

```
SETROPTS CLASSACT(SURROGAT) RACLIST(SURROGAT)
RDEFINE SURROGAT BPX.SRV.CFZADM UACC(NONE)
PERMIT BPX.SRV.CFZADM CL(SURROGAT) ACCESS(READ) ID(CFZADM)

SETROPTS GENERIC(SURROGAT) RACLIST(SURROGAT) REFRESH
```

The following sample shows the required RACF commands to create the client user ID profile, where the CIM server’s user ID is CFZADM and the CIM client’s user ID is CFZUSR:

Example:

```
RDEFINE SURROGAT BPX.SRV.CFZUSR UACC(NONE)
PERMIT BPX.SRV.CFZUSR CL(SURROGAT) ACCESS(READ) ID(CFZADM)
SETROPTS RACLIST(SURROGAT) REFRESH
```

Configuring the CIM server HTTPS connection using AT-TLS

The CIM server runtime environment can profit from the Application Transparent Transport Layer Security (AT-TLS) functionality. The communication between the CIM client and the CIM server can be secured by encryption (SSL). Additionally the CIM client can be authenticated by a certificate and mapped to a local z/OS user ID.

The following task describes how to configure the CIM server HTTPS connection using AT-TLS.

___ 1. **Prerequisites**

- ___ Ensure that the basic setup for the Policy Agent is done
See *z/OS Communications Server: IP Configuration Guide* about policy-based networking and data protection.
- ___ Ensure that the basic certificates setup is done
For handling certificates for secure communications for RACF, see *z/OS Security Server RACF Security Administrator's Guide* about RACF and digital certificates.

___ 2. **Configuring the CIM server runtime**

- ___ a. Set the configuration property **enableHttpsConnection** to true.
- ___ b. Ensure that the configuration property **httpsPort** is set to 5989.
This default should not be changed.
Based on this configuration, the CIM server opens a second listener for receiving client connections and ensures that these connections are secured by AT-TLS. The level of protection depends on the configuration of AT-TLS. If a connection on this port is not secured by AT-TLS, the connection is closed and an appropriate error message is issued on the operator console.

___ 3. **Configuring the Policy Agent to secure communication for the CIM server**

- ___ Enable the Policy Agent for AT-TLS. See *z/OS Communications Server: IP Configuration Guide* about Application Transparent Transport Layer Security data protection.
- ___ Configure the Policy Agent to secure the communication for the CIM server at the configured HTTPS port (configuration property **httpsPort**). For sample Policy Agent policies, see "Example: Configuring AT-TLS for secure communication" to configure either an SSL protection or an SSL protection including a certificate based authentication.
- ___ Optionally you can protect the indication delivery on a specific port range with SSL.

Example: Configuring AT-TLS for secure communication

This topic shows the exemplary setup of the Policy Agent to secure communication for the CIM server.

- SSL protection only (see "Prerequisite" and "SSL protection only" on page 29)
- SSL protection including certificate based authentication (see "Prerequisite" and "SSL protection including certificate based authentication" on page 30)
- SSL protected indication delivery (see "Prerequisite" and "SSL protected indication delivery" on page 31)

For a more detailed explanation about Policy Agent AT-TLS policy see *z/OS Communications Server: IP Configuration Reference* about Policy Agent and policy applications and Application Transparent Transport Layer Security (AT-TLS) policy statements.

Prerequisite: Common certificate setup

To enable AT-TLS to secure the communication, a valid server certificate, the associated server private key, and the certificate of trusted Certificate Authority's (CA) are needed. These examples are using a key ring named CFZCIMServerRing to store these credentials. This key ring must be

accessible by the CIM server user ID (e.g. CFZADM), and the server certificate must be the default certificate.

For a sample setup with RACF, see *z/OS Security Server RACF Security Administrator's Guide* about RACF and digital certificates, implementation scenario 1 or 2. For handling certificates and key rings, please refer to the documentation of your SAF product.

SSL protection only

Simple SSL protection means that the communication between the client and the server is encrypted without having established a trust relationship between the client and the server. So the client needs to send a user ID and a password for authentication.

To set up AT-TLS with simple SSL protection for the CIM server, a policy for the Communications Server Policy Agent has to be created that restricts AT-TLS to the CIM server port 5989 and to inbound TCP/IP communication.

Sample Policy Agent policy for a simple SSL protection:

Example

```
TTLRule          CFZCIMServerRuleInbound
{
    Jobname          CFZCIM*
    LocalPortRange    5989
    Direction         Inbound
    TLSGroupActionRef grp_StartUp
    TLSEnvironmentActionRef CFZCIMServerEnvActionInbound
}

TLSEnvironmentAction CFZCIMServerEnvActionInbound
{
    HandshakeRole      Server
    TLSEnvironmentAdvancedParms
    {
        ClientAuthType PassThru
    }
    TLSKeyRingParms
    {
        Keyring          CFZCIMServerRing
    }
}

# Common StartUp Group that new Rules may use
# Shows how each connection maps to policy
TLSGroupAction grp_StartUp
{
    TLSEnabled On
    Trace 0          # Log Errors and Info messages to syslogd
}
```

CIM server specific notes to the AT-TLS Policy parameters:

TTLRule: Jobname

Jobname identifies where this rule applies. In the example, it is the started task job name. If you set up the connection this way, the configuration does not influence other parts of the system.

TTLSRule: LocalPortRange

This property must match the HTTPS port definition of the CIM server.

SSL protection including certificate based authentication

Since z/OS 1.9, you can use SSL secured communications and certificates based authentication between the CIM client and the CIM server, because the CIM server is aware of AT-TLS. The CIM server queries AT-TLS if the client is identified by a client certificate and mapped to a local user ID.

Authentication based on SSL certificates means:

- the communication between the client and the server is encrypted,
- the trust relationship is established, and
- the client certificate is matched to a local z/OS user ID.

No user ID and password have to be provided by the client. All subsequent authorization checking is done with the mapped user ID.

The CIM client sends an SSL certificate to AT-TLS, AT-TLS sends the certificate to RACF and RACF associates the certificate to the appropriate user ID, which then can access the CIM server. Vice versa, the CIM server returns its responses to clients' requests using SSL certificates.

This method of authentication provides more security than sending user IDs and passwords between client and server. If you want to use this enhanced method based on certificates, you must create the inbound/outbound rules as follows:

To set up AT-TLS with authentication based on SSL certificates for the CIM server, a policy for the Communications Server Policy Agent has to be created that restricts AT-TLS to the CIM server port 5989 and to inbound TCP/IP communication. Also the SAF facility has to be set up to match certificate subjects to local z/OS user ID's.

For setting up the SAF facility to map certificates to local user IDs, see *z/OS Security Server RACF Security Administrator's Guide* about RACF and digital certificates, Certificate Name Filtering.

Sample Policy Agent policy for authentication based on SSL certificates:

Example

```
TTLRule CFZCIMServerRuleInbound
{
  Jobname CFZCIM*
  LocalPortRange 5989
  Direction Inbound
  TLSGroupActionRef grp_StartUp
  TLSEnvironmentActionRef CFZCIMServerEnvActionInbound
}

TLSEnvironmentAction CFZCIMServerEnvActionInbound
{
  HandshakeRole ServerWithClientAuth
  TLSEnvironmentAdvancedParms
  {
    ClientAuthType SAFCheck
  }
  TLSKeyRingParms
  {
    Keyring CFZCIMServerRing
  }
}

# Common StartUp Group that new Rules may use
# Shows how each connection maps to policy
TLSGroupAction grp_StartUp
{
  TTLEnabled On
  Trace 0 # Log Errors and Info messages to syslogd
}
```

CIM server specific notes to the AT-TLS Policy parameters:

TTLRule: Jobname

Jobname identifies where this rule applies. In this example it is the started task job name. If you set up the connection this way, the configuration does not influence other parts of the system.

TTLRule: LocalPortRange

This property must match the HTTPS port definition of the CIM server.

SSL protected indication delivery

This topic shows an exemplary setup for the usage of RACF to deliver secured indications with AT-TLS.

To deliver secured indications from the CIM server to an indication end point means that the CIM server is establishing an encrypted connection to deliver indications. If a trusted relationship is established or not depends on the end point configuration.

In this case the CIM server is a client to the indication end point and therefore an outbound policy has to be specified with AT-TLS. To deliver secured indication the job name of the CIM server and the port specified in the indication handler destination property must match. An indication is defined by the application programmer so there must be an agreement between application programmer and the system programmer on which port secured indications are sent from the CIM server to the indication end points.

Sample Policy Agent policy for delivering of secured indications:

Example

```
TTLRule          CFZCIMServerRuleOutbound
{
    Jobname          CFZCIM*
    RemotePortGroupRef CFZCIMServerRemotePortGroup
    Direction        Outbound
    TLSGroupActionRef grp_StartUp
    TLSEnvironmentActionRef CFZCIMServerEnvActionOutbound
}

TLSEnvironmentAction CFZCIMServerEnvActionOutbound
{
    HandshakeRole      Client
    TLSKeyRingParams
    {
        Keyring CFZCIMServerRing
    }
}

PortGroup        CFZCIMServerRemotePortGroup
{
    PortRange
    {
        Port 5989
    }

    PortRange
    {
        Port 6000-7000
    }
}

# Common StartUp Group that new Rules may use
# Shows how each connection maps to policy
TLSGroupAction grp_StartUp
{
    TTLEnabled On
    Trace 0          # Log Errors and Info messages to syslogd
}
```

CIM server specific notes to the AT-TLS Policy parameters:

TTLRule: Jobname

Jobname identifies where this rule applies. In this example it is the started task job name. If you set up the connection this way, the configuration does not influence other parts of the system.

PortGroup

All indications which do have a port specified within the indication handler destination property and do match to any PortRange defined within the PortGroup are delivered secure via AT-TLS. If the destination property protocol is specified as https and no other port is specified, port 5989 will be used by the CIM server. So please ensure that always port 5989 is within a PortRange. In this example, all indications with port 5989 and port 6000-7000 are delivered secured.

Defining the CFZAPPL profile for the APPL class

If the APPL class for the security product is active, the CFZAPPL profile can be defined to allow only certain users to sign on to the CIM server. You can manage access to the CIM server application by a profile for CFZAPPL in the APPL class with an access list that contains only those users who are allowed to use the CIM server.

In general, you need not define a profile for CFZAPPL unless you have a generic profile (*) that prevents access to applications without a more specific profile.

For special purposes only, the CIM server can use the application ID 'OMVSAPPL', if the CIM server configuration property *enableCFZAPPLID* is set to false at server startup. In this case the system console message CFZ17204I indicates that the CIM server PassTicket validation is using application ID 'OMVSAPPL'. For changing configuration properties, see “Changing current configuration properties” on page 41 and “Changing planned configuration properties” on page 41.

Defining an application ID for PassTicket validation

The CIM server can alternatively validate a user ID and a *PassTicket* instead of a user ID and a password for authentication.

For more information about PassTickets, see *z/OS Security Server RACF Security Administrator's Guide*.

A PassTicket is validated against an application ID. The default application ID for the CIM server is 'CFZAPPL'.

To enable 'CFZAPPL' for the CIM server,

— Define 'CFZAPPL' profile in the PTKTDATA class in RACF.

Example:

```
SETROPTS CLASSACT (PTKTDATA)
SETROPTS RACLIST (PTKTDATA)
RDEFINE PTKTDATA CFZAPPL -
                SSIGNON(KEYMASKED(<key>))
SETROPTS RACLIST(PTKTDATA) REFRESH
```

where <key> is the 16 digit encryption key.

For special purposes only, the CIM server can use the application ID 'OMVSAPPL', if the CIM server configuration property *enableCFZAPPLID* is set to false at server startup. In this case the system console message CFZ17204I indicates that the CIM server PassTicket validation is using application ID 'OMVSAPPL'. For changing configuration properties, see “Changing current configuration properties” on page 41 and “Changing planned configuration properties” on page 41.

Additional setup for specific providers

Some providers as the network, job and cluster providers have additional security and setup requirements. Please see Chapter 5, “z/OS Management Instrumentation for CIM,” on page 79 for these providers. See also “Appendix D. Cluster / CoupleDataset / JES2-JES3Jobs Provider Setup” on page 193 for additional RACF setup.

Customizing the CIM server startup

There are two ways to start the CIM server: either from the started task procedure CFZCIM or from within a UNIX System Services shell. If you want to start the CIM server as started task, you must customize the JCL procedure CFZCIM and the according environment variable file */etc/wbem/cimserver.env*.

If you want to start the CIM server from a UNIX System Services shell or a remote UNIX session (telnet, ssh), the file *profile.add* contains the environment variables that you need to define. How to perform these tasks is described in the following subsections.

You should unambiguously decide for one of the possibilities and not mix them, because it is not possible to stop a CIM server started via UNIX System Services shell from the operator console using */s CFZCIM*.

Customizing the started task procedure CFZCIM

You can start the CIM server via started task procedure CFZCIM if you include CFZCIM in your PROCLIB concatenation. A sample of CFZCIM is shipped with the default PROCLIB.

The DDNAMEs STDOUT and STDERR in path */var/wbem/logs* are used to redirect the output from the console into the HFS-files *cimserver.out* and *cimserver.err*. If the started task is ended, the console output will be copied to the JCL job log in job steps two and three.

The DDNAME STDENV points to the hierarchical file system file containing environment variables required to run the CIM server. For running the CIM server as a started task, the environment variables are set in file *cimserver.env* located in the */etc/wbem* hierarchical file system directory. See “Customizing environment variables” on page 35 for details on how to set environment variables for the z/OS CIM server.

To run the CIM server with a user ID for which the security setup has been completed, either set up the STARTED class or use the started procedures table (ICHRIN03). For further details refer to *z/OS Security Server RACF Security Administrator's Guide*, chapter *Assigning RACF User IDs to Started Procedures*.

Example of the RACF commands required to setup the CIM server for the STARTED class:

Example:

```
SETROPTS RACLIST(STARTED)
RDEFINE STARTED CFZCIM.* STDATA(USER(<userID for CIM server>)
GROUP(<group CIM server userID belongs too>))

SETROPTS RACLIST(STARTED) REFRESH
```

Customizing the UNIX System Services shell for CIM server startup

To start the CIM server from a UNIX System Services shell, you must set the required environment variables which are contained in file */usr/lpp/wbem/install/profile.add*. You can add the contents of the profile to */etc/profile* to enable the CIM

server environment for all users of the UNIX System Services shell or to the individual `~/.profile` of each user. The environment variable `PEGASUS_HOME` should reflect your environment, that is, point to the installation directory of the CIM Runtime environment (usually, this is `/usr/lpp/wbem`).

Customizing environment variables

Environment variables are set in file *`cimserver.env`* if the CIM server will run as started task, or, if you use the CIM server from the UNIX System Services command prompt, they are set in UNIX System Services *`.profile`* in the home path of the user ID from which the CIM server is started.

Setting the trace variables is not required for normal operation.

Note that changes to the environment variables become effective only after a restart of the CIM server.

The environment variable file *`cimserver.env`* can be found in the hierarchical file system at *`/etc/wbem/cimserver.env`*. After installation, you can still find the originally shipped version in *`/usr/lpp/wbem/`*. Set the following environment variables contained in this file to start the CIM server:

PEGASUS_HOME

Must be set to the hierarchical file system directory where the CIM server is installed. By default this is *`/usr/lpp/wbem`*.

LIBPATH

Must include the CIM server's lib and provider hierarchical file system directory paths. By default this is set to

`/usr/lpp/wbem/lib:/usr/lpp/wbem/provider:/usr/lib`

OSBASE_TRACE

Defines the trace level for the z/OS OS management CIM instrumentation. Valid values range from 0 through 4, where 4 provides the most details.

OSBASE_TRACE_FILE

Defines the filename for the z/OS CIM instrumentation traces.

PATH Only for running the CIM server or any of the CIM server command-line utilities in UNIX System Services. Must include the CIM server's bin hierarchical file system directory path so that the CIM server's executable programs are automatically found when you enter the according command at the UNIX System Services command prompt. By default this is set to *`/usr/lpp/wbem/bin`*.

The following variables starting with `RMF_` only apply when RMF is installed and you use the `RMF @server` monitoring providers:

RMF_CIM_HOST

defines the target TCP/IP address or hostname of the z/OS MVS image on which the DDS responsible for this system is running.

RMF_CIM_PORT

defines the TCP/IP port number of the DDS (default: 8803).

RMF_CIM_TRACE

defines the trace level of the RMF CIM provider. Valid values range from 0 through 4, with 0 providing no trace and 4 providing all information possible.

RMF_CIM_TRACE_FILE

defines the file name for storing the trace data for the z/OS RMF CIM instrumentation.

RMF_CIM_BENCH

used for performance benchmarks, for example, to identify the response time of the underlying RMF infrastructure. If this variable is set to 1, the RMF CIM provider will print some benchmarking information on various RMF operations, suitable for RMF development.

RMF_INDICATION_RESTTIME

specifies the time in seconds that a provider should sleep after checking an indication subscription and before checking the next one, in order to reduce system load. The default is one second.

RMF_INDICATION_POLLING_INTERVAL

specifies the time interval in which every single subscription is checked at most once. The default is 100 seconds.

The following variables starting with WLM_ only apply when WLM is installed and you use the WLM providers:

WLM_CIMPROVIDER_TRACE_FILE

defines the output file name for z/OS WLM provider traces. The default trace file is `/var/wlmptraces.trc`.

WLM_CIMPROVIDER_TRACE_LEVEL

defines the trace level for the z/OS WLM provider. Valid values range from 0 through 5, where 5 provides the most details. The default is 0, meaning that no trace is written.

Selecting a WLM service class for z/OS CIM priority

If you plan to use the z/OS CIM server as part of your monitoring or management infrastructure, it should run at a priority higher than the work to be managed. You should classify the CIM server into a single period service class with a velocity goal at an appropriate importance level.

Provider-based authorization model

The CIM server offers an optional authorization check. This check is optionally performed on a per provider basis, meaning that a RACF profile in class WBEM can be related to a single provider library. Correlation between a provider and a RACF profile occurs during provider registration by the addition of a property in the **PG_Provider** class.

The provider based authorization is defined by the vendor of a provider rather than by the CIM server administrator. Therefore, specific RACF requirements will need to be documented on a per provider base.

In addition, this check is done depending on the type of CIM request that is performed. For example, in order to access CIM operations that change the states of objects, WRITE access to the RACF profile defined for a provider is required. Schema manipulation is only available to users with CONTROL access to SAF profile CIMSERV in class WBEM.

Table 3 on page 26 lists the type of access required for the different types of CIM operations.

Starting with z/OS 1.8, the OpenPegasus CIM class *PG_Provider* contains a string type attribute named *SecurityAccessProfile*. Providers that register with an instance of class *PG_Provider* containing the *SecurityAccessProfile* property, must specify their RACF profile with this property in order to define it to the system. In addition, requesting users must have the according level of authorization for the named profile.

Class *PG_Provider* contains the z/OS-specific property *SecurityAccessProfile*. This property defines the name of a z/OS security server's profile in the CIM server's WBEM class that will be checked for a requestor's access before a request is routed to this provider. Depending on the type of the CIM operation, a different level of access for the security profile is required as listed in Table 3 on page 26.

Providers existing since z/OS 1.7 are not affected by this new feature and will continue to work as before. However, if you want to let these existing providers exploit this new feature, you must remove (unregister) them using the *cimprovider* utility and then re-register them with the added security profile name in property *SecurityAccessProfile* in their provider registration MOF file. The existence of a specified security profile will not be checked during provider registration. Instead, this check is done during runtime, when a request is received for the according provider.

Running providers in a designated user context

When an invocation is caused by an external CIM operation, by default the provider is processed in the context of the **requestor's user ID**. As the provider runs under the identity of the requestor's user ID, all resource access authorization occurs against this user ID. So the requestor must be authorized for all resources that a provider accesses during a request.

To avoid that a CIM client user ID has global access to all the resources that a provider uses for gathering data, starting with z/OS 1.10, a provider can be registered with a **designated user ID**. The designated user ID specifies a separate security context which is used to process the provider. The designated user ID must be authorized to access all the resources accessed by the provider. Instead of directly using a requestor's user ID when accessing the resource, the provider code now has to perform custom authorization checks based on the requestor's user ID, to prevent unauthorized access to resources. The security definitions for the designated user ID should be similar to those of regular client users, as described in "Switching identity (surrogate)" on page 27.

The properties *UserContext* and *DesignatedUserContext* of CIM class *PG_ProviderModule* specify the provider's processing context. You can specify the values for these properties in the provider registration MOF file for each provider module. By default, it is installed at `/usr/lpp/wbem/provider/schemas/...`. For further details, see "PG_ProviderModule" on page 188.

Generally, the vendor of a provider (implementing a certain CIM class) defines if a provider should run under a designated user context and also supplies the according documentation describing the specific setup steps.

Chapter 4. How to work with the CIM server

The CIM server is used to receive the clients' requests, collect the requested metrics/data from the managed system and return the results to the client. This chapter contains all information required for working with the CIM server on z/OS. It describes the following topics:

- "Controlling the CIM server"
- "CIM server command-line utilities and commands" on page 52
- "Backing up the CIM server repository" on page 73
- "Sample client applications" on page 74
- "Troubleshooting" on page 74

While you must set up the CIM server only once to make it ready to use, you can configure your CIM server environment as often as you want during operation to best meet your requirements. To help you in configuring the CIM server you can use the commands described in "CIM server command-line utilities and commands" on page 52. If you run into problems while setting up or using the CIM server you can find information for problem solving in "Troubleshooting" on page 74.

Controlling the CIM server

The CIM server provides the ability to set a number of configuration options.

Many tasks and operations for the CIM server are performed under z/OS UNIX[®] System Services, ideally within a telnet session.

For a user to be able to use the command line tools, the UNIX System Services environment has to be set up as described in "Customizing the UNIX System Services shell for CIM server startup" on page 34.

Starting and stopping the CIM server

Start the CIM server from either the UNIX System Services command prompt or as a started task, as described in the following sections.

Running the CIM server from the UNIX System Services command prompt

To start the CIM server from the UNIX System Services shell, type the `cimserver` command at the command prompt of a z/OS UNIX System Services shell. To run the CIM server runtime environment in its own address space, set the environment variable `BPX_SHAREAS=NO` in your z/OS UNIX System Services shell beforehand.

Stop the CIM server by entering `cimserver -s` at the command line.

To let the `cimserver` command work from a UNIX System Services shell, make sure you have completed the configuration steps described in "Customizing the UNIX System Services shell for CIM server startup" on page 34.

Running the CIM server as started task

The standard way to start the CIM server on z/OS is through the started task CFZCIM. Thus you can start the CIM server from the z/OS console by entering

S(TART) CFZCIM

Before starting the CIM server for the first time through CFZCIM, customize the procedure to match the correct installation paths in the DD statements and also update the *cimserver.env* file installed in */etc/wbem* to match the correct installation paths for the CIM server. When using the default installation directory */usr/lpp/wbem*, no changes to either CFZCIM nor *cimserver.env* are required. For information on how to customize CFZCIM and *cimserver.env* refer to “Customizing the CIM server startup” on page 34.

When the CIM server was started through CFZCIM, it can also be stopped from the console by entering

(STO)P CFZCIM

After a successful start of the CIM server, the following message is shown on the console and issued to the syslog:

CFZ10030I: Started CIM server version 2.7.1 for z/OS.

Running providers in their own address space

Since z/OS 1.8, the *Out-of-Process* provider feature from OpenPegasus is supported by the z/OS CIM server and is enhanced to comply with the guidelines for z/OS security. The *Out-of-Process* feature manages providers in separate address spaces rather than loading and calling provider libraries directly within the CIM server process.

A new configuration property, *forceProviderProcesses* (see “Advanced configuration properties” on page 43), is used to enable/disable the *Out-of-Process* support for all active providers. By default, *forceProviderProcesses* is set to ‘false’, so that the existing in-process provider implementation is not affected. When *forceProviderProcesses* is set to ‘true’, providers will be executed within a newly created Provider Agent process.

In addition to the support provided in OpenPegasus for *Out-of-Process* providers, for z/OS a number of options are introduced for the provider registration that help to control the security model of the Provider Agent in which a provider will be executed. Depending on which options are set during provider registration, providers will execute in different types or instances of Provider Agents.

Two different types of Provider Agents are supported starting with z/OS 1.8, one that runs APF authorized and provides an execution environment for APF authorized providers. The other (default) type of Provider Agent runs unauthorized. To control this behavior, a property *RunAuthorized* is provided for the *PG_ProviderModule* registration class. When set to ‘true’, a provider will run in the authorized Provider Agent address space. By default, *RunAuthorized* is set to ‘false’. Though this is an officially supported feature for z/OS, it is not recommended to be used unless it is absolutely required for a provider.

Providers for which *RunAuthorized* is set to ‘true’ during registration, must be APF authorized themselves, otherwise they will not be loaded by the Provider Agent to avoid that the Provider Agent address space is marked dirty.

The granularity for which the *RunAuthorized* property can be set, is the Provider Module (= shared library). This means that when multiple providers are physically packed in one module, they all will run with the same security model.

There is no limit to the number of provider modules that may register with a CIM server which supports *Out-of-Process* providers. There is also no limit to the number of providers in a provider module. However, one *Out-of-Process* Provider Agent can correspond to one or more provider modules, depending on the setting of the *ShareAS* property. When *ShareAS* is set to 'true' in the provider module definition, the according provider code runs in one of the pthreads within an *Out-of-Process* address space. There will be other CIM providers running on different pthreads within this address space, too. If *ShareAS* is set to 'false', the designated CIM provider code runs in its own dedicated Provider Agent address space. No other CIM providers (that is, a different CIM provider other than the designated CIM provider) will be allowed to run in this address space.

Note: Both the *RunAuthorized* as well as the *ShareAS* property are z/OS-specific extensions that are not available in OpenPegasus on other platforms.

CIM server configuration

The default configuration setting for the CIM server should work for the majority of environments. However, the CIM server provides command-line utilities to change the default configuration settings (see "CIM server command-line utilities and commands" on page 52).

Changing current configuration properties

Use the `cimconfig` UNIX System Services shell command (see "cimconfig" on page 55) or the `MODIFY` console command (see "MODIFY console command" on page 72) to dynamically change the current configuration properties of the CIM server.

Current configuration properties that you can change are:

- the *shutdownTimeout* property,
- the logging properties, and
- the tracing properties.

You can update the current configuration while the CIM server is running.

Using the `cimconfig` command without the `-p` option or the `MODIFY` console command without the `PLANNED` option results in a non-permanent change. After a restart of the CIM server, these changes will be reset to the planned configuration values. For making permanent changes, you must change the planned configuration values.

Changing planned configuration properties

To change values for the planned configuration properties, use the `cimconfig` UNIX System Services shell command with the `-p` option or the `MODIFY` console command with the `PLANNED` option.

cimconfig

The use the `cimconfig` command is independent of whether the CIM server is running or stopped. If you change the planned configuration properties while the CIM server is running, those changes do not take effect until the CIM server is restarted. Then the planned configuration properties become the current configuration properties.

MODIFY The CIM server must be running to use the `MODIFY` console command. When you use the `MODIFY` console command with the `PLANNED` option, those changes do not take effect until the CIM server is restarted. Then the planned configuration properties become the current configuration properties.

If during the CIM server startup a deprecated configuration property is detected in file `/etc/wbem/cimserver_planned.conf`, then the CIM server issues warning message CFZ02300I and continues to start. Starting with z/OS CIM server V1R9, the properties `logdir`, `enableNamespaceAuthorization` and `httpAuthType` are deprecated and should be removed from the property file to avoid this message.

CIM server configuration properties

Using the `cimconfig` command, you can change configuration properties from the following categories:

- “Basic configuration properties”
- “Advanced configuration properties” on page 43

Column *dynamic* Y/N in the tables in these sections tells you whether you can dynamically change the property.

For those properties which you cannot dynamically change, you must use

- either the `-p` parameter of the `cimconfig` command to indicate your change, and then you must stop and restart the CIM server (see “`cimconfig`” on page 55),
- or the `PLANNED` option of the `MODIFY` console command (see “`MODIFY` console command” on page 72).

Basic configuration properties

Table 4. CIM server basic configuration properties

Option name	Description	Default value	dynamic Y/N
<code>daemon</code>	The foreground/background process property. Set <code>Daemon</code> to ‘false’ to run the CIM server as foreground process or as a started task.	true For running the CIM server as a started task, this option is set to ‘false’.	N
<code>enableHttpConnection</code>	The HTTP connection to the CIM server. Enables and disables connections to the CIM server over HTTP. When turned off only local connections are accepted.	true	N
<code>enableHttpsConnection</code>	The HTTPS connection to the CIM server. Enables and disables secure connections to the CIM server via HTTPS. Note that it is not sufficient to turn on this option, but you must also enable an SSL connection through the AT-TLS feature at the z/OS Communications Server as described in “Configuring the CIM server HTTPS connection using AT-TLS” on page 27.	false	N
<code>httpPort</code>	The port to listen for HTTP requests. It is not recommended to change this value.	5988	N
<code>httpsPort</code>	The port to listen for HTTPS requests. AT-TLS should be configured to use this port. It is not recommended to change this value.	5989	N

Advanced configuration properties

Table 5. CIM server advanced configuration properties

Option name	Description	Default value	dynamic Y/N
enableAssociationTraversal	The CIM server traverses associations along the inheritance tree. Disabling this increases performance, but some associations become not resolvable.	true	N
enableAuditLog	When this option is set to true, the CIM server is writing SMF records 86. For details see “Audit logging with SMF record 86” on page 47.	false	Y
enableCFZAPPLID	If the property is set to true, the application ID (APPLID) is ‘CFZAPPL’. Otherwise the used application ID is ‘OMVSAPPL’ and the message CFZ17204I is issued at CIM server startup.	true	N
enableIndicationService	‘true’ means the indication service is enabled. ‘false’ will disable the indication service.	true	N
enableRemotePrivileged UserAccess	The remote privilege for users. Enables and disables remote access for users with UID 0.	false	N
forceProviderProcesses	When this option is set to ‘true’, providers will run in one or more separate address spaces. Note: In z/OS 1.7, an option called <i>enableProviderProcesses</i> had been documented and indicated as not being supported. Starting with z/OS 1.8, this option has been renamed to <i>forceProviderProcesses</i> and is supported.	false	N
logLevel	The detail level for logging. Possible values are TRACE, INFORMATION, WARNING, SEVERE or FATAL. TRACE returns the most detailed log information (see also “Configure logging” on page 45).	INFORMATION	Y
messageDir	The message bundle directory. Do not change the default.	msg	N
providerDir	The name of the directory where the provider libraries reside. You can specify multiple directories here, separated by a colon (‘:’). Since the CIM server has its own set of providers, its lib directory always needs to be present in the list of provider directories. When adding new provider directories, it is also recommended to update the LIBPATH environment variable according to the new values of providerDir. This is required, because a provider may need other supplemental DLLs, which the CIM server is not aware of and therefore would otherwise fail to load. Note: You must provide the full path for all directories when changing the default.	lib:provider	N

Table 5. CIM server advanced configuration properties (continued)

Option name	Description	Default value	dynamic Y/N
repositoryDir	The name of the directory for the repository.	/var/wbem/ repository	N
repositoryIs DefaultInstanceProvider	The CIM server repository serves as the default provider for CIM instances when no dynamic provider has been registered for a CIM class.	true	N
shutdownTimeout	The timeout value in seconds that the CIM server uses to wait for the shutdown process to complete. This value includes terminating active providers.	30 seconds	Y
socketWriteTimeout	The timeout value in seconds that the CIM server uses to wait for a client to receive data from the socket. After the timeout the CIM server will close the socket.	20 seconds	N
slp	The CIM server uses the SLP Protocol to announce itself over the network.	true	N
traceComponents	This option turns on and off tracing for single components of the CIM server. Tracing should only be used for debugging purposes and is, by default, disabled. If this parameter is not set, tracing is turned off. If this parameter is defined, it specifies the component(s) that you want to trace. Setting this value to ALL enables tracing for all components. For more information refer to section "Configuring CIM server tracing," which also lists the valid components.		Y
traceFilePath	The name of the directory where the trace file is written.	/tmp/cimserver.trc	Y
traceLevel	The trace level. Controls the level of detail for tracing. Choose one of the following trace levels: level 1 Function entry and exit level 2 Basic flow, trace messages, low data detail level 3 Interfunction logic flow, medium data detail level 4 All information, high data detail Note: This does not include tracing for the providers. See also "Configuring CIM server tracing."	1	Y

Configuring CIM server tracing

Tracing should only be used for debugging purposes and is, by default, disabled (that is, *traceComponents* is not set). You can, however, enable the tracing mechanism. Use the `cimconfig` command or the `MODIFY` console command to

specify the components that you want to trace with the *traceComponents* parameter. Specify the trace level with the *traceLevel* parameter. Choose from one of the following trace levels:

level 1	Function entry and exit
level 2	Basic flow, trace messages, low data detail
level 3	Interfunction logic flow, medium data detail
level 4	All information, high data detail

Trace data is saved in the file specified by the *traceFilePath* configuration parameter. By default, the *traceFilePath* parameter is set to the */tmp/cimserver.trc* file.

To specify the component or components that you want to trace, use the *traceComponents* parameter. The components that you can specify with this parameter are the following:

ALL	Authentication	Authorization
BinaryMessageHandler	CimData	CIMExportRequestDispatcher
CIMOMHandle	CMPIProvider	CMPIProviderInterface
Config	ControlProvider	CQL
DiscardedData	Dispatcher	ExportClient
Http	IndicationFormatter	IndicationGeneration
IndHandler	IndicationHandlerService	IndicationReceipt
IndicationService	IndicationServiceInternal	IPC
L10N	Listener	MessageQueueService
ObjectResolution	OsAbstraction	ProviderAgent
ProviderManager	Registration	Repository
Server	Shutdown	SSL
StatisticalData	Thread	UserManager
WQL	XmlIO	XmlParser
XmlReader	XmlWriter	

Using ALL instead of one or more specific component names traces all available components. If the *traceComponents* parameter is not set to any component, tracing is off regardless of the *traceLevel* parameter setting.

You can modify the trace configuration parameters by using the *cimconfig* command or the *MODIFY* console command while the CIM server is running. For example, to set the trace level to trace all information with high data detail in the *Thread* and *ProviderManager* components, go into the UNIX System Services shell and type the following commands:

```
cimconfig -s traceLevel=4
cimconfig -s traceComponents=Thread,ProviderManager
```

To disable all tracing, type the following command:

```
cimconfig -s traceComponents=
```

For more information about the *cimconfig* command, see “*cimconfig*” on page 55. For more information about the *MODIFY* console command, see “*MODIFY* console command” on page 72.

Configure logging

The CIM Server sends log messages to the z/OS system console as well as to the z/OS Communications Server's system logger (syslog) daemon. While logging to

the z/OS system console will always occur, the syslog daemon has be configured as described in *z/OS Communications Server: IP Configuration Reference* and *z/OS Communications Server: IP Configuration Guide*.

Generally logging for the CIM server is enabled and cannot be turned off. However, you can configure the level of logging by adopting the *logLevel* configuration option, setting it to one of the following:

TRACE (only through syslogd!)
INFORMATION
WARNING
SEVERE
FATAL

TRACE returns the most detailed trace information, while FATAL only produces log messages for fatal errors. The default setting for *logLevel* is INFORMATION and should not be changed unless there is a specific need for a more or less detailed logging. TRACE level logging causes an excessive amount of messages to be generated and is only written to the syslog daemon. TRACE level log messages are never sent to the z/OS system console.

You can change the logging level using the *logLevel* configuration property. For example, you can go into the UNIX System Services shell and type the following command while the CIM server is running:

```
cimconfig -s logLevel=INFORMATION
```

For more information about the `cimconfig` command, refer to “`cimconfig`” on page 55.

Using the syslog daemon for CIM server logging

The z/OS CIM server will connect to the syslog daemon and send all of its log messages to it, where the filtering according to the *logLevel* configuration property applies as described above. Therefore no messages will be submitted to the syslog daemon which have a higher log level than what’s specified in the current value of the *logLevel* configuration property.

Messages that go to the syslog daemon are prepended with an according z/OS message number, which is either one of the generic CFZ00001E, CFZ00002W or CFZ00004I messages followed by a PG5xxxxx message number, or one of the directly mapped z/OS specific CFZxxxxx message numbers.

Syslog messages from the z/OS CIM server will have an identifier of “CFZCIM” and also contain the CIM server’s process ID.

The log levels of the z/OS CIM server are mapped to the following syslog levels:

Table 6. Log and syslog levels

FATAL	-->	LOG_ERR
SEVERE	-->	LOG_ERR
WARNING	-->	LOG_WARNING
INFORMATION	-->	LOG_INFO
TRACE	-->	LOG_DEBUG

If the syslog service is not properly configured for CIM, or if the syslog daemon has not been started, this level of logging will not occur for CIM. Logging then only occurs to the z/OS console. Since the logging to the z/OS console will contain

all messages except for level TRACE, configuring the syslog service is not really required for CIM unless TRACE level logging is needed or if the log messages should be captured in a separate file.

Following is a sample syslog configuration file (*/etc/syslog.conf*) entry for CIM, which tells the syslog daemon to create log files for the CIM server, similar to the files the CIM server had previously created in z/OS 1.7 and z/OS 1.8 by its own implementation:

Example:

```
...
*.CFZ*.*.debug /var/wbem/logs/cimserver_%Y.%m.%d.syslog
...
```

When configured like this, the CIM server log messages will appear in the format shown by the following example:

Example:

```
Nov  7 12:48:38 BOECFZ1 CFZCIM[33557318]:
CFZ10025I: The CIM server is listening on HTTP port 5,988.
Nov  7 12:48:38 BOECFZ1 CFZCIM[33557318]:
CFZ10028I: The CIM server is listening on the local connection socket.
Nov  7 12:48:38 BOECFZ1 CFZCIM[33557318]:
CFZ10030I: Started CIM Server version 2.6.1
Nov  7 12:48:38 BOECFZ1 CFZCIM[33557318]:
CFZ12533I: The CIM server failed to register with ARM using
element name CFZ_SRV_PEG2: return code 0x0C, reason code 0x0168.
Nov  7 12:49:01 BOECFZ1 CFZCIM[33557318]: CFZ10031I: CIM Server stopped.
```

Except for the CIM server's *logLevel* property, all configuration now occurs through the syslog service as described in *z/OS Communications Server: IP Configuration Reference* and *z/OS Communications Server: IP Configuration Guide*.

Note: Configuration of the syslog daemon for specific processes/daemons is done based on the jobname of the process writing the logs. While this is always CFZCIM when running the CIM server as started task, it is also possible to have the CIM server started from the UNIX System Services command prompt. In that case the jobname of the CIM server will be that of the user ID that it was started from, rather than CFZCIM. In order to set the jobname of the CIM server correctly to CFZCIM when started from the command prompt, environment variable `_BPX_JOBNAME` should be set to "CFZCIM". Otherwise it will be difficult to create a syslog configuration for the CIM server.

Audit logging with SMF record 86

The CIM server can file audit log records to SMF record 86. These records contain information about authentication, configuration, provider status, and CIM operations. For details of SMF record 86, see *z/OS MVS System Management Facilities (SMF)*.

To enable writing audit SMF record 86, modify the SMF, the CIM server, and the security configuration:

SMF configuration:

- Ensure that record 86 is part of your active SMF configuration SMFPRMXX parmlib member.

CIM server configuration:

- To enable the CIM server to write audit records, set the configuration property **enableAuditLog** to true.

When recording is switched on, the current CIM server configuration and the status of the currently loaded providers is recorded. To disable recording, set the configuration property to false. This property can be changed dynamically during CIM server runtime.

Security configuration:

- In order to write SMF records, the CIM server needs at least READ access to the BPX.SMF profile of the FACILITY class at your SAF product.

Example for RACF:

```
RDEFINE FACILITY BPX.SMF UACC(NONE)
PERMIT BPX.SMF CL(FACILITY) ACCESS(READ) ID(CFZADM)
SETOPTS RACLIST(FACILITY) REFRESH
```

If the CIM server audit logging is enabled, but SMF does not collect SMF record 86 or subtypes, or SMF is not enabled at all, no records are written.

Backing up the CIM server configuration

After you have set up and configured the z/OS CIM server properly as described in the subsections of Chapter 3, “First-time CIM server set up,” on page 21 and “Controlling the CIM server” on page 39, you should backup the following CIM server property configuration files located in */etc/wbem*:

- **cimserver_planned.conf** contains planned values, not yet in effect and that are not defaulted. They will be picked up at the next CIM server restart.
- **cimserver.env** contains the environment variables for the started task CFZCIM.

Note: Do not edit configuration files directly. Use the **cimconfig** command to change the property values in the files.

How to backup the CIM server repository is described in “Backing up the CIM server repository” on page 73.

Automatic restart of the CIM server

Since the CIM server serves as a primary system management interface for a system, it should be continuously available.

To support the CIM server availability, startup and shutdown messages are logged to the z/OS console to be used with a systems management program such as IBM Tivoli System Automation.

Starting with z/OS 1.9, the z/OS CIM server is also enabled for the Automatic Restart Manager (ARM). This section only documents the information that is relevant for ARM with the CIM server. For details on ARM refer to *z/OS MVS Setting Up a Sysplex*.

The CIM server needs no additional configuration to use ARM. It will always try to register itself to ARM. When ARM is active and the CIM server is authorized to register with ARM, then success message CFZ12532I will be displayed in the system log (see “z/OS-specific messages” on page 158). Otherwise, information message CFZ12533I will be displayed in the system log to inform you that the CIM server is not registered to ARM.

You can use ARM only for started task procedures or batch jobs. So if you start the CIM server from the UNIX System Services shell, you also get the message CFZ12533I. If you do not plan to use ARM, you can ignore this message, which is issued every time when the CIM server is started.

The CIM server issues the registration and the ready request after a successful bind to the communication socket/s (HTTP, HTTPS, and/or Local). It will be deregistered from ARM during its normal shutdown procedure. In all other cases, the CIM server remains registered and is restarted based on the active ARM policy.

In a sysplex, you can start only one CIM server per OS image. Therefore ARM can only be used to restart after an application ABEND and not for cross-system restarts. You must use other facilities to start the CIM server during an IPL.

ARM security considerations

If the CIM server is configured to use ARM in a sysplex, you must ensure that the XCF address space has the proper authorization to perform a restart. ARM must be able to issue operator commands from the XCF address space (XCFAS) to start the CIM server.

The CIM server is not running in supervisor mode. Therefore, the user ID running the CIM server must have proper SAF authorization to be allowed to register to ARM. Therefore the user ID running the CIM server also needs the SAF authorization for UPDATE access to the following FACILITY class resource:

Example:

```
IXCARM.DEFAULT. CFZ_SRV_<system_name>
```

Here is an example for entitling the CIM server user ID CFZADM to register the CIM server for all machines within a sysplex using RACF:

Example:

```
SETROPTS CLASSACT(FACILITY)
SETROPTS RACLIST (FACILITY)

RDEFINE FACILITY IXCARM.DEFAULT. CFZ_SRV_* UACC(NONE)

PERMIT IXCARM.DEFAULT.CFZ_SRV_* CLASS(FACILITY) +
      ID(CFZADM) ACCESS(UPDATE)

SETROPTS RACLIST(FACILITY) REFRESH
```

ARM policy considerations

The CIM server has the following requirements for exploiting the ARM restart policy:

- The ARM element name used for the CIM server is CFZ_SRV_<system_name>, where <system_name> is substituted by the value of the system symbol SYSNAME.
- The CIM server can only be restarted on the system where it failed. A cross-system restart within a sysplex is not possible. Therefore the termination type has to be ELEMTERM.
- The restart occurs through starting the CIM server started task procedure CFZCIM.

The sample JCL CFZARMP is installed to the SYS1.SAMPLIB during SMP/E z/OS installation of the CIM component.

```

//CFZARMP JOB MSGCLASS=C,MSGLEVEL=(1,1),USER=XXXXXX,NOTIFY=XXXXXX
//*****
/*
/* PROPRIETARY STATEMENT:
/* Licensed Materials - Property of IBM
/* 5694-A01
/* (C) Copyright IBM Corp. 2005, 2007
/*
/* STATUS=HPG7750
/*
/* DESCRIPTIVE NAME:
/*
/* SAMPLE JCL TO UPDATE THE ADMINISTRATIVE POLICY DATA FOR CIM
/* SERVER IN THE COUPLE DATA SET FOR ARM (AUTOMATIC RESTART MANAGER)*
/*
/* NOTES:
/*
/* 1. SYSPRINT DD IS A REQUIRED DD STATEMENT FOR THE UTILITY
/* OUTPUT.
/* 2. SYSIN DD IS A REQUIRED DD STATEMENT FOR THE UTILITY
/* CONTROL STATEMENTS.
/* 3. DATA TYPE(ARM) STATEMENT IS REQUIRED TO SPECIFY WHAT TYPE
/* OF COUPLE DATA SET IS TO BE UPDATED.
/* 4. REPLACE %COUPLDS% AND %VOLSER% WITHIN THE JCL TO SPEVIFIY
/* THE DATA SET NAME AND VOLSER OF THE COUPLE DS
/* 5. REPORT KEYWORD IS OPTIONAL. WHEN REPORT(YES) IS SPECIFIED,
/* AN ARM ADMINISTRATIVE POLICY REPORT WILL BE GENERATED IN
/* THE OUTPUT. THE DEFAULT VALUE FOR REPORT IS YES.
/* 6. REPLACE KEYWORD IS OPTIONAL. WHEN REPLACE(YES) IS SPECIFIED
/* FOR A POLICY, THE POLICY WILL BE REPLACED IF IT ALREADY
/* EXISTED IN THE COUPLE DATA SET.
/* IF REPLACE(NO) IS SPECIFIED FOR AN EXISTING POLICY,
/* THE UPDATE JOB WILL BE FAILED AND NO CHANGES WILL BE MADE
/* TO THE COUPLE DATA SET.
/* 7. TO DELETE AN EXISTING POLICY IN A COUPLE DATA SET,
/* INCLUDE THE FOLLOWING LINE IN THE SYSIN DD CARD:
/* DELETE POLICY NAME(CFZARMP0)
/* WHERE POLNAME IS THE NAME OF THE POLICY TO BE DELETED.
/*
/******
//STEP1 EXEC PGM=IXCMIAPU
//STEPLIB DD DSN=SYS1.MIGLIB,DISP=SHR
//SYSPRINT DD SYSOUT=A
//SYSABEND DD SYSOUT=A
//SYSIN DD *

DATA TYPE(ARM)
REPORT(YES)

DEFINE POLICY NAME(CFZARMP0) REPLACE(YES)

RESTART_GROUP(CFZCIMRESGRP)
/* List all servers the CIM Server can be started on */
TARGET_SYSTEM(SYS1)
/* Wait 10 sec before restarting to free resources */
RESTART_PACING(10)

ELEMENT(CFZ_SRV_*)
RESTART_ATTEMPTS(3,300)
RESTART_TIMEOUT(300)
READY_TIMEOUT(300)
/* cross-system restart are not allowed. */
/* No restart after system failure */

TERMTYPE(ELEMTERM)
RESTART_METHOD(ELEMTERM,STC,'S CFZCIM')

```

CIM server command-line utilities and commands

The CIM server includes a set of command-line utilities that you can use to control or change the CIM server environment. During normal use, you should rarely need to use these commands.

You must run all of the command-line utilities from a z/OS UNIX System Services shell. The command-line utilities need the same environment setup as described in “Controlling the CIM server” on page 39. Users of these command-line utilities must have CONTROL access to profile CIMSERV in class WBEM.

Note: All of these utilities generate ASCII output. Without proper setup of your shell these utilities will show unreadable output. If you have problems with viewing this output, see “ASCII-EBCDIC conversion” on page 75.

The commands include:

cimmof/cimmofl

These commands are used to compile provider registrations and to compile Common Information Model (CIM) class descriptions written in the managed object format (MOF) language. The compiled information is put into the class schema stored in the repository.

`cimmofl` is a version of `cimmof` that does not use the CIM server. The CIM server must be stopped before using this command. The usage of `cimmofl` is not recommended, since it bypasses the CIM server and directly manipulates files in the file system (see “`cimmof/cimmofl`” on page 53).

cimconfig

This command configures the options for the CIM server. Depending on the property being configured, the CIM server may need to be restarted after using this command (see “`cimconfig`” on page 55).

cimprovider

This command enables or disables a registered provider, primarily during testing. The CIM server must be running to use this command (see “`cimprovider`” on page 57).

cimcli This command lets you perform CIM client requests/operations. It implements most of the DMTF CIM operations.

Each execution of `cimcli` invokes a CIM operation with the corresponding parameters equivalent to the CIM operations defined in the *CIM Operations over HTTP* specification. Additionally, the `cimcli` command-line interface implements a number of other specific operations that support testing and querying CIM servers, including operations to query for namespaces and to get all instances in a namespace (see “`cimcli`” on page 59).

cimsub This command lets you manage CIM indications on the local CIM server. The command can list, enable, disable and remove indication subscriptions, filters and handlers (see “`cimsub`” on page 69).

MODIFY console command

Like the `cimconfig` command, the MODIFY console command configures the options for the CIM server while the CIM server is running. Depending on the property being configured, the CIM server may need to be restarted after using this command (see “MODIFY console command” on page 72).

Note: The **wbemexec** utility is also included with CIM. It is used to directly send CIM-XML requests to a CIM server. However, this tool is not supported, but just supplied on an 'as-is-base'.

You can specify most options provided by the utilities in two ways:

- a short form introduced by a single dash, for example `-f<file>`
- a long form introduced by a double dash, for example `--file=<file>`

To understand the syntax of the description of the command-line utilities, see "How to read syntax diagrams" on page x.

cimmof/cimmofl

Description

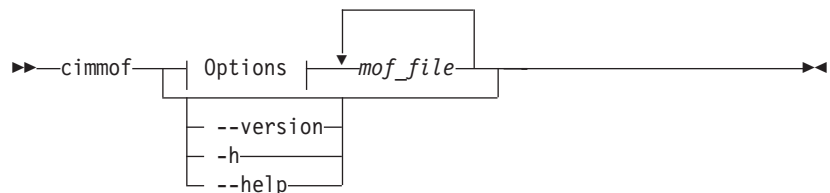
These commands are used to compile provider registrations and to compile CIM class descriptions written in the MOF language and store the information in the repository. For `cimmof`, the CIM server must be started before using this command. `cimmofl` runs without using the CIM server. This version of the MOF compiler does only limited error checking, may handle instance operations incorrectly, and does not protect against concurrent access to the CIM repository. Therefore, `cimmof` is the recommended MOF compiler.

The CIM server MOF compiler is a command-line utility that compiles MOF files (using the MOF format defined by the DMTF CIM Specification) into a CIM server repository. It allows compiling from structures of MOF files using the `include #pragma` and can either compile into a CIM server repository or check the syntax of the MOF files. The compiler requires that the input MOF files are in the current directory or that a fully qualified path is given. MOF files that are included using the `include #pragma` must be in the current directory or in a directory specified by a `-I` command-line switch. For using `cimmofl` the binary option has to be specified.

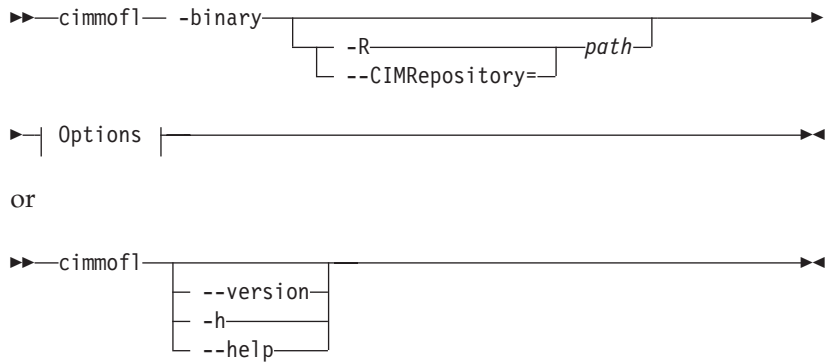
For using the `cimmof/cimmofl` commands against the CIM server namespaces (`root/PG_Internal`, `root/PG_InterOp`), a user needs to have CONTROL access to profile CIMSERV in class WBEM.

Syntax

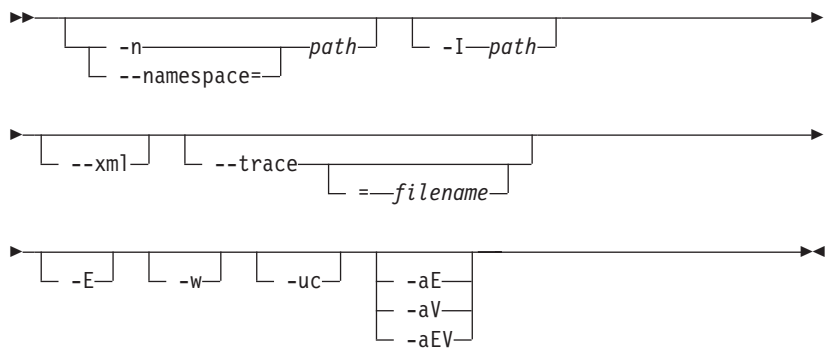
cimmof



cimmofl



Options:



Options

mof_file

Specifies the MOF file or MOF files to compile.

--version

Displays the CIM server version.

-h, -help, or no specified option

Prints out a usage message with command definitions.

-binary

Binary repository load.

-R path, --CIMRepository=path

If specified, this overrides the current repository path used by the CIM server. Specify an absolute or relative path.

-I path Specifies a path to the included MOF files.

-n path, --namespace=path

Overrides the default CIM repository namespace path. The default is root/cimv2.

--xml Generates XML to standard output. This option does not update the repository.

--trace, --trace=filename

Writes trace information to a file. If *filename* is omitted, the output destination is standard output. Those files are written with ASCII encoding.

- E Performs a syntax check on the input. This option does not update the repository.
- w Suppresses warning messages.
- uc Allows the update of an existing class definition. This option lets you update a leaf class. It does not allow updates of superclasses or classes that have subclasses.
- aE Allows the addition or modification of classes with the experimental qualifier.
- aV Updates a class that results in a version change. The version must be specified in a valid format. The format is m.n.u where m is major version, n is minor release and u is update. For example, 2.7.0 is a valid format for CIM Schema 2.7.0. If the input class has the same version as the class in the repository, the class is not updated.
- aEV Allows both Experimental and Version Schema changes.

Example

cimmof -w -I./myDir myDir/CIM_Schema211.mof

In this example, the managed object format (MOF) file that is located in directory myDir with the name CIM_Schema211.mof is compiled into the default namespace root/cimv2.

CIM_Schema211.mof includes #pragmas for other MOF files that are also in the myDir directory. Therefore an include (-I) option is required for the myDir directory. The -w option suppresses warning messages.

cimconfig

Description

Use the cimconfig command to manage CIM server configuration properties. You can get, set, unset, or list these properties. See “Controlling the CIM server” on page 39 for more information.

You can use the cimconfig command to set the current or planned configuration properties of the CIM server.

Current configuration properties:

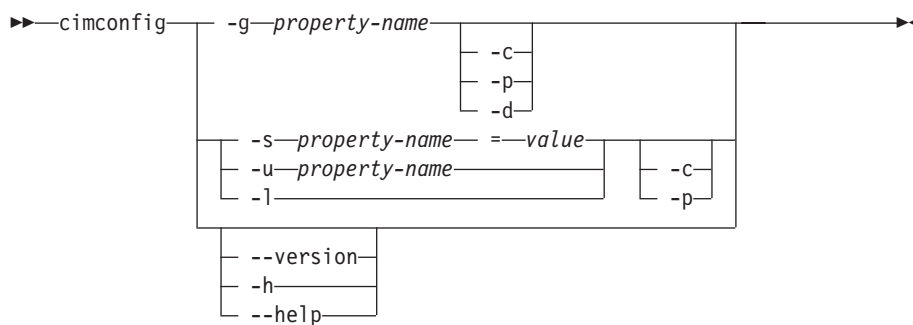
You can update the current configuration properties only while the CIM server is running. After a restart of the CIM server, these changes will be reset to the planned configuration values. For making permanent changes, you must change the planned configuration values.

Planned configuration properties:

Planned configuration properties can be modified even if the CIM server is stopped. If the planned configuration properties are changed when the CIM server is running, those changes do not take effect until the CIM server is restarted.

For using the cimconfig command, a user needs to have CONTROL access to profile CIMSERV in class WBEM.

Syntax



Options

The cimconfig command recognizes the following options:

- g *property-name*, -g *property-name* -c**
Gets the current value of the configuration property *property-name*. Returns an error when the CIM server is not running.
- g *property-name* -p**
Gets the planned value of the configuration property *property-name*.
- g *property-name* -d**
Gets the default value of the configuration property *property-name*. Returns an error when the CIM server is not running.
- s *property-name*=*value*, -s *property-name*=*value* -c**
Sets the current configuration property *property-name* to the value *value*. Returns an error when the CIM server is not running or the specified property cannot be updated dynamically.
- s *property-name*=*value* -p**
Sets the planned configuration property *property-name* to the value *value*.
- u *property-name*, -u *property-name* -c**
Unsets the value of the current configuration property *property-name* to its default value. Returns an error when the CIM server is not running or the specified property cannot be updated dynamically.
- u *property-name* -p**
Unsets the value of the planned configuration property *property-name* to its default value.
- l** Lists the names of all configuration properties. Returns an error when the CIM server is not running.
- l -c** Lists the name and value pairs of all current configuration properties. Returns an error when the CIM server is not running.
- l -p** Lists the name and value pairs of all planned configuration properties.
- version**
Displays the CIM server version.
- h, --help, no options specified**
Displays the command help information.

Examples

```
cimconfig -s traceLevel=4
```

```
cimconfig -s traceComponents=xml-io,http
```

Sets the trace level to trace all information with high data detail in the *xml-io* and *http* components.

```
cimconfig -s logLevel=ERROR -p
```

Sets the *logLevel* configuration property to the value `ERROR` in the *cimserver_planned.conf* file.

cimprovider

Description

The `cimprovider` command lets you disable, enable, remove, and list registered CIM providers or CIM provider modules and the according module status.

disable

When a CIM provider is disabled, the CIM server rejects any requests to the provider. When a provider module is disabled, any new requests to the providers that are contained in the specified provider module are rejected.

enable

When a CIM provider is enabled, the CIM server forwards requests to the provider. When a provider module is enabled, the providers that are contained in the provider module are ready to accept a new request.

remove

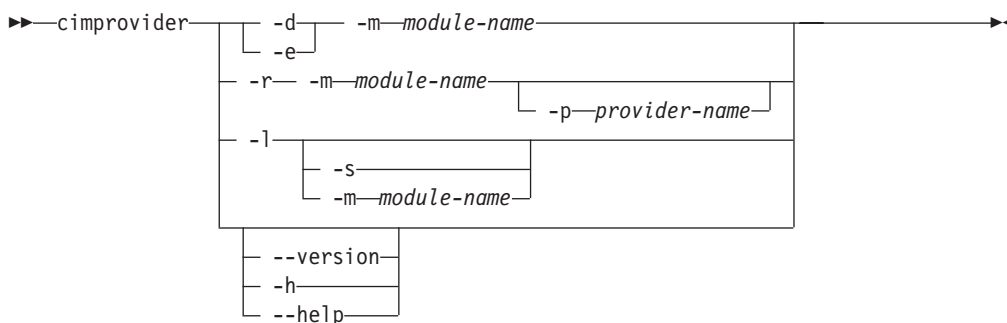
When a CIM provider is removed (unregistered), the CIM server will no longer have any information about the provider. When a CIM provider module is removed (unregistered), the CIM server will no longer have any information about any provider contained in the module. If you want to address requests to a provider after removal, the provider or provider module must be registered again (typically by loading its registration schema using the `cimmo fl` command).

list

You can list all registered provider modules and the according module status or all providers in the specified provider module.

For using the `cimprovider` command, the CIM server must be running, and the user needs to have `CONTROL` access to profile `CIMSERV` in class `WBEM`.

Syntax



Options

The cimprovider command recognizes the following options:

-d -m *module-name*

Disables the CIM provider module *module-name*. If the module is already disabled, an error message is returned.

-e -m *module-name*

Enables the CIM provider module *module-name*. If the module is already enabled or is currently being disabled, an error message is returned.

-r -m *module-name*

Removes the provider module *module-name* and all of its contained providers.

-r -m *module-name* -p *provider-name*

Removes the provider *provider-name* in the provider module *module-name* without affecting any other providers in that module.

-l

Displays all registered provider modules.

To list all providers in all modules, type a cimprovider -l command, followed by cimprovider -l -m for each listed module.

-l -s

Lists the status of all registered provider modules.

-l -m *module-name*

Lists all registered providers in module *module-name*.

--version

Displays the CIM server version.

-h, --help, no option specified

Displays the command help information.

Limitations

This command disables, enables, or removes one CIM provider module or CIM provider at a time.

Examples

cimprovider -d -m myProviderModule

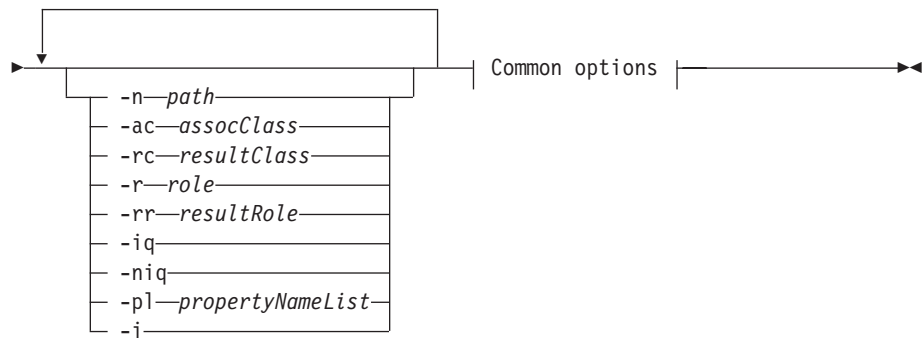
Disables provider module myProviderModule and all of its contained providers (placing them in a stopped state).

cimprovider -e -m myProviderModule

Enables provider module myProviderModule and all of its contained providers (placing them in an OK state).

cimprovider -r -m myProviderModule

Removes (unregisters) the myProviderModule provider module and all of its contained providers.

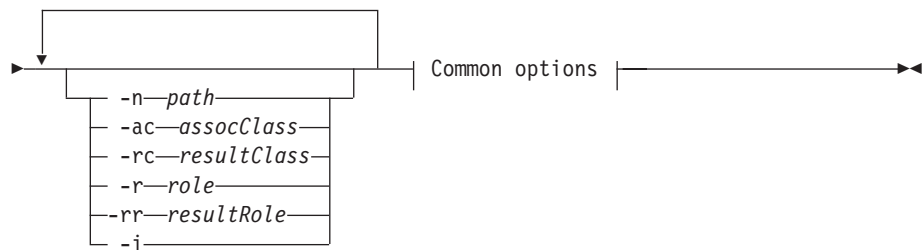
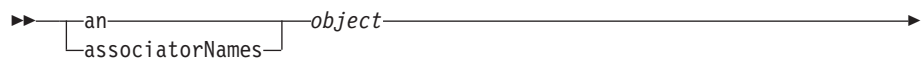


Enumerates the classes or instances linked (associated) to *object*, where *object* can be either the name of a CIM class or the name of a CIM instance. For a given class, the list of associated classes is returned. For a given instance name the list of associated instances is returned.

Example:

```
cimcli a 'IBMzOS_Test.handle="1"'
      -ac IBMzOS_BaseMetricValue
      -niq
or
cimcli a IBMzOS_Process -niq
```

an, associatorNames



Enumerates the class or instance names linked (associated) to *object*, where *object* can be either the name of a CIM class or the name of a CIM instance. For a given class, the list of associated class names is returned. For a given instance name the list of associated instance names is returned.

Example:

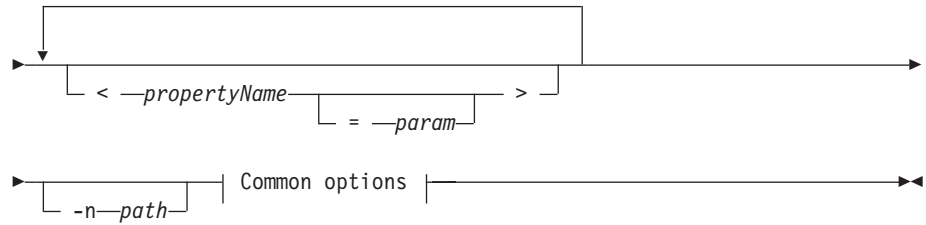
```
cimcli an 'IBMzOS_Test.handle="1"
        -ac IBMzOS_BaseMetricValue

or

cimcli an IBMzOS_Process
```

ci, createInstance





Creates one instance of the specified class with the parameters provided. It requires the *classname* parameter defining the class for which the instance is to be created and also allows for an optional set of parameters that define the properties to be provided in the format *<propertyName=value>*. If the *propertyName* parameter is provided with no value, the default is inserted. The command actually reads the CIM_Class, inserts the properties that have been defined on the command line using the value type from the class. The command will be rejected if the class does not exist in the namespace. It returns the object path of the created instance if the call to the CIM server was performed. Otherwise it returns the exception received.

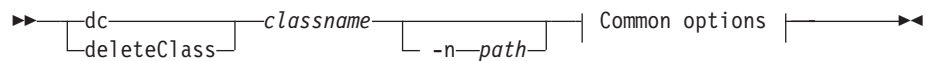
Example:

```

cimcli ci CIM_Person Name=Michael Title=Engineer
----> createInstance of class CIM_Person

```

dc, deleteClass



Deletes the CIM class specified by *classname*.

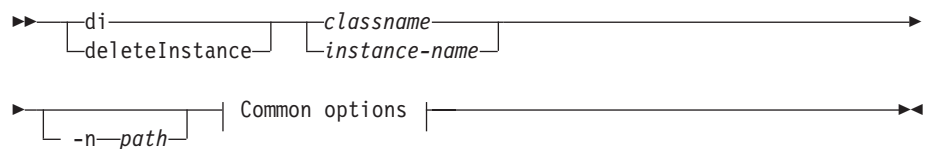
Example:

```

cimcli dc CIM_Person
----> deletes the Class when there are no instances
and sub-classes for this class

```

di, deleteInstance



Deletes the instance *instance-name* or interactively one instance from *classname*. If this parameter is a full instance name with classname and key bindings, the deleteInstance CIM operation is performed directly. If it is a class name with no keybindings, the enumerateInstances command is performed and the list of returned instances presented for the user to select one to delete. cimcli then performs deleteInstance with the selected instance name. There is no response if the instance was successfully deleted, or an exception returned if there were any errors.

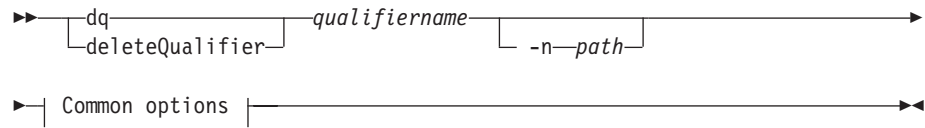
Example:

```

cimcli di CIM_Person
----> deleteInstance of class CIM_Person interactively

```

dq, deleteQualifier

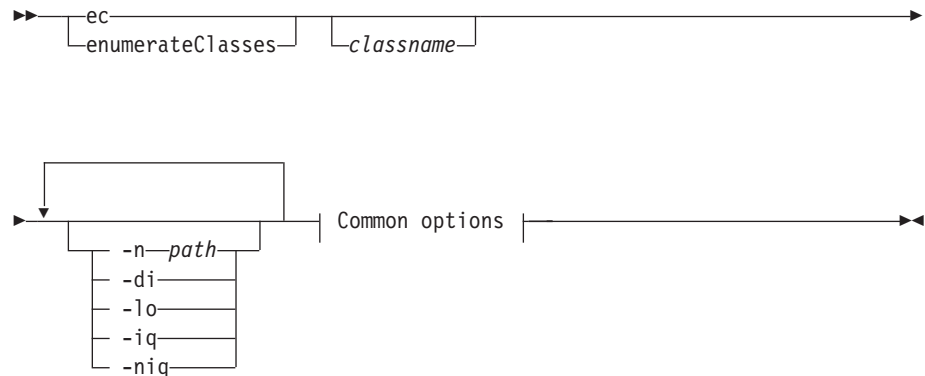


Deletes the CIM qualifier specified by *qualifiername*.

Example:

```
cimcli dq ASSOCIATION
----> deleteQualifier Association
      (generally not recommended)
```

ec, enumerateClasses

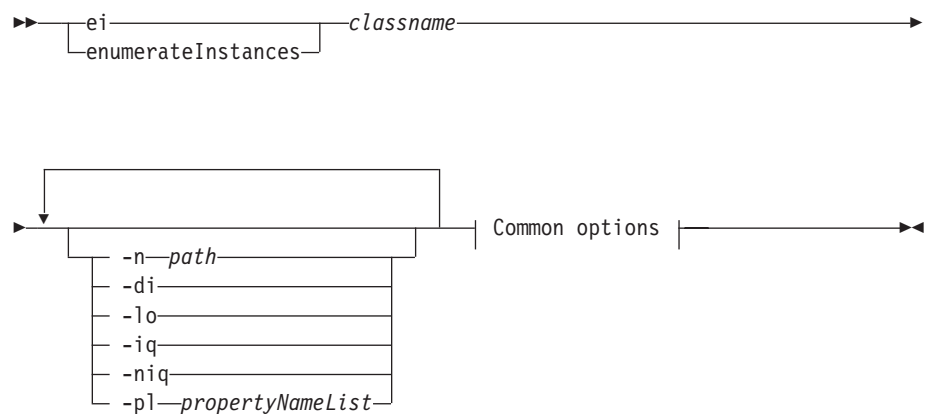


Enumerates the classes starting at the level defined by *classname*. If it is omitted, `cimcli` inserts an empty *classname*.

Example:

```
cimcli ec -n root/cimv2 -niq
----> enumerate classes from the root of the
      root/cimv2 namespace
```

ei, enumerateInstances

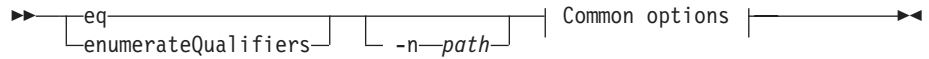


Enumerates instances of *classname*, where *classname* is the name of a CIM class.

Example:

```
cimcli ei CIM_ComputerSystem -niq -di
---> enumerate instances of class CIM_Computersystem,
      not including qualifiers (-niq) but also
      listing properties of inherited classes (-di)
```

eq, enumerateQualifiers

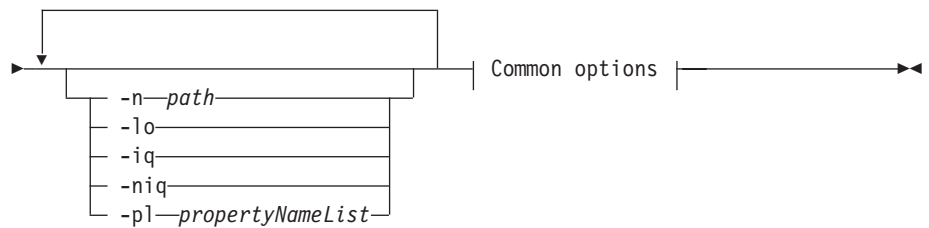


Enumerates all qualifiers in the specified or default namespace.

Example:

```
cimcli eq
---> enumerateQualifiers in the default
      root/cimv2 namespace
```

gc, getClass

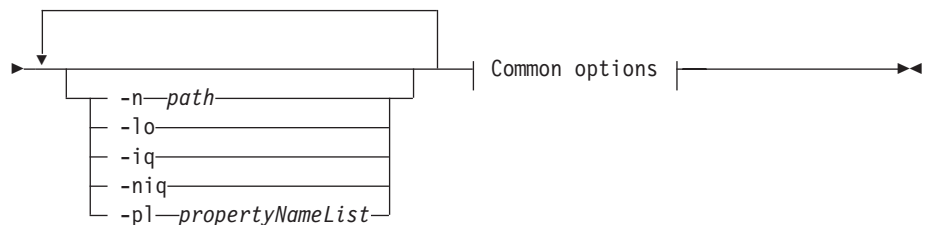
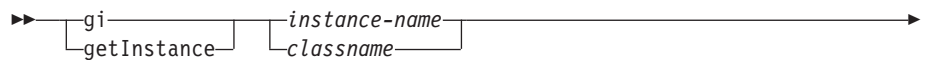


Gets the class of *classname*.

Example:

```
cimcli gc IBMzOS_Process
---> Get the definition for class IBMzOS_Process
```

gi, getInstance



getInstance *instance-name* gets the instance defined by the *instance-name* parameter and displays it in the format chosen for this operation (xml or mof).

If the parameter is an instance with keys defined (a CIMObjectPath), the getInstance CIM operation is performed and the return from the CIM server is presented.

If the input is a class name, an enumerateInstanceNames CIM operation is performed, and if any instance names are returned, the result is presented for the user to select one of the instances to be deleted.

If there are no instances, the command returns an empty response.

Example:

```
cimcli gi IBMzOS_UnixLocalFileSystem -niq
---> This is an interactive request that returns a list
      of instances from an enumerateInstance of
      IBMzOS_UnixLocalFileSystem from which the user can
      select one which cimcli will return as a complete
      instance.
```

gq, getQualifier

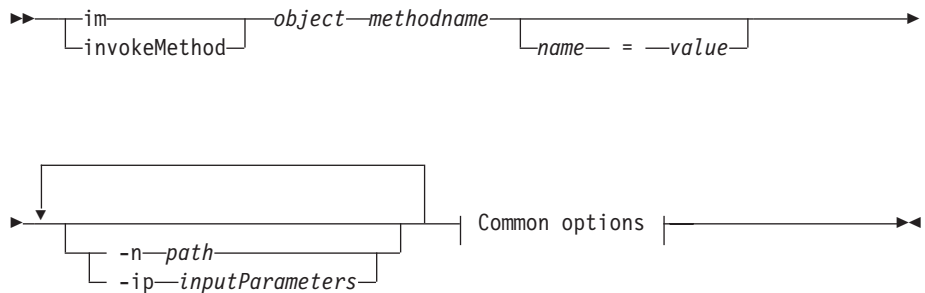


Gets the CIM qualifier specified by *qualifiername*.

Example:

```
cimcli gq Association
---> Get the qualifiers in mof output format
```

im, invokeMethod

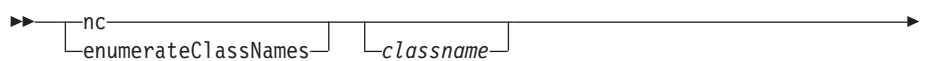


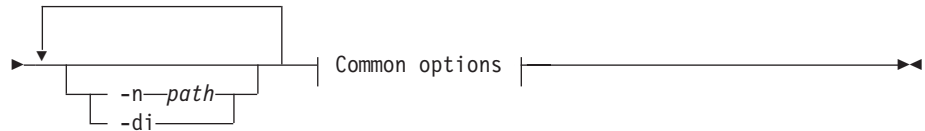
This command causes the CIM operation to perform an extrinsic method *methodname* on a CIM class or instance *object*. The parameters are supplied as *name=value* pairs. In the current version, all parameters are treated as strings.

Example:

```
cimcli im 'IBMzOS_Test.handle="1"' TriggerIndication -ip
      NumberOfIndications=3
```

nc, enumerateClassNames





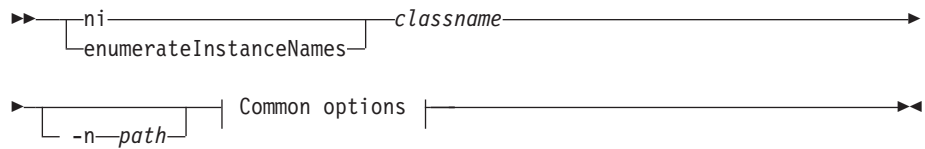
Enumerates sub class names of *classname* or all top level class names of a given namespace. Note that on z/OS all class names are returned in lowercase due to a z/OS specific performance optimization. Use the `getClass` operation to receive the exact case of the class name.

Example:

```

cimcli nc -di
    ---> enumerate all class names from the
          root/cimv2 namespace,
          including subclasses (-di)
  
```

ni, enumerateInstanceNames



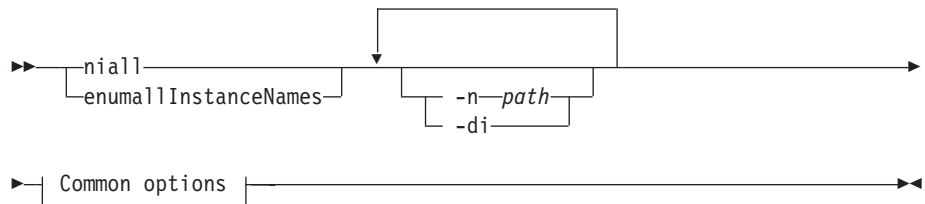
Enumerates instance names of *classname*, where *classname* must be the name of a CIM class.

Example:

```

cimcli ni CIM_Processor -n root/cimv2
  
```

niall, enumallInstanceNames



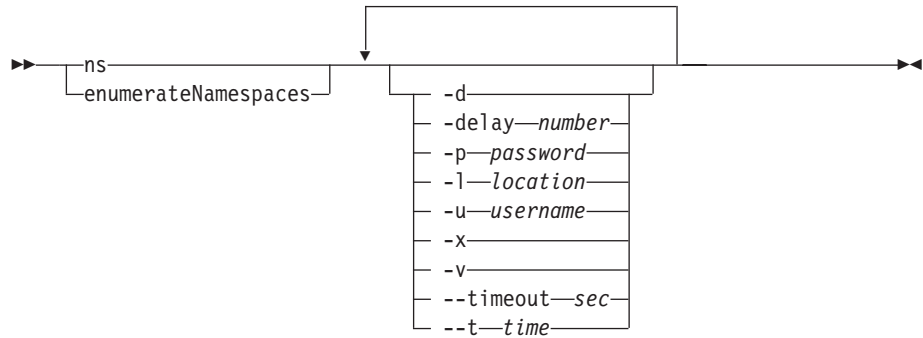
Performs an `enumerateInstanceNames` on all classes to get all instance names within the defined namespace. This function searches the complete namespace and returns the list of all instance names found.

Example:

```

cimcli niall -n root/cimv2
    ---> enumerateinstancenames of all classes under the
          namespace root/cimv2
  
```

ns, enumerateNamespaces

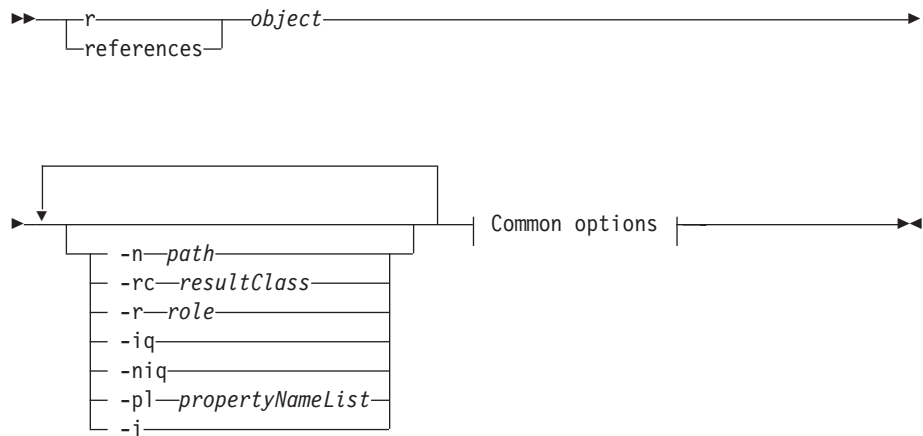


Requests an enumeration of all the namespaces in the target CIM server. This command uses both the CIM_Namespace class and if that fails, the __Namespace class to determine the list of namespaces.

Example:

```
cimcli ns ---> enumerate all Namespaces
```

r, references



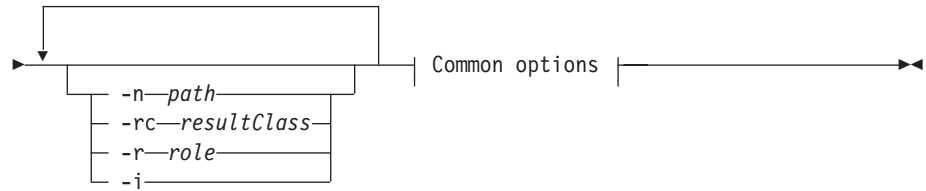
Enumerates the association classes or association instances linked to *object*, where *object* can be either the name of a CIM class or the name of a CIM instance. For a given class, the list of linked association classes is returned. For a given instance name the list of linked association class instances is returned.

Example:

```
cimcli r 'IBMzOS_ComputerSystem.  
          CreationClassName="IBMzOS_ComputerSystem",  
          Name="sys1"' -niq  
or  
cimcli r IBMzOS_OperatingSystem -rc CIM_OSProcess
```

rn, referenceNames



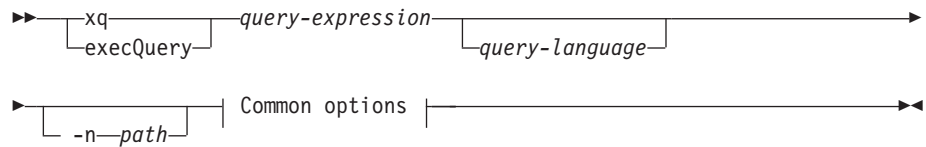


Enumerates the association class or instance names linked to *object*, where *object* can be either the name of a CIM class or the name of a CIM instance. For a given class, the list of linked association class names is returned. For a given instance name the list of linked association instance names is returned.

Example:

```
cimcli rn 'IBMzOS_ComputerSystem.  
          CreationClassName="IBMzOS_ComputerSystem",  
          Name="sys1"  
or  
cimcli rn IBMzOS_OperatingSystem -rc CIM_OSProcess
```

xq, execQuery



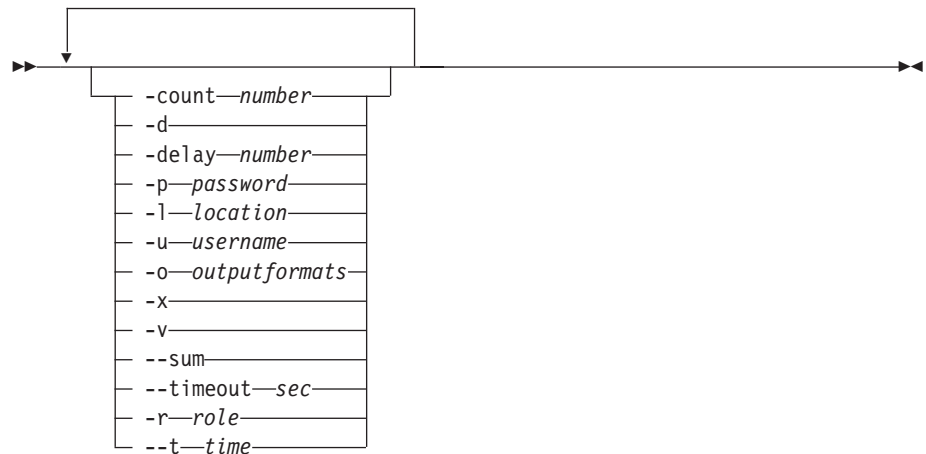
Performs the execQuery CIM operation with the specified *query-expression*. Note that the use of the execQuery operation has been deprecated by the DMTF and it may be removed in a future version of the "Specification for CIM Operations over HTTP".

Example:

```
cimcli xq "select handle,name from CIM_process  
          where handle = \"1\" WQL
```

Options

Common options:



Options are identified on the command line with the '-' or '--' notation. An option that is not used by a particular operation is ignored. The cimcli

command recognizes the following options:

- ac** *assocClass* Defines an *assocClass* string for Associator calls. Default is empty.
- ar** *assocRole* Defines a role string for an Associator's *AssocRole* parameter. Default is empty.
- count** *number* Expected number of objects returned, if the **-sum** option is set. Tests this number and displays the difference. Term nonzero is returned if test fails.
- d** Displays more detailed debug messages.
- delay** *number* Delay in seconds between connection and request. Default is 0.
- di** Specifies the *deepInheritance* parameter for selected commands. The default is 'false'. This option has different meanings for different commands and is used only with the enumerate commands. For further information, refer to the *CIM Operations over HTTP* published by the DMTF.
- h** Prints help usage message.
- hc** Prints CIM Operation command list.
- help** Prints full help message with commands, options, and examples.
- ho** Prints list of options.
- i** Interactively asks the user to select instances. Used with associator and reference operations.
- ip** *inputParameters* Defines an *invokeMethod* input parameter list. Format is *p1=v1,p2=v2,...,pn=vn* (without spaces). Default is empty.
- l** *location* Allows input of the host name for the CIM server and optionally the port (*HostName:port*). The default is *localhost:5988*. The port component is optional. The default is 5988.
- n** *path* Specifies the namespace for the operation. The default is *root/cimv2*.
- nlo, !lo** When set, sets *localOnly* = 'false' on operations. Default is 'false'.

Note that option *localOnly* has been deprecated by the DMTF for some operations and will completely be removed with the next major version of CIM.
- niq, !iq** Sets *includeQualifiers* = 'false' on operations. Default is 'false'.

Note that option *includeQualifiers* has been deprecated by the DMTF for some operations and will completely be removed with the next major version of CIM.
- p** *password* Allows the input of a password for the command's server authentication. The default is empty.
- u** *username* Allows the input of a user name for authentication. The default is empty.
- o** *outputformats* Output in xml, mof, txt. Default is mof.
- pl** *propertyNameList* Defines a *propertyNameList*. Format is *p1,p2,p3* (without spaces). Use "" for an empty list.
- r** *role* Defines a role string for a reference role parameter. Default is empty.

--r <i>repeat</i>	Sets the number of times to repeat the function. Zero means one time. Repeats the operation without disconnecting. Default is 0.
-rc <i>resultClass</i>	Defines a resultClass string for References and Associators. Default is empty.
-rr <i>resultRole</i>	Defines a role string for an Associator's operation resultRole parameter. Default is empty.
--sum	Presents only summary information, not the complete output. Generally this option presents counts of objects returned instead of the names or objects themselves.
--t <i>time</i>	Measures the time for operation and presentation of the results upon command completion.
--timeout <i>sec</i>	Sets the connection timeout in seconds. Default is 0.
-trace <i>traceLevel</i>	Sets the common components trace. Sets the trace level. 0 is off. Default is 0. Valid values are 0 to 4.
-v	Verbose display. Includes detailed parameter input display.
--version	Displays the software version.
-x	Output objects in xml instead of mof format.

cimsub

Description

The cimsub command lets you manage CIM indications on the local CIM server. The command can list, enable, disable and remove indication subscriptions, filters and handlers. However, you cannot modify or create a handler or a filter. The CIM indication must be created or modified by a CIM client program.

list Lists all or selected indication subscriptions, filters, and handlers, and displays the requested information about the instance(s).

enable

Enables a specific subscription. Sets a subscription into the enabled state, and the CIM server starts to process it.

disable

Disables a specific subscription. Sets a subscription into the disabled state, and it is no longer processed by the CIM server.

remove

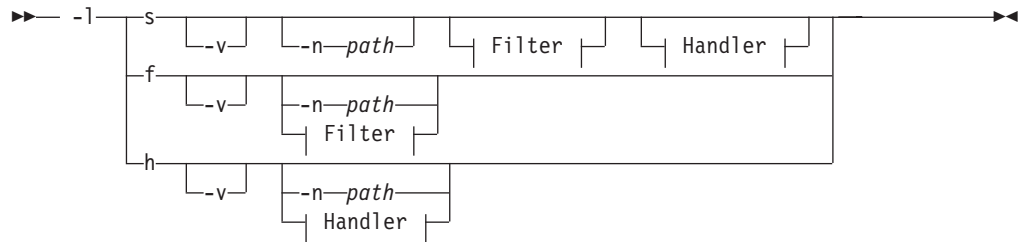
Removes a specific indication subscription, filter, and/or handler from the CIM server. The information is removed within the CIM server and can only be recreated by a client application. The administrator must take care that a filter or handler is not referenced by any other subscription. If this is the case, but the filter or handler is deleted anyway, this subscription will no longer work.

In order to use the cimsub command, the CIM server must be running on the local system and a user needs to have CONTROL access to profile CIMSERV in class WBEM.

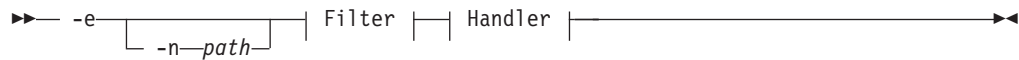
Syntax



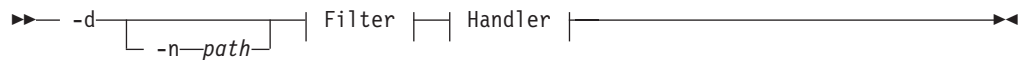
list:



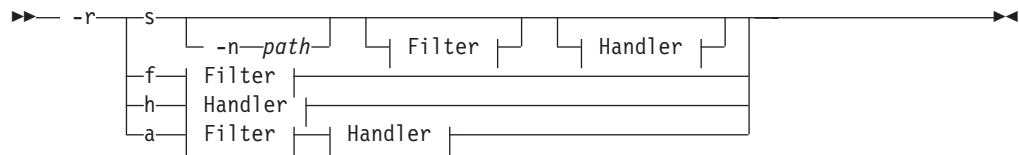
enable:



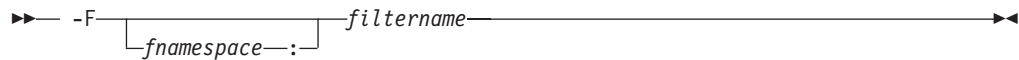
disable:



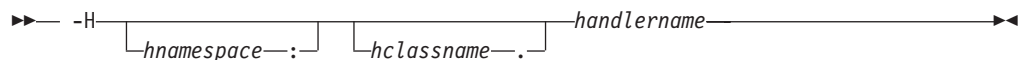
remove:



Filter:



Handler:



Options

The cimsub command recognizes the following options:

- l Lists all or selected indication subscriptions (-ls)
filters (-lf)
handlers (-lh)

Options -F and -H are superseding the -n namespace option, if -n is set together with either -F or -H.

- e Sets the subscription state to enabled.
- d Sets the subscription state to disabled.
- r Removes a specific
 - indication subscription (-rs)
 - filter (-rf)
 - handler (-rh)
 - or all three together (-ra)

Options -F and -H are superseding the -n namespace option, if -n is set together with either -F or -H.

- v Displays verbose information (for example, subscription state, filter query, handler destination) for each listed instance.
- F [*fnamespace:*]*filtername*
Specifies the name of the filter instance used for the subscription operation. If the filter namespace [*fnamespace:*] is not specified, the operation is using the namespace of the subscription.
- H [*hnamespace:*][*hclassname.*]*handlername*
Specifies the name of the handler instance used for the subscription operation. If the handler namespace [*hnamespace:*] is not specified, the operation is using the namespace of the subscription. If the handler class name [*hclassname.*] is not specified, the operation is using the *CIM_ListenerDestinationCIMXML* handler class name.

Note: Currently the only supported handler is an instance of the *CIM_ListenerDestinationCIMXML* class or subclass.

- n *path*
Specifies the namespace for the operation. For the -l option, if no namespace is specified, instances in all namespaces are listed. For all other operations, if no namespace is specified, the *cimsub* command operates on instances of the *root/PG_InterOp* namespace.

Note: It is recommended not to use any other namespace for indications than *root/PG_InterOp*.

- help Displays the command help information.
- version
Displays the CIM server version.

Examples

The following example lists all subscriptions in the namespace *root/PG_InterOp* in verbose mode:

cimsub -ls -v

Output:

```
Namespace:      root/PG_InterOp
Filter:         root/PG_InterOp:IndicationTest_indicationFilter
Handler:        root/PG_InterOp:CIM_ListenerDestinationCIMXML.IndicationTest
Query:         "SELECT * FROM TestIndication"
Destination:    http://test.server.com/
SubscriptionState: Enabled
```

```
cimsub -d -F IndicationTest_indicationFilter -H IndicationTest
```

Disables the subscription specified by -F and -H, and displays the result in verbose mode.

```
cimsub -ls -v
```

Output:

```
Namespace:      root/PG_InterOp
Filter:         root/PG_InterOp:IndicationTest_indicationFilter
Handler:       root/PG_InterOp:CIM_ListenerDestinationCIMXML.IndicationTest
Query:         "SELECT * FROM TestIndication"
Destination:   http://test.server.com/
SubscriptionState: Disabled
```

MODIFY console command

In addition to the cimconfig command-line utility (see “cimconfig” on page 55), starting with z/OS 1.10 the CIM server configuration can be changed from the z/OS system console using the MODIFY command. The general syntax for using the MODIFY command to pass information to a Unix System Services Application is described in *z/OS MVS System Commands*.

Syntax

Following is the specific syntax for using the MODIFY command to pass configuration changes to the CIM server. Between the options, no spaces are allowed:

```

>> MODIFY jobname, -APPL = -CONFIG,
    F
>property = 'value',
            value, -PLANNED

```

Options

Basically the CIM server accepts the same options for the MODIFY command as for the cimconfig utility.

jobname

The name of the job that runs the CIM server. When the CIM server is run as a started task, this will usually be CFZCIM.

APPL=CONFIG

This is the indicator for the CIM server that a configuration change was requested through the z/OS system console.

property

The name of the configuration property to be changed. For a complete list of CIM server configuration properties see “CIM server configuration properties” on page 42. Typically, the only current configuration properties that you can change dynamically are the *shutdownTimeout* property and the logging and tracing properties. Permanent changes require a CIM server restart. They are indicated using the PLANNED keyword at the end of the MODIFY command string.

'value', value

The new value for the configuration property to be changed. For case sensitive property values such as path names the value needs to be enclosed in quotes, otherwise it will be converted to uppercase. To reset a property to its default value, omit the *value* parameter.

PLANNED

Indicates that the configuration change should be made permanently. This means that the change will only become effective after a CIM server restart, and that the change will also persist further restarts until it is changed again. If PLANNED was not specified at the end of the command, the changes will only stay in effect until the next restart of the CIM server.

Examples

F CFZCIM,APPL=CONFIG,traceComponents=xml io

F CFZCIM,APPL=CONFIG,traceLevel=4

Turns on tracing of the CIM server's XML traffic.

F CFZCIM,APPL=CONFIG,enableRemotePrivilegedUserAccess=true,PLANNED

Permanently enables superusers (UID=0) to issue requests against the CIM server from a remote system.

Backing up the CIM server repository

The CIM server keeps definitions of the data about managed objects and their providers in its repository. Four namespaces install with the CIM server. Others may be added by CIM clients and providers. The four that are automatically installed are:

root The root namespace exists to conform to the DMTF specifications.

root/cimv2

The standard CIM Schemas as well as the schemas for the shipped providers are located here.

root/PG_Interop

Among other purposes, this namespace is used for provider registration. All providers must register here (see “cimprovider” on page 57).

root/PG_Internal

This space is reserved for use by the OpenPegasus CIM server only.

It is important to schedule backups of the repository directories and files. If the repository is deleted or corrupted, backups of the repository files need to be restored. If the repository files cannot be restored from a backup, refer to section “Migration from z/OS 1.8 or z/OS 1.9 to z/OS 1.10” on page 18 for information on how to recover the repository.

As recommended in the *z/OS Program Directory*, the path */var/wbem* should be mounted as a separate data set to simplify backing up. It is also recommended to stop the CIM server during backup to avoid data corruption.

Note: If the repository was backed up from a prior z/OS release, it should not be restored onto a system that runs a later version of the OS. Once a new version of z/OS was installed and the CIM server has been initially started, you should immediately back up the upgraded repository and discard old repository backups.

Sample client applications

An OCO sample client application **cimivp** is delivered as executable with the product in file `/usr/lpp/wbem/bin/cimivp`. You can use it as an installation verification program. It displays information about the z/OS system and lists currently active UNIX System Services processes.

You invoke this program as job CFZIVP contained in SYS1.SAMPLIB or from the UNIX System Services command line as `cimivp`.

On successful completion, it generates an output similar to the one shown hereafter. If the program produces unreadable output, check whether you forgot to tag the output files as EBCDIC. See also “ASCII-EBCDIC conversion” on page 75.

```
cimivp Main started ...
cimivp::_connectToServer(127.0.0.1:5988) called
cimivp::_connectToServer - success
> Found Computer System : BOEMT11.boeb.de.ibm.com
                        (CPUID: 026F7A2084, VMGuestID: MT11)
> Found Operating System : MT11 (Version: 01.10.00, Sysplex: MT1PLEX,
> Found the following USS processes:
--> 1 - BPXPINPR
--> 5 - EZBTCPIP
--> 7 - EZBTSSL
--> 16777227 - inetd
--> 67108877 - EUVFSKDC
--> 67108883 - otelnetd
--> 50331670 - cimserver
--> 83886103 - PAGENT
--> 27 - cimivp
cimivp - All tests completed successfully.
```

The source code for a sample client monitoring application is printed in the z/OS *RMF Programmer's Guide*.

Troubleshooting

This chapter contains the following subsections:

- “Problem determination”
- “Error messages” on page 75

You can find further helpful information in:

- “Logging” on page 157
- “CIM server messages” on page 157

Problem determination

If you have problems with using the CIM server you can trace the CIM server activities. Therefore, set the following options using the `cimconfig` command (see “CIM server configuration properties” on page 42):

- `traceLevel`
- `traceComponents`

These can be set dynamically:

```
cimconfig -s traceLevel=2
```

or permanently:

```
cimconfig -p -s traceLevel=2
```

or temporarily on the UNIX System Services command line of the CIM server:

```
cimserver traceLevel=2 traceComponents=all
```

Note: Tracing can have a negative impact on performance.

ASCII-EBCDIC conversion

Since the z/OS CIM server and all of its command-line utilities operate in the enhanced ASCII environment, all output is written using ASCII encoding. This can lead to garbage being displayed when watching the output from the CIM server command-line utilities, sample programs or from the CIM server itself. By default, the configuration files *cimserver.env* and *profile.add* shipped with the CIM server provide the required settings for automatic conversion to the correct encoding. For details on how to enable the automatic conversion and about Enhanced ASCII in general, Refer to *Using Enhanced ASCII functionality* in *z/OS UNIX System Services Planning*.

One important issue is that automatic conversion so far only occurs for **UNIX System Service** applications. When the output of the CIM server or any of its clients should be consumed or displayed by applications other than **UNIX System Services** applications, the conversion must take place when the data is created. To achieve this, the output files need to be tagged as EBCDIC so that, for example, the CIM server's output is converted to EBCDIC before it is consumed by these applications. There is a command for tagging files as EBCDIC:

```
chtag -t -c IBM-1047 /var/wbem/cimserver.trc
```

Error messages

The following is a list of typical errors that can be observed when working with CIM:

Error: BPXP014I ENVIRONMENT MUST REMAIN CONTROLLED FOR DAEMON (BPX.DAEMON) PROCESSING.

BPXP015I HFS PROGRAM /usr/lpp/wbem/provider/<provider_library> IS NOT MARKED PROGRAM CONTROLLED.

The provider <provider_library> is not marked program controlled.

When or where seen: Messages on the console.

Solution: Mark the DLL /usr/lpp/wbem/provider/<provider_library> as program controlled by using the command `extattr +p <fully qualified dll name>`. Restart the CIM server and try again.

Error: CIM_ERR_ACCESS_DENIED

Access to a CIM resource was not available to the client: "Not authorized to run <name of a CIM Operation> in the namespace root/PG_Internal"

When or where seen: Client application / Details in the CIM server trace log

Solution: Permit the user ID to execute a configuration command with CONTROL access to Security profile CIMSERV in CLASS WBEM.

Error: CIM runtime environment user ID requires either READ access to BPX.SERVER or it must be user ID 0.

When or where seen: The CIM server error log after CIM server fails to start

Solution: Either permit the user ID READ access to BPX.SERVER if BPX.SERVER is set up, or run the command under a privileged user ID (UID 0).

Error: CIM runtime environment user ID requires CONTROL access to profile CIMSERV in class WBEM.

When or where seen: The CIM server error log after CIM server fails to start

Solution: The CIM server startup fails because the CIM server user ID fails to have CONTROL access to profile CIMSERV in class WBEM. Grant the CIM server user ID CONTROL access to profile CIMSERV in class WBEM.

Error: failed to bind to HTTP port

When or where seen: CIM server startup console message

Solution: The CIM server cannot start because it fails to listen on one of the ports 5988 (for http) or 5989 (for https). Either the CIM server is already running, another Server is listening on one of these ports, or the ports have been blocked in the TCPIP configuration.

Error: HTTP Error (401 Unauthorized)

When or where seen: Client application

Solution: The user authentication failed. The client application either did not provide user ID and password on a request at all, or the supplied user ID and password are not valid for the z/OS system on which the CIM server is running.

Permit the user ID to execute a client request with at least READ access to Security profile CIMSERV in CLASS WBEM. Check the server log for a detailed error report.

Error: ICH14080I

Warning: RACF detected a possible error in the dynamic class descriptor table, entry WBEM, error code 01. The class is available for further processing. The class name does not contain a national character nor a number. To assure IBM does not create an IBM-defined class in the future by this same name, you should choose a class name which contains at least one national character or a number.

When or where seen: RACF setup of dynamic class WBEM

Solution: Ignore the warning.

Error: CFZ17201W: ACCESS IS NOT ENABLED FOR REMOTE USERS WITH SUPERUSER AUTHORITY.

When or where seen: On the client side.

Solution: The remote client uses a local user with UID=0. However, the CIM server is configured to reject remote access if the local user is a super-user (parameter enableRemotePrivilegedUserAccess=false). If you want to enable the local user with remote privileged access, then switch the parameter to true. Otherwise, change the local user to a non-super-user by setting the UID ≠ 0.

Client Side Error: CIM_ERR_ACCESS_DENIED

Access to a CIM resource was not available to the client: "EDC5139I
Operation not permitted."

When or where seen: Client application / Details in the CIM server trace log

Solution: Permit the CIM server runtime environment user ID as surrogate for the requesting client user ID to use the command: PERMIT
BPX.SRV.<client uid> CL(SURROGAT) ID(<CIMServer UID>) ACCESS(READ)

Chapter 5. z/OS Management Instrumentation for CIM

The CIM standard provides the ability to develop management applications that work with systems management data. To work with CIM, developers should have a thorough understanding of the CIM standard defined by the DMTF. For more information about the CIM standard, see *Common Information Model (CIM) Standards* on the DMTF web site.

IBM has developed providers for z/OS for z/OS that support basic operating system information and some performance metrics. A CIM provider is the link between the CIM server and the system (see Figure 2 on page 3). This interface allows CIM to access and manage the resources. Each CIM provider makes accessible the resources it represents in a standard way.

Notes:

1. IBM only supports the classes and properties listed in the present document or in other z/OS documentation provided by IBM. All other use classes or properties which are not documented by IBM, IBM does not support, and bears no responsibility for their use.
2. Not all properties of the supported CIM classes described in this document are implemented by z/OS. Those properties implemented by z/OS are documented in each of the following subchapters. For all CIM properties not implemented by z/OS, the CIM server returns no values.

The following CIM classes and associations are implemented as IBM-supplied providers to provide basic operating system information:

Base classes

(See page 83)

- IBMzOS_ComputerSystem: subclass of CIM_ComputerSystem
- IBMzOS_OperatingSystem: subclass of CIM_OperatingSystem
- IBMzOS_OSProcess: subclass of association CIM_OSProcess
- IBMzOS_Process: subclass of CIM_Process
- IBMzOS_RunningOS: subclass of association CIM_RunningOS
- IBMzOS_UnixProcess: subclass of CIM_UnixProcess
- IBMzOS_LogicalDisk: subclass of CIM_LogicalDisk
- IBMzOS_LogicalDiskDevice: subclass of association CIM_SystemDevice

BaseBoard classes

(See page 89)

- IBM_BaseBoard: subclass of CIM_Card
- IBMzOS_BaseBoard: subclass of IBM_BaseBoard

Processor classes

(See page 91)

- IBMzOS_CSProcessor: subclass of association CIM_SystemDevice
- IBMzOS_Processor: subclass of CIM_Processor

File System classes

(See page 96)

- IBMzOS_HostedFileSystem: subclass of association CIM_HostedFileSystem

- IBMzOS_NFS: subclass of CIM_NFS
- IBMzOS_UnixLocalFileSystem: subclass of CIM_UnixLocalFileSystem

Network classes

(See page 98)

- IBMzOS_EthernetPort: subclass of CIM_EthernetPort
- IBMzOS_CSNetworkPort: subclass of association CIM_SystemDevice
- IBMzOS_IPProtocolEndpoint: subclass of CIM_IPProtocolEndpoint
- IBMzOS_NetworkPortImplementsIPEndpoint: subclass of association CIM_PortImplementsEndpoint

Job classes

(See page 101)

- IBMzOS_Job: subclass of CIM_Job
- IBMzOS_JES2Job: subclass of IBMzOS_Job
- IBMzOS_JES3Job: subclass of IBMzOS_Job
- IBMzOS_SysoutDataset: subclass of CIM_LogicalFile
- IBMzOS_JES2SysoutDataset: subclass of IBMzOS_SysoutDataset
- IBMzOS_JES3SysoutDataset: subclass of IBMzOS_SysoutDataset
- IBMzOS_Subsystem: subclass of CIM_Service
- IBMzOS_JobsManagementSettings: subclass of CIM_SettingData
- association IBMzOS_SubsystemJES2Jobs (between IBMzOS_Subsystem and IBMzOS_JES2Job)
- association IBMzOS_SubsystemJES3Jobs (between IBMzOS_Subsystem and IBMzOS_JES3Job)
- association IBMzOS_UsesJES3SysoutDatasets (between IBMzOS_JES3Job and IBMzOS_JES3SysoutDataset)
- association IBMzOS_UsesJES2SysoutDatasets (between IBMzOS_JES2Job and IBMzOS_JES2SysoutDataset)

Cluster classes

(See page 125)

- IBMzOS_Sysplex: subclass of IBMzOS_Cluster
- IBMzOS_SysplexNode: subclass of IBMzOS_ClusterNode
- IBMzOS_CouplingFacility: subclass of IBMzOS_ClusterAggregatedResource
- IBMzOS_CFStructure: subclass of IBMzOS_ClusterAggregatedResource
- IBMzOS_CFStructureConnector: subclass of IBMzOS_ClusterResource
- IBMzOS_ClusterResource
- IBMzOS_ClusterGlobalResource
- IBMzOS_ClusterAggregatedResource
- IBMzOS_Cluster
- IBMzOS_ClusterNode
- association IBMzOS_CollectionOfSysplexNodes
- association IBMzOS_CollectionOfCFs
- association IBMzOS_HostedCFStructures
- association IBMzOS_HostedCFStructureConnectors
- association IBMzOS_CFStructureDependsOn
- association IBMzOS_UsesCFs

- association IBMzOS_UsesCouplingFunctions
- association IBMzOS_UsesSysplexCoupleDatasets
- association IBMzOS_UsesCFRMCoupleDatasets
- association IBMzOS_UsesCFRMPolicies

Cluster indications

(See also “Support of CIM indications” on page 4)

- IBMzOS_SysplexInstCreation
- IBMzOS_SysplexInstModification
- IBMzOS_Sysplex_ReallocateInitiated
- IBMzOS_Sysplex_ReallocateCompleted
- IBMzOS_Sysplex_CFRM_CDS_Initialized
- IBMzOS_SysplexNodeInstCreation
- IBMzOS_SysplexNodeInstDeletion
- IBMzOS_SysplexNodeInstModification
- IBMzOS_CouplingFacilityInstCreation
- IBMzOS_CouplingFacilityInstDeletion
- IBMzOS_CouplingFacilityInstModification
- IBMzOS_CFStructureInstCreation
- IBMzOS_CFStructureInstDeletion
- IBMzOS_CFStructureInstModification
- IBMzOS_CFStrConnectorInstCreation
- IBMzOS_CFStrConnectorInstDeletion
- IBMzOS_CFStrConnectorInstModification
- IBMzOS_CollectionOfSysplexNodesInstCreation
- IBMzOS_CollectionOfSysplexNodesInstDeletion
- IBMzOS_CollectionOfCFsInstCreation
- IBMzOS_CollectionOfCFsInstDeletion
- IBMzOS_HostedCFStructureInstCreation
- IBMzOS_HostedCFStructureInstDeletion
- IBMzOS_HostedCFStrConnectorInstCreation
- IBMzOS_HostedCFStrConnectorInstDeletion

WLM classes

(See *z/OS RMF Programmer's Guide*)

- IBMzOS_WLMServiceDefinition: subclass of CIM_ManagedElement (to exploit this functionality, RMF must be installed and running. See *z/OS RMF Programmer's Guide*)
- IBMzOS_WLMServiceClassPeriod (to exploit this functionality, RMF must be installed and running. See *z/OS RMF Programmer's Guide*)
- IBMzOS_WLM (see page 199)
- association IBMzOS_WLMOS (between IBMzOS_WLM and IBMzOS_ComputerSystem) (see page 199)

WLM indications

- IBMzOS_WLMPolicyActivationIndication (see page 201)

CIM classes implemented by RMF

(See *z/OS RMF Programmer's Guide*)

- IBMzOS_BaseMetricValue
- IBMzOS_BaseMetricDefinition
- IBMzOS_MetricForME
- IBMzOS_MetricDefForME
- IBMzOS_MetricInstance
- IBMzOS_Channel
- IBMz_CEC
- IBMz_ComputerSystem

To exploit this functionality, RMF must be installed and running.

Notes:

1. The z/OS Communications Server provides documentation of these CIM classes. For details refer to *Considerations for Common Information Model (CIM) providers* in *z/OS Communications Server: IP Configuration Guide*.
2. For all classes, the properties that are common for @server and the z/OS specific properties are documented in separate tables.
3. Starting with z/OS 1.9, the CIM server exploits the functionality of Common event adapter (CEA). CEA is a z/OS component that provides the ability to deliver z/OS events to C-language clients. A CEA address space is started automatically during initialization of every z/OS system. In order for the address space to start successfully, you must configure CEA to work with z/OS. Failure to do so will cause CEA to run in a minimum function mode. For details refer to *z/OS Planning for Installation*.
4. An extra security setup is needed for the Job and Cluster classes.

To understand the syntax of the graphics showing class structures, see “Legend for graphics showing class structures” on page x.

Supported CIM operations

While the z/OS CIM server supports all of the CIM operations from the DMTF's *CIM Operations over HTTP* specification, only a specific subset of operations is supported by the OS management CIM providers delivered with this release of z/OS.

The following operations are available for all OS management classes or for association classes.

Available for all OS management classes:

- EnumerateInstanceNames
- EnumerateInstances
- GetInstance

Additionally available for all association classes:

- Associators
- AssociatorNames
- References
- ReferenceNames

OS management Base classes

Figure 8 illustrates the relationship between the IBM extension classes, and the CIM Base classes that they extend. The packages, in which the classes are defined in the CIM Schema, are indicated in parenthesis.

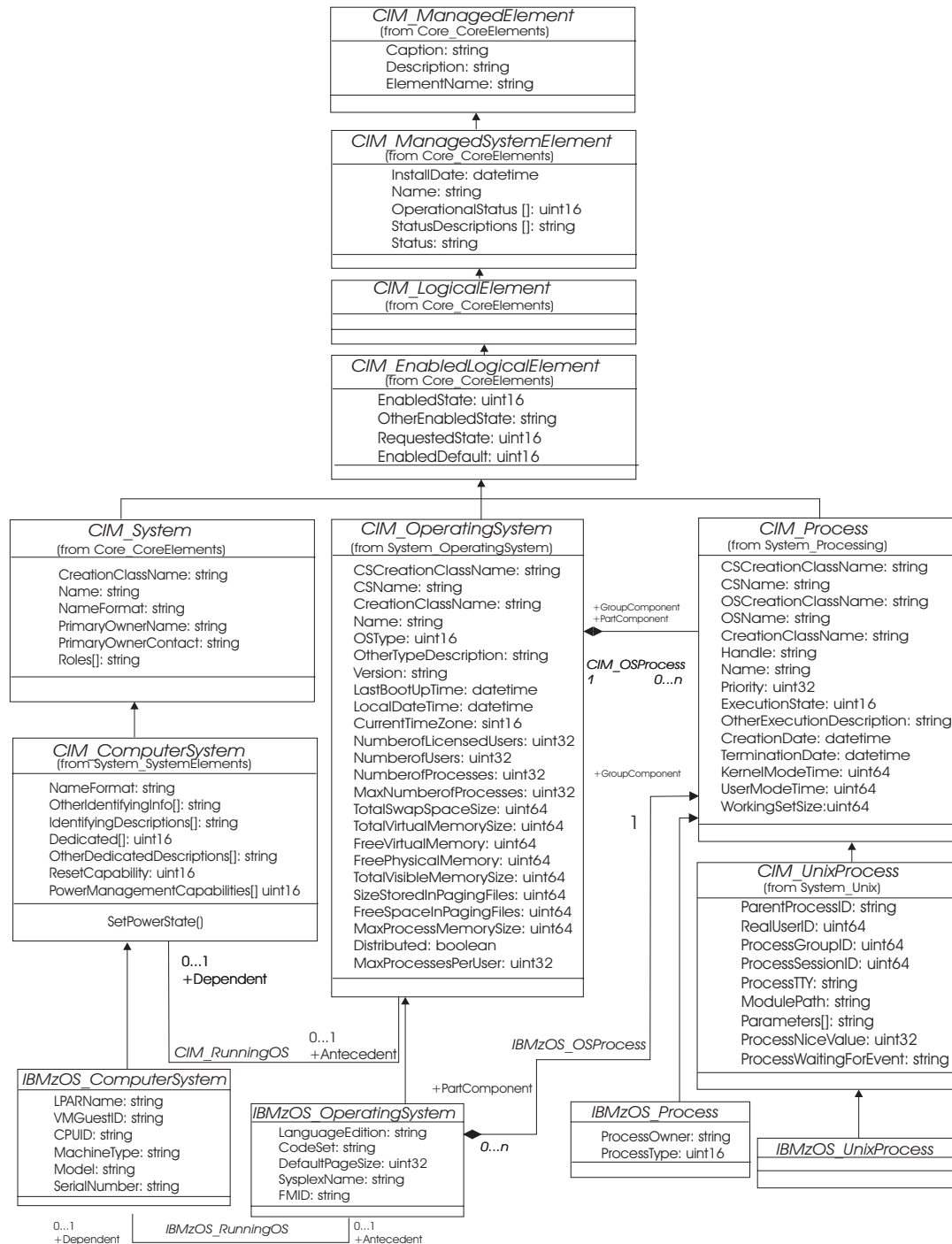


Figure 8. CIM Base classes extended by z/OS-specific classes (1)

The MOF files that define these classes can be found in directory *schemas/os_management* relative to where the providers for z/OS have been installed. The default is */usr/lpp/wbem/provider*.

CIM_ComputerSystem

This class represents either virtual or physical computer systems in the sense of a container inside which an operating system may run. This is the central class of the OS Management data model and aggregates all other resource classes.

The z/OS specific subclass is **IBMzOS_ComputerSystem**.

IBMzOS_ComputerSystem

This class provides basic computer system information such as computer name, and status information. A provider instruments this class so that it can be used by client applications to identify the managed system on which the provider is running (typically a server or an application).

The following properties are common for @server:

Property	Description
string Caption	Always set to 'IBM z/OS Computer System'.
string Description	Always set to 'This is an IBMzOS_ComputerSystem'.
string ElementName	Same as Name
string Name [key]	The fully qualified IP host name.
string CreationClassName [key]	Always set to 'IBMzOS_ComputerSystem'
string NameFormat	Describes the format used to build the Name property. Always set to 'IP'.
uint16 Dedicated[]	Indicates whether this is a special purpose system. Always set to 0 (not dedicated).
string UUID	The universally unique identifier of the server. For z/OS, no value is supplied for this property, but it is maintained for compatibility with the other IBM @server platforms.

The following properties have data that may be specific to z/OS, or may map to z/OS specific attributes.

Property	Description
string LPARName	Name of the zSeries® logical partition that makes up the computer system. If not running in LPAR mode, a blank string is returned here.
string VMGuestID	z/VM user ID of the virtual machine, of which the current z/OS image is a guest. If z/OS is not running as a guest under z/VM, a blank string is returned here.
string CPUID	Unique ID of the physical server on which this computer system is running.
string SerialNumber	IBM allocated number used to identify the server on which this computer system is running.
string MachineType	Processor family of this z/OS server.
string Model	Model number of the server.
string Manufacturer	The name of the company that produced the server.

Property	Description
uint16 LPARid	Logical partition number. This number distinguishes the configuration from all other level-2 configurations provided by the same LPAR hypervisor.
string Plant	Plant of manufacturer for the CPU.

CIM_OperatingSystem

This class represents a running operating system with its basic properties.

The z/OS specific subclass is **IBMzOS_OperatingSystem**.

IBMzOS_OperatingSystem

This class is for use by client applications to obtain basic properties of a running z/OS operating system.

The following properties are common for @server:

Property	Description
string Name [key]	The name of the z/OS operating system.
uint16 OperationalStatus[]	Overall system status.
uint16 OSType	Always 68 ('z/OS').
string Version	Version, release and modification of the operating system in the format of "VV.RR.MM". For example, for z/OS V1.7.0, this will return "01.07.00".
datetime LastBootUpTime	Time when the operating system was IPLed.
datetime LocalDateTime	Local time of the operating system
sint16 CurrentTimeZone	Time zone for the operating system, offset in minutes from GMT.
uint32 NumberOfUsers	The number of currently logged on TSO and Unix System Services users.
uint32 NumberOfProcesses	Total number of UNIX processes and active address spaces.
uint32 MaxNumberOfProcesses	The maximum number of processes configured in MaxProcSys.
uint64 MaxProcessMemorySize	The maximum number of KBytes of memory that can be allocated to a process (RLIMIT_AS).
uint64 TotalVirtualMemorySize	Total number of KBytes of virtual memory available to the operating system.
uint64 FreeVirtualMemory	Number of KBytes of virtual memory currently unused and available.
uint64 FreePhysicalMemory	Number of KBytes of physical memory currently unused and available.
uint64 TotalVisibleMemorySize	The total amount of physical memory (in KBytes) available to the operating system.
uint64 SizeStoredInPagingFiles	The total number of KBytes that can be stored in the operating system's page data sets.

Property	Description
uint64 FreeSpaceInPagingFiles	The total number of KBytes currently free in the operating system's page data sets.

The following properties have data that may be specific to z/OS, or may map to z/OS specific attributes.

Property	Description
string LanguageEdition	eServer specific extension for the language version of the OS. For z/OS always returns 'en-US'.
string CodeSet	eServer specific extension for the default OS code page. For z/OS this returns the code page for the CIM server process.
uint32 DefaultPageSize	eServer specific extension. The default size of pages used by the virtual memory management in units of bytes. Always 4096 for z/OS.
string SysplexName	The name of the z/OS Sysplex to which this operating system belongs.
string FMID	Function modification identifier of the z/OS operating system.
uint32 LastBootUpDuration	Indicates the time in seconds used to complete the IPL.
string IPLProfile[]	HMC profile from which the operating system was IPLed. <code>IPLProfile</code> contains 4 elements: ipaiodfu IODF unit address ipaloads LOADxx suffix ipaprompt Operator prompt flag ipanucid Nucleus ID

CIM_RunningOS

This class associates a computer system with the currently running operating system (see Figure 8 on page 83).

The z/OS specific subclass is **IBMzOS_RunningOS**.

IBMzOS_RunningOS

This class is for use by clients to find associations between a computer system and the operating system that is currently running on the computer system.

CIM_Process

This class represents currently active processes on an operating system. For z/OS this is mapped to address spaces and USS processes.

The z/OS specific subclasses are:

- **IBMzOS_Process** (for address spaces)
- **IBMzOS_UnixProcess** (for USS processes)

IBMzOS_Process

This class provides basic process information such as process name, priority, and run-time state. Instances of class **IBMzOS_Process** are mapped to z/OS address

spaces. Client applications can use this class to give clients an understanding of the processes (address spaces) running on the managed system within the context of their operating system.

The following properties are common for @server:

Property	Description
string Name	The name of the z/OS address space.
string Handle [key]	The decimal representation of the address space ID(ASID).
uint32 Priority	The address space's dispatching priority.
datetime CreationDate	The time when the address space was created.
uint64 KernelModeTime	Not supported for z/OS.
uint64 UserModeTime	Not supported for z/OS.

The following properties have data that may be specific to z/OS, or may map to z/OS specific attributes.

Property	Description
string ProcessOwner	The primary z/OS user ID under which an address space was started.
uint16 ProcessType	The type of address space. Possible values are: 0 (Other), 1 (TSO User), 2 (Started Task), 3 (Job), 4 (System Address Space), 5 (Initiator).

Note: z/OS also provides the notion of a UNIX process through the UNIX System Services (USS). In addition, those processes running under USS are supported by the extra IBMzOS_UnixProcess class which is derived from class CIM_UnixProcess. When a client enumerates all instances of class CIM_Process, it gets the complete list of z/OS address spaces, as well as all processes running under USS. However, if the client enumerates the instances of class IBMzOS_Process directly, it only gets the list of address spaces since class IBMzOS_UnixProcess is not derived from IBMzOS_Process but only from CIM_UnixProcess. Ideally, IBMzOS_UnixProcess should inherit from IBMzOS_Process, besides inheriting from CIM_UnixProcess, however, multiple inheritance is not the current standard in CIM version 2. For inheritance information of the mentioned classes refer to Figure 8 on page 83.

IBMzOS_UnixProcess

This class provides basic information about z/OS processes running in the Unix System Services (USS) subsystem. It supports all properties from CIM_Process plus a set of properties typical for Unix processes.

Class IBMzOS_UnixProcess is not derived from IBMzOS_Process, and therefore no instances of IBMzOS_UnixProcess are returned when a client enumerates the instances of class IBMzOS_Process, rather than class CIM_Process.

The following properties are common for @server:

Property	Description
string Name	The name of the z/OS Unix process. This is usually the name of the executable that started the process.
string Handle [key]	The z/OS Unix process ID.
uint32 Priority	The process priority.
uint16 ExecutionState	The process state (ready, blocked, suspended, stopped, and so on).
datetime CreationDate	The time when the process was started.
uint64 KernelModeTime	Not supported on z/OS.
uint64 UserModeTime	Not supported on z/OS.
string ParentProcessID	The parent process ID.
uint64 RealUserID	The real user ID.
uint64 ProcessGroupID	The process group ID.
uint64 ProcessSessionID	The process session ID.
string ProcessTTY	The TTY currently associated with this process.
string ModulePath	The executing process's command path.
string Parameters[]	The operating system parameters provided to the executing process. These are the argv[] values.

Class IBMzOS_UnixProcess has no z/OS specific properties.

CIM_OSProcess

This class associates an operating system with the set of currently active address spaces and USS processes.

The z/OS specific subclass is **IBMzOS_OSProcess**.

IBMzOS_OSProcess

This class provides a link between the operating system and process(es) running in the context of this operating system. Client applications can use this provider to give clients an understanding of the processes (jobs) running on the managed system within the context of its operating system.

OS management BaseBoard class

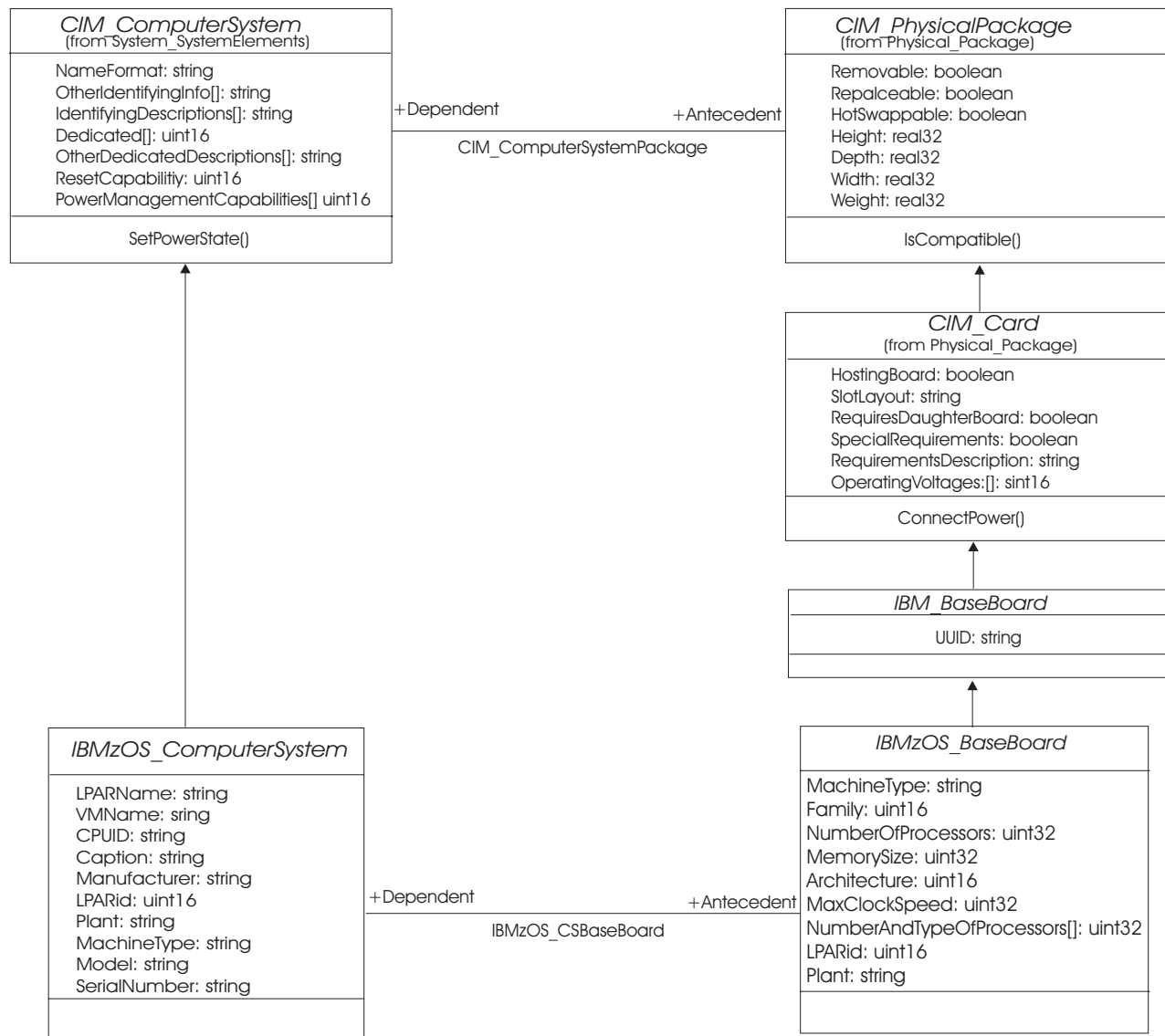


Figure 9. OS management BaseBoard Class

IBM_BaseBoard

This class represents the unique characteristics of the physical hardware as recognized by the z/OS operating system running on that hardware (the inband view). On most platforms these are the characteristics of the main board, and therefore, the name **IBM_BaseBoard** was chosen for this class. Instances of this class are either identified by a unique ID that was assigned to the main board (property **UUID**) or by the combination of manufacturer, model and serial number. The major purpose of this class is to provide the ability to determine which instances of computer systems are running on the same physical hardware.

The z/OS specific subclass is **IBMzOS_BaseBoard**.

The following properties are common for @server:

Property	Description
string Caption	Always returns 'Base Board'.
string Description	Always returns 'A class derived from Card to deliver the systems base board hardware information.'
string ElementName	Same as property <i>Tag</i> .
string Tag [key]	A combination of manufacturer, model and serial number in the following format: manufacturer:model:serialnumber.
string CreationClassName [key]	Always returns 'IBMzOS_BaseBoard'.
string SerialNumber	IBM allocated number used to identify the CEC.
string Model	The model number of the CEC, for example '314'.
string Manufacturer	The name of the company that produced the CEC.
string PartNumber	Not supported for z/OS.
boolean HostingBoard	Always returns TRUE, indicating that this card is a main board.
string UUID	The unique ID assigned to the main board. For z/OS, no value is supplied for this property, but it is maintained for compatibility with the other IBM @server platforms.

IBMzOS_BaseBoard

The following properties have data that may be specific to z/OS, or may map to z/OS specific attributes.

Property	Description
string MachineType	Processor type for the class of this z/OS server, for example: 2084
uint16 Family	The processor family. For z/OS, a value of 204 (z/Architecture base) is returned.
uint32 NumberOfProcessors	The number of general purpose processors installed on the system board.
uint32 MemorySize	The total amount of physical memory (in Kbytes) available to the operating system through which this data was provided. Note that this is not the total amount of installed memory for the zSeries CEC. This is the inband view of z/OS.
uint16 Architecture	The processor architecture.
uint32 NumberAndTypeOfProcessors[]	An array of uint32 where the first element is the number of general purpose processors, the second element is the number of zAAPs, the third element is the number of zIIPs, if supported.
uint16 LPARId	Logical partition number. This number distinguishes the configuration from all other level-2 configurations provided by the same LPAR hypervisor.
string Plant	Plant of manufacturer for the CPU.

CIM_ComputerSystemPackage association

This class associates a **ComputerSystem** with the physical main board(s) of the system(s) on which it runs.

The z/OS specific subclass is **IBMzOS_CSBaseBoard**.

IBMzOS_CSBaseBoard

This class associates a z/OS computer system with the physical zSeries CEC on which it runs (see Figure 9 on page 89). It has no properties.

OS management Processor class

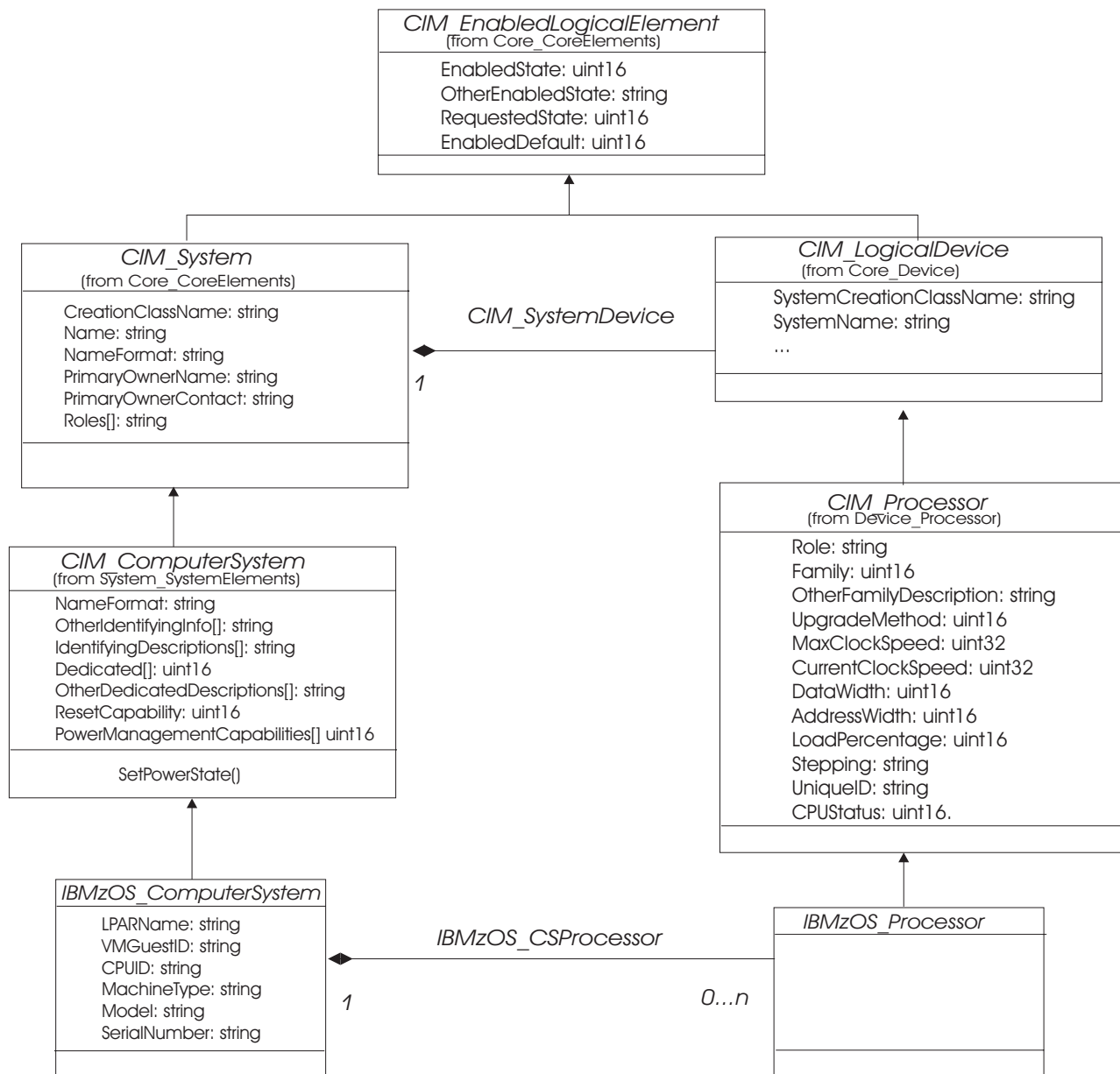


Figure 10. OS management Processor classes

CIM_SystemDevice association

This class associates a **ComputerSystem** with the instrumented processors.

The z/OS specific subclass is **IBMzOS_CSProcessor**.

CIM_Processor

This class represents the physical processors that are available to the operating system.

The z/OS specific subclass is **IBMzOS_Processor**.

IBMzOS_Processor

The following properties are common for @server:

Property	Description
string Caption	Always set to 'zSeries logical processor'.
string Description	Always set to 'This class represents instances of processors currently available to the z/OS operating system'.
string ElementName	Same as DeviceID.
string DeviceID [key]	Concatenation of the CPUID of the physical processor (PCCACPID) + colon (':') + CPU address. CBA987654321:2 is an example for a valid DeviceID. If a CPU is in Reserved or Offline state, the CPUID is FFFFFFFFFF.
uint16 EnabledState	2 Online 3 Reserved 6 Offline 9 Offline by WLM
string Role	CP Central Processor (including zEAP Processors) ZIIP zIIP processor ZAAP zAAP processor UNKNOWN no assigned role
uint16 Family	200 (= 'S/390® and zSeries Family').
string OtherFamilyDescription	'S/390 and zSeries Family' or specific model like 'z990'.
uint32 MaxClockSpeed	Not supported for z/OS.
uint32 CurrentClockSpeed	Not supported for z/OS.
uint16 LoadPercentage	For z/OS provided through RMF metrics provider only.
string Stepping	Not supported for z/OS.
string UniqueID	CPUID of the physical processor (PCCACPID).
uint16 CPUStatus	Not supported for z/OS.

Class IBMzOS_Processor has no z/OS specific properties.

OS management Logical Disk class

Figure 11 on page 94 illustrates the relationship between the IBM extension classes, and the CIM Base classes that they extend. This figure focuses on class *IBMzOS_LogicalDisk* which was provided in z/OS 1.9 CIM server to support the management of logical disks.

The packages, in which the classes are defined in the CIM Schema, are indicated in parenthesis.

Note: The described metrics are only available for active disks, but not for inactive or offline disks.

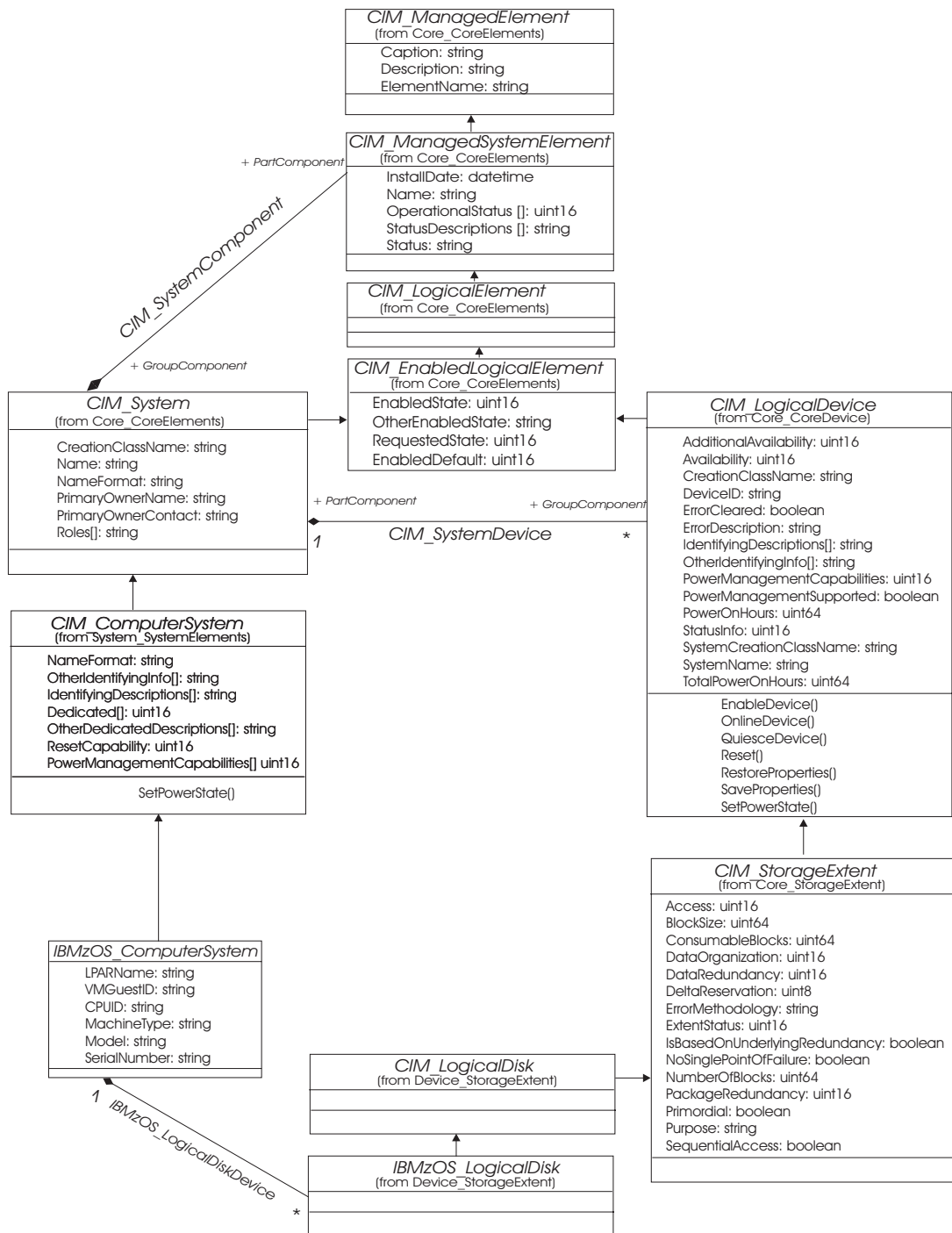


Figure 11. CIM Base classes extended by z/OS-specific classes (2)

CIM_LogicalDisk

This class represents logical disks attached to an operating system.

The z/OS specific subclass is **IBMzOS_LogicalDisk**.

IBMzOS_LogicalDisk

This class provides basic information about disk devices known to the z/OS operating system based on the logical view.

Property	Description
string Caption	Always returns “z/OS Storage Volume”
string Description	Always returns “Represents a storage volume as seen by z/OS”
string ElementName	Same as Name (Volume Serial Number)
string Name	Volume Serial Number obtained from UCBVOLI through UCBSCAN
uint16 OperationalStatus	“Unknown” (0) For “pending offline” = Shutting Down (4)
uint16 EnabledState	Mapped from the UCBONLI and UCBBOX values retrieved through UCBSCAN. See Table 7 for mapping values of <i>EnabledState</i> to system data.
string CreationClassName [key]	Always returns “IBMzOS_LogicalDisk”
string DeviceID [key]	Channel Device ID obtained from UCBCHAN through UCBSCAN
string[] IdentifyingDescriptions	[0]=Device Node Element Description obtained from the NEDID field of the matching IHACDR control block.
string[] OtherIdentifyingInfo	[0]=“Device Node Element Descriptor”
string SystemCreationClassName [key]	Always returns “IBMzOS_ComputerSystem”
string SystemName [key]	The systems fully qualified hostname (see <i>IBMzOS_ComputerSystem:colon;Name</i>). Obtained through the <i>OSBase_Common.get_system_name()</i> function.

The property *enabledState* is set based on the UCB control block information as shown in the following table:

Table 7. UCB control block information

UCBONLI	UCBBBOX	
	Boxed	Not boxed
Online	Quiesce (9)	Enabled (2)
Offline	Disabled (3)	Disabled (3)
Pending Offline	Shutting down (4)	

IBMzOS_LogicalDiskDevice association

This class associates an *IBMzOS_ComputerSystem* with an *IBMzOS_LogicalDisk*.

OS management File System classes

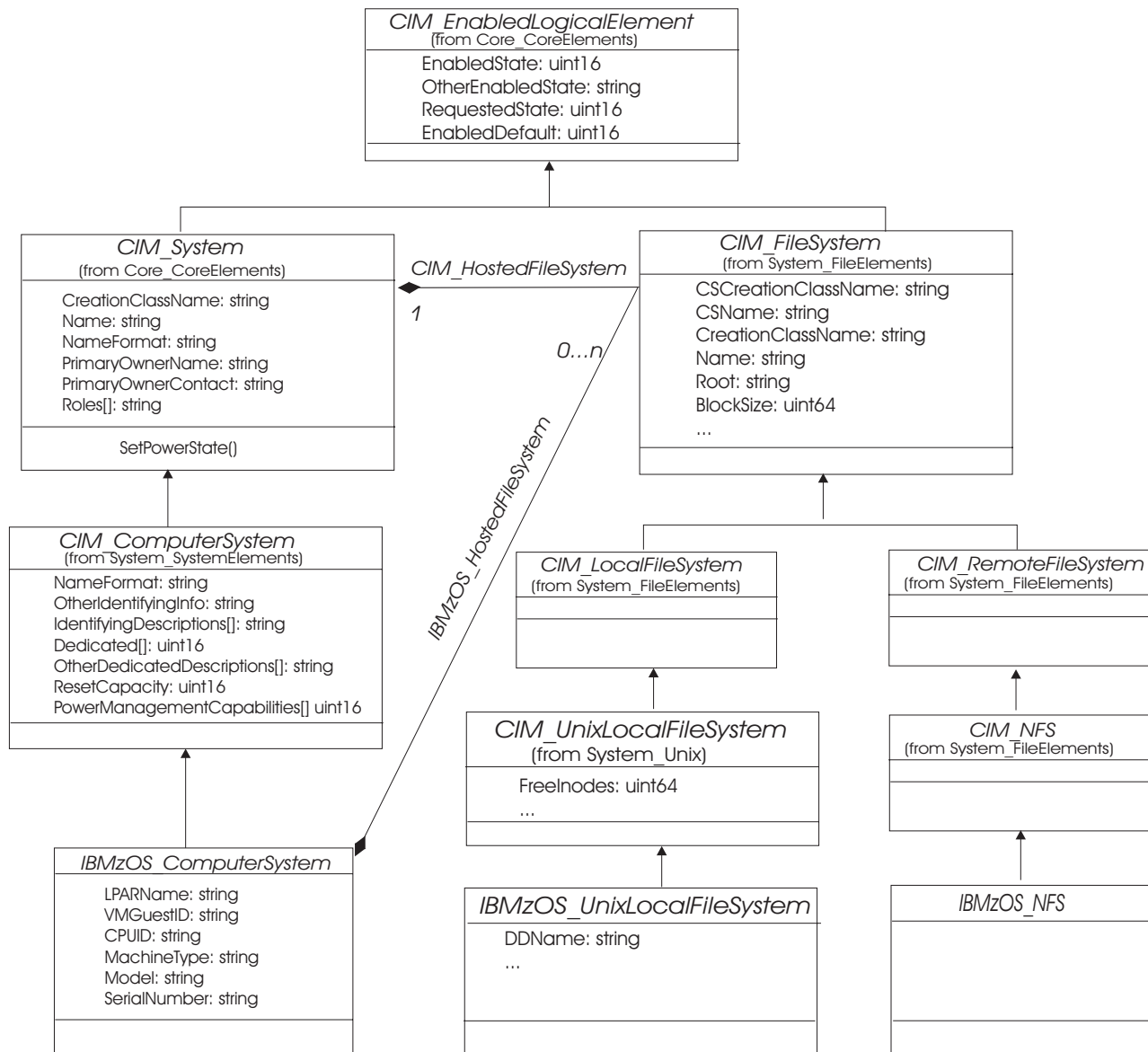


Figure 12. OS management File System classes

CIM_HostedFileSystem

The CIM_HostedFileSystem association associates a ComputerSystem with the set of currently mounted USS file systems.

The z/OS specific subclass is **IBMzOS_HostedFileSystem**.

CIM_LocalFileSystem

This class represents file systems that are locally attached to a computer system. On z/OS, hierarchical file systems HFS and zFS are supported.

The z/OS specific subclass is **IBMzOS_UnixLocalFileSystem**.

IBMzOS_UnixLocalFileSystem

The following properties are common for @server:

Property	Description
string Caption	Always set to 'z/OS hierarchical local file system'.
string Description	Always set to 'This class represents instances of currently mounted local hierarchical file systems'.
string ElementName	Same as Name.
string Name [key]	File system name (z/OS data set name).
string Root	Name of the directory where the file system is mounted.
uint64 FileSystemSize.	File system size in bytes.
uint64 AvailableSpace	Space available on the file system in bytes.
boolean ReadOnly	Indicates whether the file system is mounted read only.
string FileSystemType	File system type, for example 'NFS'.

The following properties have data that may be specific to z/OS, or may map to z/OS specific attributes.

Property	Description
DDName	DD name that was specified on mount.
FSParentDeviceID	Device ID of the parent file system.
FSDeviceID	Device number which the STAT command will return for all files in this file system.
MountParameters	The parameters that were specified for the mount command.
FSOwner	MVS Owner ID of the file system.
FSTypeName	The file system type name from the PARMLIB statement.

CIM_RemoteFileSystem

This class represents file systems that are accessed remotely by a computer system. On z/OS, only NFS is supported.

The z/OS specific subclass is **IBMzOS_NFS**.

IBMzOS_NFS

The following properties are common for @server:

Property	Description
string Caption	Always set to 'z/OS mounted network file system'.
string Description	Always set to 'This class represents instances of currently mounted network file systems'.
string ElementName	Same as Name.
string Name [key]	File system name (corresponds to the <i>file system argument of the mount command</i>).

Property	Description
string Root	Name of the directory where the file system is mounted.
uint64 FileSystemSize	File system size in bytes.
uint64 AvailableSpace	Space available the on file system in bytes.
boolean ReadOnly	Indicates whether the file system is mounted read only.
string FileSystemType	File system type, for example 'NFS'.

Class IBMzOS_NFS has no z/OS specific properties.

OS management Network classes

The classes described in this section are implemented by the z/OS Communication Server. For details on these CIM classes, refer to *z/OS Communications Server: IP Configuration Guide*.

Access to TCP/IP stack data is controlled by a security resource. Such a security resource is required if a user ID, associated with the client of the CIM server, is not defined as a z/OS UNIX superuser. The resource name is **EZB.CIMPROV.sysname.tcpname**. It is defined in the SERVAUTH class. Access is granted if the user ID associated with the client of the CIM server is permitted for READ access to the resource.

The providers are installed in the /usr/lpp/tcpip/lib hierarchical file system directory and linked to the CIM server's provider directory.

The z/OS CS CIM class definition and provider registration files are installed in the /usr/lpp/tcpip/mof hierarchical file system directory and are already integrated into the CIM server.

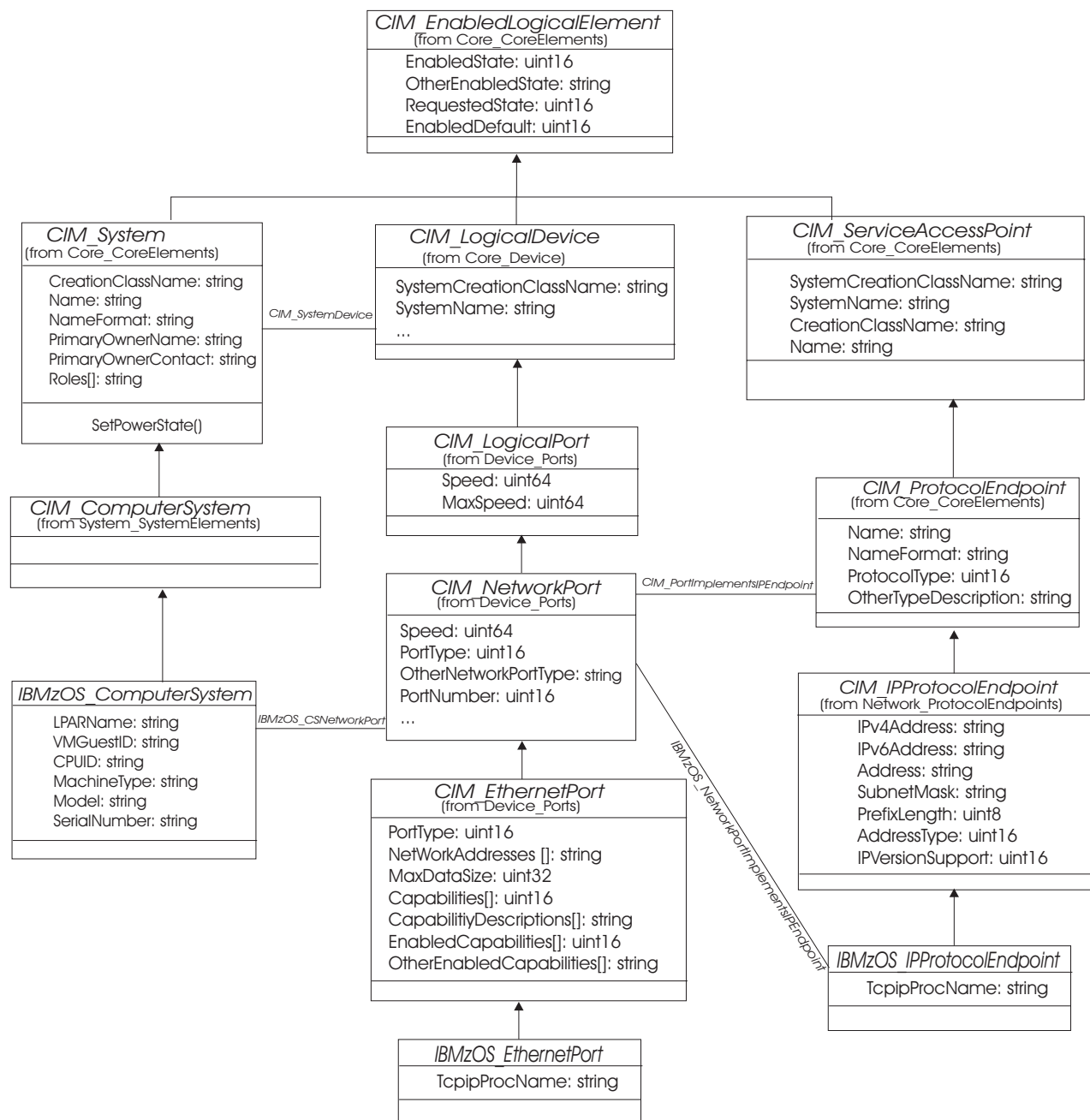


Figure 13. OS management Network classes

CIM_EthernetPort

This class represents network ports (interfaces) of type ethernet. For z/OS, all the ethernet interfaces configured to the TCP/IP stacks on the MVS image are supported.

The z/OS specific subclass is **IBMzOS_EthernetPort**.

IBMzOS_EthernetPort

The following properties are common for @server:

Property	Description
string Caption	Always set to 'IBMzOS EthernetPort'.
string Description	Variable, depending on the type of interface, for example, 'IP Assist Queued Direct I/O Ethernet protocol port'.
string ElementName	Same as Name.
string Name	The label by which the NetworkPort is known to the operating system ('tcpprocname_intfname').
uint16 EnabledState	Indicates whether the protocol endpoint is active or not.
string DeviceID [key]	Identifying information to uniquely name the ethernet port. ('tcpprocname_intfname').
uint64 Speed	The current bandwidth of the port in bits per second.
uint64 MaxSpeed	The maximum bandwidth of the port in bits per second. For z/OS, this is always the same value as <i>Speed</i> .
uint16 LinkTechnology	Always 2 (=Ethernet).
string OtherLinkTechnology	Not set for z/OS.

The following properties have data that may be specific to z/OS, or may map to z/OS specific attributes.

Property	Description
TcpipProcName	z/OS TCP/IP stack name.

CIM_PortImplementsEndpoint

This class associates a network port with its installed network protocols. Currently, only IP protocols defined for Ethernet ports are returned.

The z/OS specific subclass is **IBMzOS_NetworkPortImplementsIPEndpoint**.

CIM_IPProtocolEndpoint

This class represents the installed IP protocols. For z/OS, all IPv4 addresses configured to the TCP/IP stacks on the MVS image are supported.

The z/OS specific subclass is **IBMzOS_IPProtocolEndPoint**.

IBMzOS_IPProtocolEndpoint

The following properties are common for @server:

Property	Description
string Caption	Always set to 'IBMzOS Protocol Endpoint for IP'.
string Description	Always set to 'A communication point to send and receive data. This class is dedicated to relate IP interfaces to Logical Networks'.

Property	Description
string ElementName	Same as Name.
string Name [key]	The unique name of the protocol endpoint, constructed according to the template in NameFormat.
uint16 EnabledState	Returns whether the protocol endpoint is active or not.
string NameFormat	Describes the format of the name property. For z/OS, this is always set to 'TCPIPPROCNAME_TYPE_DEVICE_IPADDR(_ETH)'.
string IPv4Address	The IPv4 IP address.
string IPv6Address	Not yet supported for z/OS instrumentation.
string SubnetMask	The IPv4 IP subnet mask.
uint16 IPVersionSupport	Always returns 1 (=IPv4 only).

The following properties have data that may be specific to z/OS, or may map to z/OS specific attributes.

Property	Description
TcpipProcName	z/OS TCP/IP stack name.

CIM_SystemDevice association

This class associates a **ComputerSystem** with the instrumented network ethernet ports.

The z/OS specific subclass is **IBMzOS_CSNetworkPort**.

OS management Job classes

The classes described in this section are implemented by z/OS to instrument the z/OS jobs subsystems, JES2 and JES3.

For using these providers you need an extra security setup as described in "Appendix D. Cluster / CoupleDataset / JES2-JES3Jobs Provider Setup" on page 193.

For a list of the Jobs providers' reason codes, see "Appendix F. Reason codes" on page 203.

IBMzOS_Job

This class represents a z/OS job. Jobs are associated with a subsystem, such as JES2, JES3, or MSTR. Some properties may require significant overhead, including I/O, to obtain their data. These properties are identified with the qualifier of "Expensive". To reduce system overhead, the provider will only return the values for these expensive properties if they are explicitly requested by name.

IBMzOS_JES2Job

This class is a subclass of IBMzOS_Job and contains those properties that are unique to a job that has run, or will run, under JES2.

The following properties have been implemented for z/OS:

Property	Description
string Caption	A short description of the class. Returns <i>'IBM z/OS JES2 Job'</i> .
string Description	A description of the class. Returns <i>'This is an IBMzOS_JES2Job'</i> .
string ElementName	Name given to this instance of the class (same as Name)
datetime InstallDate	Not supported for z/OS.
string Name [key]	The property is overridden by <i>IBMzOS_Job</i> . It contains a unique identifier for this job.

Property	Description
uint16 OperationalStatus[]	The current status of the JES2 job. 1 No subchain exists 2 Active in CI in FSS address space 3 Awaiting postscan (batch) 4 Awaiting postscan (damsel) 5 Awaiting volume fetch 6 Awaiting start setup (JES3), Awaiting setup (JES2) 7 Awaiting/active in MDS system select processing 8 Awaiting resource allocation 9 Awaiting unavailable volumes 10 Awaiting volume mounts 11 Awaiting/active in MDS system verify processing 12 Error during MDS processing 13 Awaiting selection on main (JES3), Awaiting execution (JES2) 14 Scheduled on main (JES3), Active executing (JES2) .. Obsolete (ASP) 17 Awaiting breakdown (JES3), Active in output (JES2) 18 Awaiting MDS restart processing 19 Main MDS processing complete 20 Awaiting output service (JES3), Awaiting hardcopy (JES2) 21 Awaiting output service writer 22 Awaiting reserved services 23 Output service complete 24 Awaiting selection on main (demand select job) 25 Ending function rq waiting or I/O completion 26 Ending function rq not processed 27 Maximum rq index value .. Unknown 128 Active in input processing 129 Awaiting conversion 130 Active in conversion 131 Active in setup 132 Active in spin 133 Awaiting output 134 Awaiting purge 135 Active in purge 136 Active on NJE sysout received 137 Awaiting NJE transmission 138 Active on NJE job transmitter .. Unknown
string StatusDescriptions[]	Strings describing the various <i>OperationalStatus</i> values. Returns NULL.
string Status	Not supported for z/OS.
string JobStatus	A free form string containing information about the job. The primary job status is reflected in <i>OperationalStatus</i>. <i>JobStatus</i> provides additional implementation-specific details.

Property	Description
datetime TimeSubmitted	The time that the Job was submitted to execute. A value of all zeros indicates that the owning element is not capable of reporting a date and time. Therefore, the <i>ScheduledStartTime</i> and <i>StartTime</i> are reported as intervals relative to the time their values are required.
datetime ScheduledStartTime	Not supported for z/OS.
datetime StartTime	The time that the Job was actually started. This may be represented by an actual date and time, or by an interval relative to the time that this property is requested. Note that this property is also present in the <i>JobProcessingStatistics</i> class. This is necessary to capture the processing information for recurring Jobs, since only the 'last' run time can be stored in this single-valued property.
datetime ElapsedTime	The time interval that the Job has been executing or the total execution time if the Job is complete. Note that this property is also present in the <i>JobProcessingStatistics</i> class. This is necessary to capture the processing information for recurring Jobs, since only the 'last' run time can be stored in this single-valued property.
uint32 JobRunTimes	Number of times that the Job should be run. A value of 1 indicates that the Job is NOT recurring, while any non-zero value indicates a limit to the number of time that the Job will recur. Zero indicates that there is no limit to the number of times that the Job can be processed, but that it is terminated either AFTER the <i>UntilTime</i>, or by manual intervention. By default, a job is processed once. This property is not modifiable.

Property	Description																								
uint8 RunMonth	The month during which the Job should be processed. Permitted values are: <table> <tr><td>0</td><td>January</td></tr> <tr><td>1</td><td>February</td></tr> <tr><td>2</td><td>March</td></tr> <tr><td>3</td><td>April</td></tr> <tr><td>4</td><td>May</td></tr> <tr><td>5</td><td>June</td></tr> <tr><td>6</td><td>July</td></tr> <tr><td>7</td><td>August</td></tr> <tr><td>8</td><td>September</td></tr> <tr><td>9</td><td>October</td></tr> <tr><td>10</td><td>November</td></tr> <tr><td>11</td><td>December</td></tr> </table> This property is not modifiable.	0	January	1	February	2	March	3	April	4	May	5	June	6	July	7	August	8	September	9	October	10	November	11	December
0	January																								
1	February																								
2	March																								
3	April																								
4	May																								
5	June																								
6	July																								
7	August																								
8	September																								
9	October																								
10	November																								
11	December																								
sint8 RunDay	The day in the month on which the Job should be processed. There are two different interpretations for this property, depending on the value of <i>DayOfWeek</i>. In one case, <i>RunDay</i> defines the day-in-month on which the job is processed. This interpretation is used when the <i>DayOfWeek</i> is 0. A positive or negative integer indicates whether the <i>RunDay</i> should be calculated from the beginning or the end of the month. For example, 5 indicates the fifth day in the <i>RunMonth</i> and -1 indicates the last day in the <i>RunMonth</i>. When <i>RunDayOfWeek</i> is not 0, <i>RunDay</i> is the day-in-month on which the Job is processed, defined in conjunction with <i>RunDayOfWeek</i>. For example, if <i>RunDay</i> is 15 and <i>RunDayOfWeek</i> is Saturday, then the Job is processed on the first Saturday on or AFTER the 15th day in the <i>RunMonth</i> (e.g., the third Saturday in the month). If <i>RunDay</i> is 20 and <i>RunDayOfWeek</i> is -Saturday, then this indicates the first Saturday on or BEFORE the 20th day in the <i>RunMonth</i>. If <i>RunDay</i> is -1 and <i>RunDayOfWeek</i> is -Sunday, then this indicates the last Sunday in the <i>RunMonth</i>. This property is not modifiable.																								

Property	Description																														
sint8 RunDayOfWeek	Positive or negative integer used in conjunction with RunDay to indicate the day of the week on which the Job is processed. RunDayOfWeek is set to 0 to indicate an exact day of the month, such as March 1. A positive integer (representing Sunday, Monday, ... Saturday) means that the day of week is found on or AFTER the specified RunDay. A negative integer (representing -Sunday, -Monday, ... -Saturday) means that the day of week is found on or BEFORE the RunDay. Permitted values are: <table> <tr><td>0</td><td>ExactDayOfWeek</td></tr> <tr><td>1</td><td>Sunday</td></tr> <tr><td>2</td><td>Monday</td></tr> <tr><td>3</td><td>Tuesday</td></tr> <tr><td>4</td><td>Wednesday</td></tr> <tr><td>5</td><td>Thursday</td></tr> <tr><td>6</td><td>Friday</td></tr> <tr><td>7</td><td>Saturday</td></tr> <tr><td>-1</td><td>-Saturday</td></tr> <tr><td>-2</td><td>-Monday</td></tr> <tr><td>-3</td><td>-Tuesday</td></tr> <tr><td>-4</td><td>-Wednesday</td></tr> <tr><td>-5</td><td>-Thursday</td></tr> <tr><td>-6</td><td>-Friday</td></tr> <tr><td>-7</td><td>-Saturday</td></tr> </table> This property is not modifiable.	0	ExactDayOfWeek	1	Sunday	2	Monday	3	Tuesday	4	Wednesday	5	Thursday	6	Friday	7	Saturday	-1	-Saturday	-2	-Monday	-3	-Tuesday	-4	-Wednesday	-5	-Thursday	-6	-Friday	-7	-Saturday
0	ExactDayOfWeek																														
1	Sunday																														
2	Monday																														
3	Tuesday																														
4	Wednesday																														
5	Thursday																														
6	Friday																														
7	Saturday																														
-1	-Saturday																														
-2	-Monday																														
-3	-Tuesday																														
-4	-Wednesday																														
-5	-Thursday																														
-6	-Friday																														
-7	-Saturday																														
datetime RunStartInterval	The time interval after midnight when the Job should be processed. For example, 00000000020000.000000:000 indicates that the Job should be run on or after two o'clock, local time of UTC time (distinguished using the LocalOrUtcTime property). This property is not modifiable.																														
uint16 LocalOrUtcTime	This property indicates whether the time represented in the RunStartInterval and UntilTime properties represent local or UTC times. Time values are synchronized worldwide by using the enumeration value 2, "UTC Time". Permitted values are: <table> <tr><td>1</td><td>Local time</td></tr> <tr><td>2</td><td>UTC time</td></tr> </table> This property is not modifiable.	1	Local time	2	UTC time																										
1	Local time																														
2	UTC time																														

Property	Description
datetime UntilTime	The time after which the Job is invalid or should be stopped. This may be represented by an actual date and time, or by an interval relative to the time that this property is requested. A value of all nines indicates that the Job can run indefinitely. This property is not modifiable.
string Notify	User to be notified upon the Job completion or failure. This property can be modified using the <i>RequestPropertyChange()</i> method.
string Owner	The User that submitted the Job or the Service/method name/etc. that caused the job to be created.
uint32 Priority	Indicates the urgency or importance of execution of the Job. The lower the number, the higher the priority. Note that this property is also present in the <i>JobProcessingStatistics</i> class. This is necessary to capture the setting information that would influence a Job's results. This property can be modified using the <i>RequestPropertyChange()</i> method.
uint16 PercentComplete	Not supported for z/OS.
boolean DeleteOnCompletion	Indicates whether or not the Job should be automatically deleted upon completion. Note that the 'completion' of a recurring Job is defined by its <i>JobRunTimes</i> or <i>UntilTime</i> properties, OR when the Job is terminated by manual intervention. If this property is set to false and the Job completes, then the extrinsic method <i>DeleteInstance</i> MUST be used to delete the Job versus updating this property. This property is not modifiable.
uint16 ErrorCode	Not supported for z/OS.
string ErrorDescription	Not supported for z/OS.
uint16 RecoveryAction	Not supported for z/OS.
string OtherRecoveryAction	Not supported for z/OS.
string AbendCode	Job completed with abend code.
string AccountNumber	Account number from job card.
boolean ARMRegistered	Job is ARM registered indicator.

Property	Description
string AvailableSchedEnvSystem []	System names on which the scheduling environment required by job is available. Only valid if job requires a scheduling environment and that environment is available on at least one system.
string AvailableSeclabelSystems []	System names on which the seclabel associated with the job is available. Only valid if seclabel by system is active in the security product and the seclabel is available on at least on system.
boolean AwaitingARMRestart	Job awaiting ARM restart indicator.
string Building	NJE building. This property is "Expensive".
uint32 CardCount	Card (output) count.
string Class	Job class. This property can be modified using the <i>RequestPropertyChange()</i> method.
uint32 CompletionCode	Completion code (set for conditions marked with + in job completion indicator).
uint8 CompletionType	Specific completion type: 0 No completion info 1 Job ended normally 2 Job ended by CC 3 JCL error 4 Canceled 5 Abended 6 Converter abended 7 Security error 8 Job failed in EOM .. Unknown
uint16 CopyCount	Job copy count. This property is "Expensive".
string CSName	The scoping Computer System.
string DefaultPrintDest	Default print destination. This property can be modified using the <i>RequestPropertyChange()</i> method.
string DefaultPunchDest	Default punch destination. This property can be modified using the <i>RequestPropertyChange()</i> method.
string Department	NJE department. This property is "Expensive".
string Device	Name of device job is active on.

Property	Description
uint32 EstimatedTimeToExecution	Estimated time to execution in seconds. This field is only available if the job is awaiting execution, job is scheduled to run to a WLM managed class, job is not held (duplicate job name, operator hold, etc.), member it has affinity to is available, and the scheduling environment is available.
datetime ExecutionEndTime	Execution end time and date. This property is "Expensive".
string ExecutionMember	Execution JES2 member name. This property is "Expensive".
string ExecutionNode	Execution node. This property can be modified using the <i>RequestPropertyChange()</i> method.
datetime ExecutionStartTime	Execution start time and date. This property is "Expensive".
string ExecutionSystem	Execution MVS system name. This property is "Expensive".
uint8 HoldIndicator	Job hold indicator: 1 Not held 2 Held 3 Held for duplicate job name .. Unknown
uint32 InputCount	Job input count. This property is "Expensive".
string InputDevice	Input device name. This property is "Expensive".
datetime InputStartTime	Input start time and date. This property is "Expensive".
string InputSystem	Input system or member.
boolean JesLogSpinnable	Jeslog spinnable indicator.
boolean JobClassModeWLM	Job class mode for job. If true, mode is WLM, otherwise mode is JES.
string JobID	Job identifier.
boolean JobIsActive	Indicate job is executing.
string JobName	Job name.
uint8 JobType	Job type: 1 Started task (STC) 2 Time sharing user (TSU) 3 Batch job (JOB) 4 APPC indicator .. Unknown

Property	Description
uint32 LineCount	Line count. This property is "Expensive".
string MemberName	JES2 member on which the job is active.
string MessageClass	Message class from job card.
string NotifyNode	Notify node. This property is "Expensive".
string OriginalJobID	Original job identifier.
string OriginNode	Original node (node of submittal).
string OSName	The scoping Operating System's name.
uint32 PageCount	Job page count. This property is "Expensive".

Property	Description
uint8 Phase	Phase job is in: 1 No subchain exists 2 Active in CI in FSS address space 3 Awaiting postscan (batch) 4 Awaiting postscan (damsel) 5 Awaiting volume fetch 6 Awaiting start setup (JES3), Awaiting setup (JES2) 7 Awaiting/active in MDS system select processing 8 Awaiting resource allocation 9 Awaiting unavailable volumes 10 Awaiting volume mounts 11 Awaiting/active in MDS system verify processing 12 Error during MDS processing 13 Awaiting selection on main (JES3), Awaiting execution (JES2) 14 Scheduled on main (JES3), Active executing (JES2) .. Obsolete (ASP) 17 Awaiting breakdown (JES3), Active in output (JES2) 18 Awaiting MDS restart processing 19 Main MDS processing complete 20 Awaiting output service (JES3), Awaiting hardcopy (JES2) 21 Awaiting output service writer 22 Awaiting reserved services 23 Output service complete 24 Awaiting selection on main (demand select job) 25 Ending function rq waiting or I/O completion 26 Ending function rq not processed 27 Maximum rq index value .. Unknown 128 Active in input processing 129 Awaiting conversion 130 Active in conversion 131 Active in setup 132 Active in spin 133 Awaiting output 134 Awaiting purge 135 Active in purge 136 Active on NJE sysout received 137 Awaiting NJE transmission 138 Active on NJE job transmitter .. Unknown
string ProgrammerName	Programmer name from job card.
string RoomNumber	Job card room number.
string Seclabel	Seclabel from job.
boolean Spin	Indicator of whether jobs in the job class can be spun.
string SubSystem	Subsystem name.
string SystemName	MVS system name on which the job is active.

Property	Description
uint32 WLMActiveJobCount	Number of active jobs in this WLM service class.
uint32 WLMJobsOnQueueCount	Number of jobs on WLM service class queue.
uint32 WLMPosition	Position of this job on WLM service class queue.
uint32 WLMschedulingEnvironment	WLM scheduling environment. This property can be modified using the <i>RequestPropertyChange()</i> method.
string WLMServiceClass	WLM service class. This property can be modified using the <i>RequestPropertyChange()</i> method.
boolean ConverterWait	Job can be converted only by CNVT PCEs that can wait for OS
boolean Independent	Job is set to independent mode.
uint32 JobKey	Job key
boolean JobNotRunReasonJobBusyOnDevice	Job not running because job busy on device
boolean JobNotRunReasonJobClassHeld	Job not running because job class held
boolean JobNotRunReasonJobClassLimitReached	Job not running because job class limit reached
boolean JobNotRunReasonNoSystem	Job not running because no system with right combination of resources
boolean JobNotRunReasonSchedulingEnvironment	Job not running due to unavailable scheduling environment
boolean JobNotRunReasonSeclabelAffinity	Job not running because of seclabel affinity
boolean JobNotRunReasonSpoolNotAvailable	Job not running because spools not available
boolean JobNotRunReasonSystemAffinity	Job not running due to system affinity
boolean Protected	Job is protected
uint32 SpoolDataToken	Spool data token
string SystemAffinity []	System affinity for job
boolean SystemDataSet	Job represents a system data set
uint32 TrackGroupCount	Number of track groups of spool space used by this job
uint32 TrackGroupPCT	Percentage of track groups used by this job

IBMzOS_JES3Job

This class is a subclass of IBMzOS_Job and contains those properties that are unique to a job that has run, or will run, under JES3.

The following properties have been implemented for z/OS:

Property	Description																																																																														
string Caption	A short description of the class																																																																														
string Description	A description of the class																																																																														
string ElementName	Name of given to this instance of the class																																																																														
datetime InstallDate	Not supported for z/OS.																																																																														
string Name	The property is overridden by <i>IBMzOS_Job</i> . It contains a unique identifier for this Job.																																																																														
uint16 OperationalStatus []	The current status of the JES3 Job: <table> <tr><td>1</td><td>No subchain exists</td></tr> <tr><td>2</td><td>Active in CI in FSS address space</td></tr> <tr><td>3</td><td>Awaiting postscan (batch)</td></tr> <tr><td>4</td><td>Awaiting postscan (damsel)</td></tr> <tr><td>5</td><td>Awaiting volume fetch</td></tr> <tr><td>6</td><td>Awaiting start setup (JES3), Awaiting setup (JES2)</td></tr> <tr><td>7</td><td>Awaiting/active in MDS system select processing</td></tr> <tr><td>8</td><td>Awaiting resource allocation</td></tr> <tr><td>9</td><td>Awaiting unavailable volumes</td></tr> <tr><td>10</td><td>Awaiting volume mounts</td></tr> <tr><td>11</td><td>Awaiting/active in MDS system verify processing</td></tr> <tr><td>12</td><td>Error during MDS processing</td></tr> <tr><td>13</td><td>Awaiting selection on main (JES3), Awaiting execution (JES2)</td></tr> <tr><td>14</td><td>Scheduled on main (JES3), Active executing (JES2)</td></tr> <tr><td>..</td><td>Obsolete (ASP)</td></tr> <tr><td>17</td><td>Awaiting breakdown (JES3), Active in output (JES2)</td></tr> <tr><td>18</td><td>Awaiting MDS restart processing</td></tr> <tr><td>19</td><td>Main MDS processing complete</td></tr> <tr><td>20</td><td>Awaiting output service (JES3), Awaiting hardcopy (JES2)</td></tr> <tr><td>21</td><td>Awaiting output service writer</td></tr> <tr><td>22</td><td>Awaiting reserved services</td></tr> <tr><td>23</td><td>Output service complete</td></tr> <tr><td>24</td><td>Awaiting selection on main (demand select job)</td></tr> <tr><td>25</td><td>Ending function rq waiting or I/O completion</td></tr> <tr><td>26</td><td>Ending function rq not processed</td></tr> <tr><td>27</td><td>Maximum rq index value</td></tr> <tr><td>..</td><td>Unknown</td></tr> <tr><td>128</td><td>Active in input processing</td></tr> <tr><td>129</td><td>Awaiting conversion</td></tr> <tr><td>130</td><td>Active in conversion</td></tr> <tr><td>131</td><td>Active in setup</td></tr> <tr><td>132</td><td>Active in spin</td></tr> <tr><td>133</td><td>Awaiting output</td></tr> <tr><td>134</td><td>Awaiting purge</td></tr> <tr><td>135</td><td>Active in purge</td></tr> <tr><td>136</td><td>Active on NJE sysout received</td></tr> <tr><td>137</td><td>Awaiting NJE transmission</td></tr> <tr><td>138</td><td>Active on NJE job transmitter</td></tr> <tr><td>..</td><td>Unknown</td></tr> </table>	1	No subchain exists	2	Active in CI in FSS address space	3	Awaiting postscan (batch)	4	Awaiting postscan (damsel)	5	Awaiting volume fetch	6	Awaiting start setup (JES3), Awaiting setup (JES2)	7	Awaiting/active in MDS system select processing	8	Awaiting resource allocation	9	Awaiting unavailable volumes	10	Awaiting volume mounts	11	Awaiting/active in MDS system verify processing	12	Error during MDS processing	13	Awaiting selection on main (JES3), Awaiting execution (JES2)	14	Scheduled on main (JES3), Active executing (JES2)	..	Obsolete (ASP)	17	Awaiting breakdown (JES3), Active in output (JES2)	18	Awaiting MDS restart processing	19	Main MDS processing complete	20	Awaiting output service (JES3), Awaiting hardcopy (JES2)	21	Awaiting output service writer	22	Awaiting reserved services	23	Output service complete	24	Awaiting selection on main (demand select job)	25	Ending function rq waiting or I/O completion	26	Ending function rq not processed	27	Maximum rq index value	..	Unknown	128	Active in input processing	129	Awaiting conversion	130	Active in conversion	131	Active in setup	132	Active in spin	133	Awaiting output	134	Awaiting purge	135	Active in purge	136	Active on NJE sysout received	137	Awaiting NJE transmission	138	Active on NJE job transmitter	..	Unknown
1	No subchain exists																																																																														
2	Active in CI in FSS address space																																																																														
3	Awaiting postscan (batch)																																																																														
4	Awaiting postscan (damsel)																																																																														
5	Awaiting volume fetch																																																																														
6	Awaiting start setup (JES3), Awaiting setup (JES2)																																																																														
7	Awaiting/active in MDS system select processing																																																																														
8	Awaiting resource allocation																																																																														
9	Awaiting unavailable volumes																																																																														
10	Awaiting volume mounts																																																																														
11	Awaiting/active in MDS system verify processing																																																																														
12	Error during MDS processing																																																																														
13	Awaiting selection on main (JES3), Awaiting execution (JES2)																																																																														
14	Scheduled on main (JES3), Active executing (JES2)																																																																														
..	Obsolete (ASP)																																																																														
17	Awaiting breakdown (JES3), Active in output (JES2)																																																																														
18	Awaiting MDS restart processing																																																																														
19	Main MDS processing complete																																																																														
20	Awaiting output service (JES3), Awaiting hardcopy (JES2)																																																																														
21	Awaiting output service writer																																																																														
22	Awaiting reserved services																																																																														
23	Output service complete																																																																														
24	Awaiting selection on main (demand select job)																																																																														
25	Ending function rq waiting or I/O completion																																																																														
26	Ending function rq not processed																																																																														
27	Maximum rq index value																																																																														
..	Unknown																																																																														
128	Active in input processing																																																																														
129	Awaiting conversion																																																																														
130	Active in conversion																																																																														
131	Active in setup																																																																														
132	Active in spin																																																																														
133	Awaiting output																																																																														
134	Awaiting purge																																																																														
135	Active in purge																																																																														
136	Active on NJE sysout received																																																																														
137	Awaiting NJE transmission																																																																														
138	Active on NJE job transmitter																																																																														
..	Unknown																																																																														

Property	Description
string StatusDescriptions []	Strings describing the various Operational Status values.
string Status	Not supported for z/OS.
string JobStatus	A free form string representing the Job's status. The primary status is reflected in the inherited <i>OperationStatus</i> property. JobStatus provides additional implementation-specific details.
datetime TimeSubmitted	The time that the Job was submitted to execute. A value of all zeros indicates that the owning element is not capable of reporting a date and time. Therefore, the <i>ScheduledStartTime</i> and <i>StartTime</i> are reported as intervals relative to the time their values are required.
datetime ScheduledStartTime	Not supported for z/OS.
datetime StartTime	The time that the Job was actually started. This may be represented by an actual date and time, or by an interval relative to the time that this property is requested. Note that this property is also present in the <i>JobProcessingStatistics</i> class. This is necessary to capture the processing information for recurring Jobs, since only the 'last' run time can be stored in this single-valued property.
datetime ElapsedTime	The time interval that the Job has been executing or the total execution time if the Job is complete. Note that this property is also present in the <i>JobProcessingStatistics</i> class. This is necessary to capture the processing information for recurring Jobs, since only the 'last' run time can be stored in this single-valued property.
uint32 JobRunTimes	Number of times that the Job should be run. A value of 1 indicates that the Job is NOT recurring, while any non-zero value indicates a limit to the number of time that the Job will recur. Zero indicates that there is no limit to the number of times that the Job can be processed, but that it is terminated either AFTER the <i>UntilTime</i>, or by manual intervention. By default, a Job is processed once. This property is not modifiable.

Property	Description																								
uint8 RunMonth	The month during which the job should be processed. Permitted values are: <table> <tr><td>0</td><td>January</td></tr> <tr><td>1</td><td>February</td></tr> <tr><td>2</td><td>March</td></tr> <tr><td>3</td><td>April</td></tr> <tr><td>4</td><td>May</td></tr> <tr><td>5</td><td>June</td></tr> <tr><td>6</td><td>July</td></tr> <tr><td>7</td><td>August</td></tr> <tr><td>8</td><td>September</td></tr> <tr><td>9</td><td>October</td></tr> <tr><td>10</td><td>November</td></tr> <tr><td>11</td><td>December</td></tr> </table> This property is not modifiable.	0	January	1	February	2	March	3	April	4	May	5	June	6	July	7	August	8	September	9	October	10	November	11	December
0	January																								
1	February																								
2	March																								
3	April																								
4	May																								
5	June																								
6	July																								
7	August																								
8	September																								
9	October																								
10	November																								
11	December																								
sint8 RunDay	The day in the month on which the Job should be processed. There are two different interpretations for this property, depending on the value of <i>DayOfWeek</i>. In one case, <i>RunDay</i> defines the day-in-month on which the job is processed. This interpretation is used when the <i>DayOfWeek</i> is 0. A positive or negative integer indicates whether the <i>RunDay</i> should be calculated from the beginning or the end of the month. For example, 5 indicates the fifth day in the <i>RunMonth</i> and -1 indicates the last day in the <i>RunMonth</i>. When <i>RunDayOfWeek</i> is not 0, <i>RunDay</i> is the day-in-month on which the Job is processed, defined in conjunction with <i>RunDayOfWeek</i>. For example, if <i>RunDay</i> is 15 and <i>RunDayOfWeek</i> is Saturday, then the Job is processed on the first Saturday on or AFTER the 15th day in the <i>RunMonth</i> (e.g., the third Saturday in the month). If <i>RunDay</i> is 20 and <i>RunDayOfWeek</i> is -Saturday, then this indicates the first Saturday on or BEFORE the 20th day in the <i>RunMonth</i>. If <i>RunDay</i> is -1 and <i>RunDayOfWeek</i> is -Sunday, then this indicates the last Sunday in the <i>RunMonth</i>. This property is not modifiable.																								

Property	Description																														
sint8 RunDayOfWeek	Positive or negative integer used in conjunction with <i>RunDay</i> to indicate the day of the week on which the Job is processed. <i>RunDayOfWeek</i> is set to 0 to indicate an exact day of the month, such as March 1. A positive integer (representing Sunday, Monday, ..., Saturday) means that the day of week is found on or AFTER the specified <i>RunDay</i>. A negative integer (representing -Sunday, -Monday, ..., -Saturday) means that the day of week is found on or BEFORE the <i>RunDay</i>. Permitted values are: <table> <tr><td>0</td><td>ExactDayOfWeek</td></tr> <tr><td>1</td><td>Sunday</td></tr> <tr><td>2</td><td>Monday</td></tr> <tr><td>3</td><td>Tuesday</td></tr> <tr><td>4</td><td>Wednesday</td></tr> <tr><td>5</td><td>Thursday</td></tr> <tr><td>6</td><td>Friday</td></tr> <tr><td>7</td><td>Saturday</td></tr> <tr><td>-1</td><td>-Saturday</td></tr> <tr><td>-2</td><td>-Monday</td></tr> <tr><td>-3</td><td>-Tuesday</td></tr> <tr><td>-4</td><td>-Wednesday</td></tr> <tr><td>-5</td><td>-Thursday</td></tr> <tr><td>-6</td><td>-Friday</td></tr> <tr><td>-7</td><td>-Saturday</td></tr> </table> This property is not modifiable.	0	ExactDayOfWeek	1	Sunday	2	Monday	3	Tuesday	4	Wednesday	5	Thursday	6	Friday	7	Saturday	-1	-Saturday	-2	-Monday	-3	-Tuesday	-4	-Wednesday	-5	-Thursday	-6	-Friday	-7	-Saturday
0	ExactDayOfWeek																														
1	Sunday																														
2	Monday																														
3	Tuesday																														
4	Wednesday																														
5	Thursday																														
6	Friday																														
7	Saturday																														
-1	-Saturday																														
-2	-Monday																														
-3	-Tuesday																														
-4	-Wednesday																														
-5	-Thursday																														
-6	-Friday																														
-7	-Saturday																														
datetime RunStartInterval	The time interval after midnight when the Job should be processed. For example, 000000000020000.000000:000 indicates that the Job should be run on or after two o'clock, local time of UTC time (distinguished using the <i>LocalOrUtcTime</i> property). This property is not modifiable.																														
uint16 LocalOrUtcTime	This property indicates whether the time represented in the <i>RunStartInterval</i> and <i>UntilTime</i> properties represent local or UTC times. Time values are synchronized worldwide by using the enumeration value 2, "UTC Time". Permitted values are: <table> <tr><td>1</td><td>Local time</td></tr> <tr><td>2</td><td>UTC time</td></tr> </table> This property is not modifiable.	1	Local time	2	UTC time																										
1	Local time																														
2	UTC time																														

Property	Description
datetime UntilTime	The time after which the Job is invalid or should be stopped. This may be represented by an actual date and time, or by an interval relative to the time that this property is requested. A value of all nines indicates that the Job can run indefinitely. This property is not modifiable.
string Notify	User to be notified upon the Job completion or failure. This property can be modified using the <i>RequestPropertyChange()</i> method.
string Owner	The User that submitted the Job or the Service/method name/etc. that caused the job to be created.
uint32 Priority	Indicates the urgency or importance of execution of the Job. The lower the number, the higher the priority. Note that this property is also present in the <i>JobProcessingStatistics</i> class. This is necessary to capture the setting information that would influence a Job's results. This property can be modified using the <i>RequestPropertyChange()</i> method.
uint16 PercentComplete	Not supported for z/OS.
boolean DeleteOnCompletion	Indicates whether or not the Job should be automatically deleted upon completion. Note that the 'completion' of a recurring Job is defined by its <i>JobRunTimes</i> or <i>UntilTime</i> properties, OR when the Job is terminated by manual intervention. If this property is set to false and the Job completes, then the extrinsic method <i>DeleteInstance</i> MUST be used to delete the Job versus updating this property. This property is not modifiable.
uint16 ErrorCode	Not supported for z/OS.
string ErrorDescription	Not supported for z/OS.
uint16 RecoveryAction	Not supported for z/OS.
string OtherRecoveryAction	Not supported for z/OS.
string AbendCode	Job completed with abend code.
string AccountNumber	Account number from job card.
boolean ARMRegistered	Job is ARM registered indicator.
string AvailableSchedEnvSystems []	System names on which the scheduling environment required by job is available. Only valid if job requires a scheduling environment and that environment is available on at least one system.

Property	Description
string AvailableSeclabelSystems []	System names on which the seclabel associated with the job is available. Only valid if seclabel by system is active in the security product and the seclabel is available on at least on system.
boolean AwaitingARMRestart	Job awaiting ARM restart indicator.
string Building	NJE building. This property is "Expensive".
uint32 CardCount	Card (output) count.
string Class	Job class. This property can be modified using the <i>RequestPropertyChange()</i> method.
uint32 CompletionCode	Completion code (set for conditions marked with + in job completion indicator.
uint8 CompletionType	Specific completion type: 0 No completion info 1 Job ended normally 2 Job ended by CC 3 JCL error 4 Canceled 5 Abended 6 Converter abended 7 Security error 8 Job failed in EOM .. Unknown
uint16 CopyCount	Job copy count. This property is "Expensive".
string CSName	The scoping Computer System.
string DefaultPrintDest	Default print destination. This property can be modified using the <i>RequestPropertyChange()</i> method.
string DefaultPunchDest	Default punch destination. This property can be modified using the <i>RequestPropertyChange()</i> method.
string Department	NJE department. This property is "Expensive".
string Device	Name of device job is active on.
uint32 EstimatedTimeToExecution	Estimated time to execution in seconds. This field is only available if the job is awaiting execution, job is scheduled to run to a WLM managed class, job is not held (duplicate job name, operator hold, etc.), member it has affinity to is available, and the scheduling environment is available.
datetime ExecutionEndTime	Execution end time and date. This property is "Expensive".

Property	Description
string ExecutionMember	Execution JES2 member name. This property is "Expensive".
string ExecutionNode	Execution node. This property can be modified using the <i>RequestPropertyChange()</i> method.
datetime ExecutionStartTime	Execution start time and date. This property is "Expensive".
string ExecutionSystem	Execution MVS system name. This property is "Expensive".
uint8 HoldIndicator	Job hold indicator: 1 Not held 2 Held 3 Held for duplicate job name .. Unknown
uint32 InputCount	Job input count. This property is "Expensive".
string InputDevice	Input device name. This property is "Expensive".
datetime InputStartTime	Input start time and date. This property is "Expensive".
string InputSystem	Input system or member.
boolean JesLogSpinnable	Jeslog spinnable indicator.
boolean JobClassModeWLM	Job class mode for job. If true, mode is WLM, otherwise mode is JES.
string JobID	Job identifier.
boolean JobIsActive	Indicate job is executing.
string JobName	Job name.
uint8 JobType	Job type: 1 Started task (STC) 2 Time sharing user (TSU) 3 Batch job (JOB) 4 APPC indicator" .. Unknown
uint32 LineCount	Line count. This property is "Expensive".
string MemberName	JES2 member on which the job is active.
string MessageClass	Message class from job card.
string NotifyNode	Notify node. This property is "Expensive".
string NotifyUserid	Notify user ID.
string OriginalJobID	Original job identifier.
string OriginNode	Original node (node of submittal).

Property	Description
string OSName	The scoping Operating System's name.
uint32 PageCount	Job page count. This property is "Expensive".
uint8 Phase	Phase, the job is in. For the values and their meanings, see property <i>OperationalStatus</i> .
string ProgrammerName	Programmer name from job card.
string RoomNumber	Job card room number.
string Seclabel	Seclabel from job.
boolean Spin	Indicator of whether jobs in the job class can be spun.
string SubSystem	Subsystem name.
string SystemName	MVS system name on which the job is active.
uint32 WLMActiveJobCount	Number of active jobs in this WLM service class.
uint32 WLMJobsOnQueueCount	Number of jobs on WLM service class queue.
uint32 WLMPosition	Position of this job on WLM service class queue.
uint32 WLMschedulingEnvironment	WLM scheduling environment. This property can be modified using the <i>RequestPropertyChange()</i> method.
string WLMServiceClass	WLM service class. This property can be modified using the <i>RequestPropertyChange()</i> method.
uint8 JobNotRunReasonCodes []	List or reasons by system for why job is waiting to run
string JobNotRunSystems []	List of system names corresponding to JobNotRunReasonCodes

IBMzOS_SysoutDataset

This class represents a z/OS sysout dataset. Some properties may require significant overhead, including I/O, to obtain their data. These properties are identified with the qualifier of "Expensive". To reduce system overhead, the provider will only return the values for these expensive properties if they are explicitly requested by name.

IBMzOS_JES2SysoutDataset

This class is a subclass of IBMzOS_SysoutDataset and contains those properties that are unique to a job that has run under JES2.

The following properties have been implemented for z/OS:

Property	Description
string Caption	A short description of the class
string Description	A description of the class
string ElementName	Name of given to this instance of the class
datetime InstallDate	Not supported for z/OS.
string Name [key]	JES2 Sysout Dataset name

Property	Description
uint16 OperationalStatus []	The current status of the JES2SysoutDataset: 0 = Unknown 2 = OK 6 = Error 9 = Stopping
string StatusDescriptions []	Not supported for z/OS.
string Status	Not supported for z/OS.
string CSCreationClassName [key]	The scoping ComputerSystem's CreationClassName.
string CSName [key]	The scoping ComputerSystem's Name.
string FSCreationClassName [key]	The scoping FileSystem's CreationClassName.
string FSName [key]	The scoping FileSystem's Name.
string CreationClassName [key]	Indicates the name of the class or the subclass used in the creation of an instance. When used with the other key properties of this class, this property allows all instances of this class and its subclasses to be uniquely identified.
uint64 FileSize	Not supported for z/OS.
datetime CreationDate	Not supported for z/OS.
datetime LastModified	Not supported for z/OS.
datetime LastAccessed	Not supported for z/OS.
boolean Readable	Boolean indicating that the File can be read.
boolean Writeable	Boolean indicating the File can be written.
boolean Executable	Boolean indicating the File is executable.
string CompressionMethod	Not supported for z/OS.
string EncryptionMethod	Not supported for z/OS.
uint64 InUseCount	Not supported for z/OS.
string ActiveMember	The JES member on which the sysout is active
string ActiveSysname	z/OS system on which the sysout is active
boolean Burst	Indicates whether 'Burst' mode is supported.
uint64 ByteCount	Byte count after blank truncation
string Class	The sysout class
datetime CreateTime	Date and time the data set became available This property is "Expensive".
string DataSetName	Sysout data set name This property is "Expensive".
uint32 DataSetNumber	Data set number This property is "Expensive".
string DDName	DDName for the data set creation This property is "Expensive".
string Destination	Sysout destination
string DeviceName	Name of the device on which sysout is active

Property	Description
string FCB	The name of the File Control Block (FCB) associated with this dataset.
boolean HeldByOperator	Sysout is held due to operator command
boolean HeldBySystem	Sysout is in a system hold
boolean HeldByUser	Sysout is currently held
string Identifier	This identifier is a value associated with this sysout that can be used in operator commands. The exact contents vary based on whether JES2 or JES3 owns the sysout and the release of JES processing the SSI request.
boolean IPAddrDest	Indicates that the 'Destination' property contains an Internet Protocol (IP) address.
string JobID	Job identified
string Jobname	Job name
uint16 MaxLogicalRecordLength	Maximum logical record length This property is "Expensive".
string ModifyModname	Modify=(modname)
string ModifyTrc	Modify=(,trc)
boolean NotSelectable	Not selectable
string OutDisp	Output disposition
string Owner	Sysout owner
uint32 PageCount	Page count
uint8 Priority	Sysout priority
string ProcessMode	Processing mode
string ProcName	Procname for the step creating this data set
uint32 RecordCount	Record count
string RecordFormat	Record format This property is "Expensive".
string Seclabel	Seclabel for sysput
uint32 SegmentID	Segment ID (zero if data set is not segmented)
boolean Spin	Spin data set
string StepName	Stepname for the step creating this data set This property is "Expensive".
string Subsystem	Subsystem name
string SystemHoldReason	Reason for system hold
string TPJobName	APPC transaction program jobname that created this data set
string TranslateTable []	Printer translate table
string UCS	UCS
string WriterName	External writer name
string JobToken	Job token
string OutputGroupName	Sysout group name

Property	Description
datetime OutputGroupElementCreateTime	JOE creation time
uint16 OGID1	JOE ID1

IBMzOS_JES3SysoutDataset

This class is a subclass of *IBMzOS_SysoutDataset* and contains those properties that are unique to a job that has run under JES3.

The properties of *IBMzOS_JES3SysoutDataset* are the same as for *IBMzOS_JES2SysoutDataset* (see “*IBMzOS_JES2SysoutDataset*” on page 120 with some exceptions:

IBMzOS_JES3SysoutDataset does not provide the following properties of *IBMzOS_JES2SysoutDataset*:

- *OutputGroupName*
- *OutputGroupElementCreateTime*
- *OGID1*

The following properties are only part of *IBMzOS_JES3SysoutDataset*

Property	Description
string HeldForTSO	Sysout is held for TSO
boolean HeldForExternalWriter	Sysout is held for external writer

IBMzOS_Subsystem

This class represents a z/OS Subsystem.

The following properties have been implemented for z/OS:

Property	Description
string Caption	A short description of the class
string Description	A description of the class
string ElementName	Name given to this instance of the class
datetime InstallDate	Not supported for z/OS.
string Name [key]	Subsystem name
uint16 OperationalStatus []	The current status of the JobSubSystem: 0 Unknown 2 OK 6 Error 9 Stopping
string StatusDescriptions []	Strings describing the various Operational Status values.
string Status	Not supported for z/OS.
uint16 EnabledState	Indicates the Enabled or Disabled state.
string OtherEnabledState	String describing the Enabled State value.
uint16 RequestedState	The last requested State.
uint16 EnabledDefault	Indicates the default value for Enabled State.

Property	Description
datetime TimeOfLastStateChange	Not supported for z/OS.
string SystemCreationClassName [key]	The scoping System's CreationClassName.
string SystemName [key]	The scoping System's Name.
string CreationClassName [key]	Indicates the name of the class or the subclass used in the creation of an instance. When used with the other key properties of this class, this property allows all instances of this class and its subclasses to be uniquely identified.
string PrimaryOwnerName	Not supported for z/OS.
string PrimaryOwnerContact	Not supported for z/OS.
string StartMode	StartMode is a string value indicating whether the Service is automatically started by a System, Operating System, etc. or only started upon request. This property is deprecated in lieu of the EnabledDefault property inherited from EnabledLogicalElement, which addresses the same semantics. The change to a uint16 data type was discussed when CIM V2.0 was defined. However, existing V1.0 implementations used the string property and did not want to modify their code. Therefore, StartMode was grandfathered into the Schema. Use of the Deprecated qualifier allows the maintenance of the existing property, but also permits an improved, clarified definition using EnabledDefault.
boolean Started	True if subsystem is active.
boolean Dynamic	True is subsystem is dynamic.
boolean DynamicCommands	True if subsystem responds to SETSSI command.
boolean Primary	Indicator for primary subsystem
uint8 Type	Subsystem type code: 1 Unknown 2 JES2 3 JES3 .. Unknown

IBMzOS_SubsystemJES2Jobs association

This class associates an *IBMzOS_Subsystem* with an *IBMzOS_JES2Job*.

IBMzOS_SubsystemJES3Jobs association

This class associates an *IBMzOS_Subsystem* with an *IBMzOS_JES3Job*.

IBMzOS_UsesJES2SysoutDatasets association

This class associates an *IBMzOS_JES2Job* with an *IBMzOS_JES2SysoutDataset*.

IBMzOS_UsesJES3SysoutDatasets association

This class associates an *IBMzOS_JES3Job* with an *IBMzOS_JES3SysoutDataset*.

IBMzOS_JobsManagementSettings

The *IBMzOS_JobsManagementSettings* class provides a mechanism by which users can influence the behavior of the *IBMzOS_JES2SysoutDataset*, *IBMzOS_JES3SysoutDataset*, *IBMzOS_JES2Jobs*, and *IBMzOS_JES3Jobs* providers.

The following properties have been implemented for z/OS:

Property	Description
string Caption	A short description of the class
string Description	A description of the class
string ElementName	Name given to this instance of the class
string InstanceID [Key]	Within the scope of the instantiating NameSpace, <i>InstanceID</i> opaquely and uniquely identifies an instance of this class. In order to ensure uniqueness within the NameSpace, the value of InstanceID SHOULD be constructed using the following algorithm: <OrgID>:<LocalID> where <OrgID> and <LocalID> are separated by a colon ':', and where <OrgID> MUST include a copyrighted, trademarked or otherwise unique name that is owned by the business entity creating/defining the InstanceID, or is a recognized global authority (This is similar to the <Schema Name>_<Class Name> structure of Schema class names.) In addition, to ensure uniqueness <OrgID> MUST NOT contain a colon (':'). When using this algorithm, the first colon to appear in InstanceID MUST appear between <OrgID> and <LocalID>. <LocalID> is chosen by the business entity and SHOULD not be re-used to identify different underlying (real-world) elements. If the above 'preferred' algorithm is not used, the defining entity MUST assure that the resultant InstanceID is not re-used across any InstanceIDs produced by this or other providers for this instance's NameSpace. For DMTF defined instances, the 'preferred' algorithm MUST be used with the <OrgID> set to 'CIM'.
uint32 MaxInstances	The maximum number of instances that can be returned.
uint32 MaxProperties	The maximum number of properties that can be returned

OS management Cluster classes

The classes described in this section are implemented by z/OS to instrument the z/OS "Systems Complex" (Sysplex) clustering facility.

For using these providers you need an extra security setup as described in "Appendix D. Cluster / CoupleDataset / JES2-JES3Jobs Provider Setup" on page 193.

IBMzOS_Sysplex

This class represents a zSeries Sysplex (System Complex).

The following properties have been implemented for z/OS:

Property	Description
string Caption	A short description of the class.
string Description	A description of the class.
string ElementName	Name given to this instance of the class.
datetime InstallDate	A datetime value indicating when the object was installed. A lack of a value does not indicate that the object is not installed.
string Name	Sysplex name
uint16 OperationalStatus []	The current status of the Sysplex, based on the states of the systems it is comprised of: 0 Unknown 2 OK 6 Error 9 Stopping
string StatusDescriptions []	Strings describing the various Operational Status values.
string Status	A string indicating the current status
uint16 EnabledState	Indicates the Enabled or Disabled state.
string OtherEnabledState	String describing the Enabled State value
uint16 RequestedState	The last requested State
uint16 EnabledDefault	Indicates the default value for Enabled State
datetime TimeOfLastStateChange	The date and time Enabled State was last changed.
uint64 IdentityToken [key]	The generated identity value for sysplex cluster. (Part 1 of 2)
string IdentityName [key]	The generated identity value for sysplex cluster. (Part 2 of 2)
uint32 Type	The type of sysplex cluster: 1 Local 2 Monoplex 3 Multisystem
uint32 State	State of the Sysplex, based on the states of the systems it is comprised of: 1 Okay 2 Unknown 3 Error 4 Stopping z/OS, will only report a state of "Okay" (1)
boolean SysplexConnectionFail	Corresponds to the CONNFAIL attribute in the SFM policy. Indicates whether or not action taken when connectivity failure occurs in the sysplex.

Methods of IBMzOS_Sysplex

Class IBMzOS_Sysplex provides the following methods:

Method	Description
uint32 SetSFMAAttributes()	Updates the SFM policy to set the SFM weights for each system specified in the input, SystemArray, and will set the Sysplex Connect Fail property value for the sysplex. Successful execution of this method will indicate that all the entries in the SystemArray were processed. If any of the system entries could not be processed the method will return an error.
uint32 SetConnectionFail()	Sets the ConnectionFail property value.
uint32 ResetConnectionFail()	Resets the ConnectionFail property value.
uint32 StartReallocate()	Analyzes all structures in the Sysplex and performs corrective actions on structures that are operating outside current CFRM policy parameters. Sysplex Process Completion Indication will be generated when asynchronous processing has completed.
uint32 StopReallocate()	Stops the reallocation of CF structures. Sysplex Process Completion Indication will be generated when asynchronous processing has completed.
uint32 ForceReallocate()	Forces an in process reallocation to be stopped. Sysplex Process Completion Indication will be generated when asynchronous processing has completed.

IBMzOS_SFMAAttributes

An array of embedded instances of this class is used as input parameter to method *SetSFMAAttributes()* (see “Methods of IBMzOS_Sysplex”). This class contains the following properties:

Property	Description
uint64 IdentityToken	Is the 'IdentityToken' of the SysplexNode whose SFM attributes are to be modified. The IdentityToken is a 64 bit unsigned integer that must be converted to a 20 character field, padded on the left with the character zero ('0'). An IdentityToken and IdentityName of '0' indicates that default values should be set for all SysplexNodes.
string IdentityName	Is the 'IdentityName' of the SysplexNode whose SFM attributes are to be modified. An IdentityToken and IdentityName of '0' indicates that default values should be set for all SysplexNodes.
boolean SFMweight	Indicates that the SFM_Weight property should be updated.

Property	Description
boolean SFMAction	Indicates that the SFM_Action (and possibly the SFM_Interval) property should be updated.
boolean SetMemoryStallTime	Indicates that the SFM stalled member action for the system should be updated.
boolean ResetMemoryStallTime	Indicates that the SFM stalled member action for the system should be cleared.
uint32 SFM_Weight	Is the new SFM weight value. The SFM weight is a 32 bit unsigned integer that must be converted to a 10 character field, padded on the left with the character zero ('0').
uint32 SFM_Action	Is the new SFM action value. Valid character values are: 1 Prompt operator 2 Isolate 3 System reset 4 Deactivate
uint32 SFM_Interval	Is the time in seconds corresponding to the SFM action. It is valid only when the action is being set to isolate (2), SystemReset (3), or Deactivate (4). The time is a 32 bit unsigned integer that must be converted to a 10 character field, padded on the left with the character zero ('0').
uint32 SFM_MemoryStallTime	Is the time in seconds that must pass before SFM takes action against a stalled member causing signal sympathy sickness.

Associations from IBMzOS_Sysplex

The IBMzOS_Sysplex class has the following associations:

- IBMzOS_CollectionOfCFs (see page 153)
- IBMzOS_CollectionOfSysplexNodes (see page 153)
- IBMzOS_UsesCouplingFunctions (see page 155)

Indications for IBMzOS_Sysplex

Class IBMzOS_Sysplex provides the following indications:

Indication	Description
IBMzOS_SysplexInstCreation	A 'lifecycle' indication that indicates that an instance of the IBMzOS_Sysplex class has been created. The Sysplex supports services that may report on cluster manageable resources. This event occurs when each system has IPLed into the Sysplex with a Cluster capable Sysplex Couple Dataset. This event occurs on each system when a Cluster capable dataset has been brought into use.
IBMzOS_SysplexInstModification	A 'lifecycle' indication that indicates that an instance of the IBMzOS_Sysplex class has been modified. The SysplexConnectionFail property has changed.

Indication	Description
IBMzOS_Sysplex_CFRM_CDS_Initialized	A 'process' indication that indicates that the process of reallocating the CF Structures has completed. CFRM Resources (Coupling Facility, CF Structure and CF Structure Connectors) has been defined to the Sysplex. The z/OS Cluster MR Services should be issued to obtain the CFRM resource instances in use by the Sysplex.
IBMzOS_Sysplex_ReallocateInitiated	A 'process' indication that indicates that the Start Reallocate CF Structures process has been initiated. The reallocate command may have been initiated by an operator command or through a CIM StartReallocate() method.
IBMzOS_Sysplex_ReallocateCompleted	A 'process' indication that indicates that the Start, Stop, or Force Reallocate CF Structures command has completed processing. The reallocate command may have been initiated by an operator command or through a CIM StartReallocate(), StopReallocate(), or ForceReallocate() methods.

IBMzOS_SysplexNode

This class represents a node in a zSeries Sysplex (System Complex). There is one node in a Sysplex for every z/OS system that comprises the Sysplex.

The following properties have been implemented for z/OS:

Property	Description
string Caption	A short description of the class.
string Description	A description of the class.
string ElementName	Name given to this instance of the class.
datetime InstallDate	A datetime value indicating when the object was installed. A lack of a value does not indicate that the object is not installed.
string Name	SysplexNode name which is the same as the Operating System's System Name
uint16 OperationalStatus []	The current status of the SysplexNode: 0 Unknown 2 OK 6 Error 9 Stopping
string StatusDescriptions []	Strings describing the various Operational Status values.
string Status	A string indicating the current status
uint16 EnabledState	Indicates the Enabled or Disabled state.
string OtherEnabledState	String describing the Enabled State value
uint16 RequestedState	The last requested State
uint16 EnabledDefault	Indicates the default value for Enabled State
datetime TimeOfLastStateChange	The date and time Enabled State was last changed.
uint64 IdentityToken [key]	The generated identity value for sysplex cluster. (Part 1 of 2)

Property	Description
string IdentityName [key]	The generated identity value for sysplex cluster. (Part 2 of 2)
uint32 State	State of node: 1 Okay 2 Unknown 3 Error 4 Stopping
uint32 SubState	SubState of node: 1 Normal 2 StatusUpdateMissing 3 InActive 4 IPLing Valid when State = Error. Not valid for all other system states.
uint32 SystemSFMWeight	Corresponds to System Weight attribute on SFM policy. Relative system weight used by clique algorithm following Sysplex connectivity failure
uint32 SystemFDIInterval	Corresponds to Failure Detection Interval attribute of SFM policy. Time interval during which missing status updates are tolerated. When failure interval is exceeded the SystemPartitionPolicy determines response
uint32 SystemSFMAction	Corresponds to Action attribute on SFM policy. One of four actions are settable in the SFM policy: 1 Prompt Operator 2 Isolate (isolate system using the CF fencing controls) 3 System Reset Partition 4 Deactivate Partition (deactivate the partition using the HMC controls)
uint32 SystemSFMInterval	When the System SFM Action is Automatic, System Reset, or Deactivate, this property will contain the time value in seconds corresponding to the SFM action.
uint32 SystemMemStallTime	For MEMSTALLTIME(stalltime), SFM will take action to resolve a sympathy sickness problem attributed to a stalled XCF group member if the problem persists for stalltime seconds.
uint32 SystemOpNotify	The length of time after a system is status update missing before SFM takes action. For PROMPT, the interval used is the XCF OPNOTIFY value.

Methods of IBMzOS_SysplexNode: The class IBMzOS_SysplexNode provides the following methods:

Method	Description
uint32 SetSystemFDIInterval()	Sets the SFM failure detection interval (FDI) for the system.

Associations from IBMzOS_SysplexNode: The IBMzOS_SysplexNode class has the following associations:

- IBMzOS_HostedCFStrConnector (see page 154)

- IBMzOS_UsesCFs (see page 154)

Indications for IBMzOS_SysplexNode: Class IBMzOS_SysplexNode provides the following indications:

Indication	Description
IBMzOS_SysplexNodeInstCreation	A 'lifecycle' indication that indicates that an instance of the IBMzOS_SysplexNode class has been created.
IBMzOS_SysplexNodeInstDeletion	A 'lifecycle' indication that indicates that an instance of the IBMzOS_SysplexNode class has been deleted.
IBMzOS_SysplexNodeInstModification	A 'lifecycle' indication that indicates that an instance of the IBMzOS_SysplexNode class has been modified.

IBMzOS_CouplingFacility

This class represents a zSeries Coupling Facility, which is the system that manages a Sysplex (System Complex).

The following properties have been implemented for z/OS:

Property	Description
string Caption	A short description of the class. Returns <i>'IBM z/OS JES2 Job'</i> .
string Description	A description of the class. Returns <i>'This is an IBMzOS_JES2Job'</i> .
string ElementName	Name given to this instance of the class (same as Name)
datetime InstallDate	A datetime value indicating when the object was installed. A lack of a value does not indicate that the object is not installed.
string Name	Coupling Facility Logical Name as assigned by CFRM policy definitions. Note: CF Name is not considered an immutable property of a Coupling Facility, since the name can be changed via a CFRM policy update. The physical CF information is the immutable identification information.
uint16 OperationalStatus []	The current status of the CF (summarized from more granular CF state information): 0 Unknown 2 OK 6 Error 9 Stopping
string StatusDescriptions []	Strings describing the various Operational Status values.
string Status	A string indicating the current status
uint16 EnabledState	Indicates the Enabled or Disabled state.
string OtherEnabledState	String describing the Enabled State value
uint16 RequestedState	The last requested State

Property	Description
uint16 EnabledDefault	Indicates the default value for Enabled State
datetime TimeOfLastStateChange	The date and time Enabled State was last changed.
uint64 IdentityToken [key]	The generated identity value for sysplex cluster. (Part 1 of 2)
string IdentityName [key]	The generated identity value for sysplex cluster. (Part 2 of 2)
string MachineType	Machine type of the server hosting the CF
string Manufacturer	Name of the manufacturer of the server hosting the CF
string ManufacturerPlant	The plant number where the machine was manufactured
string SerialNumber	A manufacturer assigned number to identify the server hosting the CF
uint8 LPARid	Platform-assigned ID of a logical partition in which the CF is running. Null if the Computer System is not virtualized
uint32 CFLevel	Facility operational (functionality) level
uint32 State	CF Operational State (summarized from more granular CF state information): 0 Okay 2 Unknown 6 Error 9 Stopping
string Information	Coupling Facility information
uint16 NumberOfProcessors	Total number of CF processors
uint16 CPUUtilization	Percent CF processor utilization
uint32 FreeSpace	Currently unused storage available in the CF (in number of 4KB blocks)
uint32 TotalSpace	Total storage available in the CF (in number of 4KB blocks)
uint32 FreeDumpSpace	Currently unused allocated dump storage available in the CF (in number of 4KB blocks)
uint32 TotalDumpSpace	Total allocated dump storage available in the CF (in number of 4KB blocks)
uint32 StorageIncrementSize	Storage increment. The number of 4K blocks in a single storage increment in this CF.
boolean Standalone	Coupling Facility Standalone indicator: 0 Not Standalone 1 Standalone
boolean Volatile	Indicator of whether this CF is volatile or nonvolatile (based on battery backup or standby power source)
boolean CPUType	Indicates whether all of the CF processors are shared, or whether at least one is dedicated: 0 All shared 1 Some are dedicated

Property	Description
boolean MaintenanceMode	Indicates whether the CF is currently in Maintenance mode: 0 Not in Maintenance mode 1 CF is in Maintenance mode
boolean RecoveryMgrSite	0 Recovery Manager is not active or the CF does not reside at the recovery site 1 Recover Manager is active and the CF resides at the recovery site.
string SiteName	Name of the SITE specified in the CFRM policy.
String CPCID	Coupling Facility's Central Processor Complex (CPC) ID.

Methods of IBMzOS_CouplingFacility

The class, IBMzOS_CouplingFacility, provides the following methods:

Method	Description
uint32 StartCFMaintenanceMode()	Sets the maintenance mode of the specified coupling facility to ON. When a CF is in maintenance mode, the CF is not eligible for CF structure allocation purposes and all structure allocation processes will modify their CF selection processing accordingly.
uint32 StopCFMaintenanceMode()	Sets the maintenance mode of the specified coupling facility to OFF. When a CF is no longer in maintenance mode, the CF is eligible for CF structure allocation purposes.

Associations from IBMzOS_CouplingFacility

The IBMzOS_CouplingFacility class has the following associations:

- IBMzOS_HostedCFStructures (see page 153)

Indications for IBMzOS_CouplingFacility

Class IBMzOS_CouplingFacility provides the following indications:

Indication	Description
IBMzOS_CouplingFacilityInstCreation	A 'lifecycle' indication that indicates that an instance of the IBMzOS_CouplingFacility class has been created.
IBMzOS_CouplingFacilityInstDeletion	A 'lifecycle' indication that indicates that an instance of the IBMzOS_CouplingFacility class has been deleted.
IBMzOS _CouplingFacilityInstModification	A 'lifecycle' indication that indicates that an instance of the IBMzOS_CouplingFacility class has been modified.

IBMzOS_CFStructure

This class represents a zSeries Coupling Facility Structure.

The following properties have been implemented for z/OS:

Property	Description
string Caption	A short description of the class.
string Description	A description of the class.
string ElementName	Name given to this instance of the class.
datetime InstallDate	A datetime value indicating when the object was installed. A lack of a value does not indicate that the object is not installed.
string Name	The Name of the structure as defined in the CFRM policy.
uint16 OperationalStatus []	The current status of the CF Structure: 0 Unknown 2 OK 6 Error 9 Stopping
string StatusDescriptions []	Strings describing the various Operational Status values.
string Status	A string indicating the current status
uint16 EnabledState	Indicates the Enabled or Disabled state.
string OtherEnabledState	String describing the Enabled State value
uint16 RequestedState	The last requested State
uint16 EnabledDefault	Indicates the default value for Enabled State
datetime TimeOfLastStateChange	The date and time Enabled State was last changed.
uint64 IdentityToken [key]	The generated identity value for sysplex cluster. (Part 1 of 2)
string IdentityName [key]	The generated identity value for sysplex cluster. (Part 2 of 2)
uint32 State	CF structure operational state: 1 Okay 2 Unknown 3 Error 4 Stopping
uint32 SubState	CF structure substate: 1 Normal (no exceptional conditions. 2 Temporarily degraded (alter in progress, structure dump serialization held). 3 Permanently degraded (allocated smaller than desired size, pending CFRM policy change). 4 Recovering (Valid only when the value of 'State' is 1 (Okay) or 4 (Stopping)).
uint8 Type	Structure type based on exploiter allocation requirements: 0x03 List 0xFF Lock 0x04 Cache 0xFE Serialized List

Property	Description
boolean AllowAlter	Indicator of whether this structure can be dynamically altered, based on current conditions. All active connectors to the structure specified ALLOWAUTO = YES on the IXLCONN connect request.
boolean AllowAuto	All active connectors to the structure specified ALLOWREBLD = YES on the IXLCONN connect request.
boolean AllowRebuild	All active connectors to the structure specified ALLOWDUPREBLD = YES on the IXLCONN connect request.
boolean AllowDupRebuild	All active connectors to the structure specified ALLOWALTER = YES on the IXLCONN connect request.
boolean IsDuplexed	Indicator of whether this structure actually is duplexed at this time. Only when this property indicates that the structure is duplexed, will the following properties be valid: • MaximumStructureSize2 • InitialStructureSize2 • MinimumStructureSize2 • OverFullThreshold2 • StructureVersion2 • CFName2 • CurrentStructureSize2
boolean PendPolicyChange	Indicates that there is a change pending in structure policy.
boolean Disposition	Defines whether the structure is persistent when there are no longer any defined connections (active or failed): 0 Keep 1 Delete
string CFName1	The name of the Coupling Facility in which this structure instance has been allocated. It is possible to have two structure instances due to rebuild-in-progress or duplexing. It is possible to have no structure instances when the structure is not currently allocated. When Duplexed this is the 'Old' instance of the structure.
string CFName2	The name of the Coupling Facility in which the 'New' structure instance has been allocated. Null if not allocated. This property is only valid when Duplexed.

Property	Description
string StructureVersion1	Structure version number for the currently allocated instance of the structure. It is possible to have two structure instances due to rebuild-in-progress or duplexing. It is possible to have no structure instances when the structure is not currently allocated. When Duplexed this is the 'Old' instance of the structure.
string StructureVersion2	Structure version number for the 'New' instance of the structure, when the structure is in the process of rebuilding or has been duplexed. This property is only valid when Duplexed.
uint32 MaximumStructureSize1	The maximum size to which this instance of the structure can be expanded, in units of 4KB. When Duplexed this is the 'Old' instance of the structure.
uint32 MaximumStructureSize2	The maximum size to which the 'New' instance of the structure can be expanded, in units of 4KB. This property is only valid when Duplexed.
uint32 InitialStructureSize1	The requested initial structure allocation size, in units of 4KB, for this instance of the structure. When Duplexed this is the 'Old' instance of the structure.
uint32 InitialStructureSize2	The requested initial structure allocation size, in units of 4KB, for the 'New' instance of the structure. This property is only valid when Duplexed.
uint32 MinimumStructureSize1	The minimum size at which this instance of the structure can be allocated or contracted to, in units of 4KB. When Duplexed this is the 'Old' instance of the structure.
uint32 MinimumStructureSize2	The minimum size at which the 'New' instance of the structure can be allocated or contracted to, in units of 4KB. This property is only valid when Duplexed.
uint32 CurrentStructureSize1	The allocated structure size, in units of 4 KB, for this instance of the structure. Not provided if the structure is not allocated. When Duplexed this is the 'Old' instance of the structure.
uint32 CurrentStructureSize2	The allocated structure size, in units of 4 KB, for the 'New' instance of the structure. Not provided if the structure is not allocated. This property is only valid when Duplexed.

Property	Description
uint32 SysMgdProcessLevel1	System Managed Process Level required by the instance of the structure to participate in a system-managed process. When Duplexed this is the 'Old' instance of the structure.
uint32 SysMgdProcessLevel2	System Managed Process Level required by the 'New' instance of the structure to participate in a system-managed process. This property is only valid when Duplexed.
uint32 ElementCount1	Element Count for the structure. List set element count for List structures. Data area element count for Cache Structures. Invalid for Lock Structures. When Duplexed this is the 'Old' instance of the structure.
uint32 ElementCount2	Element Count for the 'New' structure. List set element count for List structures. Data area element count for Cache Structures. Invalid for Lock Structures. This property is only valid when Duplexed.
uint32 EntryCount1	Entry Count for the structure. List set entry count for List and Lock Structures. Directory entry count for cache structures. When Duplexed this is the 'Old' instance of the structure.
uint32 EntryCount2	Entry Count for the 'New' structure. List set entry count for List and Lock Structures. Directory entry count for cache structures. This property is only valid when Duplexed.
uint32 EMCCount1	Event Monitor Controls count for List Structures. Invalid for Cache structures and Lock structures. When Duplexed this is the 'Old' instance of the structure.
uint32 EMCCount2	Event Monitor Controls count for 'New' List Structures. Invalid for Cache structures and Lock structures. This property is only valid when Duplexed.
uint32 LockCount1	Lock Entry Count. Valid for serialized List and Lock Structures. Invalid for Cache Structures and unserialized List structures. When Duplexed this is the 'Old' instance of the structure.
uint32 LockCount2	Lock Entry Count. Valid for 'New' serialized List and Lock Structures. Invalid for Cache Structures and unserialized List structures. This property is only valid when Duplexed.

Property	Description
string LogicalVersion1	Logical Version numner for the instance of the structure. When Duplexed this is the 'Old' instance of the structure.
string LogicalVersion2	Logical Version numner for the 'New' instance of the structure. This property is only valid when Duplexed.
string PreferenceList1 []	Structure Preference List for the instance of the structure. It is an array of up to 8 Coupling Facility names. When Duplexed this is the 'Old' instance of the structure.
string PreferenceList2 []	Structure Preference List for the instance of the structure. This is an array of up to 8 coupling facility names. This property is only valid when Duplexed.
string ExclusionList1 []	The Structure Exclusion List for the instance of the structure. This is an array of up to 8 coupling facility names. When Duplexed this is the 'Old' instance of the structure.
string ExclusionList2 []	Structure Exclusion List for the 'New' instance of the structure. This is an array of up to 8 coupling facility names. This property is only valid when Duplexed.
uint32 AccessTimeMax1	This instance of the structure was allocated with access time for IXLCONN ACESSTIME(MAXIMUM). When Duplexed this is the 'Old' instance of the structure.
uint16 AccessTimeMax2	The 'New' instance of the structure was allocated with access time for IXLCONN ACESSTIME(MAXIMUM). This property is only valid when Duplexed.
uint16 MaximumConnections1	The maximum number of connections allowed when the structure was allocated in the coupling facility. When Duplexed this is the 'Old' version of the structure.
uint16 MaximumConnections2	The maximum number of connections allowed when the 'New' instance of the structure was allocated in the coupling facility. This property is only valid when Duplexed.

Property	Description
uint8 FullThreshold1	Percentage value for the structure full monitoring threshold for the structure, as defined in CFRM policy. This threshold is set on-platform and is not currently settable through the resource model. When Duplexed this is the 'Old' version of the structure.
uint8 FullThreshold2	Percentage value for the structure full monitoring threshold for the 'New' version of the structure, as defined in CFRM policy. This threshold is set on-platform and is not currently settable through the resource model. This property is only valid when Duplexed.
uint8 RebuildPercent1	REBUILDPERCENT for the instance of the structure as specified in CFRM active policy. Not valid indicates not specified. When Duplexed this is the 'Old' version of the structure.
uint8 RebuildPercent2	REBUILDPERCENT for the 'New' instance of the structure as specified in CFRM active policy. Not valid indicates not specified. This property is only valid when Duplexed.
uint8 DuplexPolicy1	The effective DUPLEX option for the structure as specified in the CFRM active policy or defaulted. When Duplexed this is the 'Old' version of the structure.
uint8 DuplexPolicy2	The effective DUPLEX option for the 'New' structure as specified in the CFRM active policy or defaulted. This property is only valid when Duplexed.
boolean OverFullThreshold1	Indicator of whether or not the instance of the structure is currently in violation of its structure full monitoring threshold. When Duplexed this is the 'Old' instance of the structure.
boolean OverFullThreshold2	Indicator of whether or not the 'New' instance of the structure is currently in violation of its structure full monitoring threshold. This property is only valid when Duplexed.
boolean AllowAutoAlter1	ALLOWAUTOALT(YES) was specified in the CFRM active policy for the structure. When Duplexed this is the 'Old' instance of the structure.
boolean AllowAutoAlter2	ALLOWAUTOALT(YES) was specified in the CFRM active policy for the 'New' structure. This property is only valid when Duplexed.

Property	Description
boolean EnforceOrder1	ENFORCEORDER(YES) was specified in the CFRM active policy for the structure. When Duplexed this is the 'Old' instance of the structure.
boolean EnforceOrder2	ENFORCEORDER(YES) was specified in the CFRM active policy for the 'New' structure. This property is only valid when Duplexed.
boolean AllowReallocate1	ALLOWREALLOCATE(YES) was specified in the CFRM active policy for the structure. When Duplexed this is the 'Old' instance of the structure.
boolean AllowReallocate2	ALLOWREALLOCATE(YES) was specified in the CFRM active policy for the 'New' structure. This property is only valid when Duplexed.
boolean AccessTimeNoLimit1	The instance of the structure was allocated with IXLCONN ACESSTIME(NOLIMIT) When Duplexed this is the 'Old' instance of the structure.
boolean AccessTimeNoLimit2	The 'New' instance of the structure was allocated with IXLCONN ACESSTIME(NOLIMIT). This property is only valid when Duplexed.
uint32 MaxElementCount1	The maximum Element Count for the structure. List set element count for List structures. Data area element count for Cache Structures. Invalid for Lock Structures. When Duplexed this is the 'Old' instance of the structure.
uint32 MaxElementCount2	The maximum Element Count for the 'New' structure. List set element count for List structures. Data area element count for Cache Structures. Invalid for Lock Structures. This property is only valid when Duplexed.
uint32 MaxEntryCount1	The maximum Entry Count for the structure. List set entry count for List and Lock Structures. Directory entry count for cache structures. When Duplexed this is the 'Old' instance of the structure
uint32 MaxEntryCount2	The maximum Entry Count for the 'New' structure. List set entry count for List and Lock Structures. Directory entry count for cache structures. This property is only valid when Duplexed.
uint32 MaxEMCCount1	The maximum Event Monitor Controls count for List Structures. Invalid for Cache structures and Lock structures. When Duplexed this is the 'Old' instance of the structure

Property	Description
uint32 MaxEMCCount2	The maximum Event Monitor Controls count for 'New' List Structures. Invalid for Cache structures and Lock structures. This property is only valid when Duplexed.

Methods of IBMzOS_CFStructure

The class, IBMzOS_CFStructure, provides the following methods:

Method	Description
uint32 Force()	Will force the asynchronously deletion of a failed connector to a structure, following a failure. For some structures this is not permitted unless the structure itself is also forced (deallocated). This will only be performed against a structure in the Error state. The Force operation is performed asynchronously. The Return and Reason codes will indicate whether the operation was initiated successfully. A CFStructureConnector InstModification indication will be generated asynchronously when the force has been completed.

Associations from IBMzOS_CFStructure

The IBMzOS_CFStructure class has the following associations:

- IBMzOS_CFStrDependsOn (see page 152)

Indications for IBMzOS_CFStructure

Class IBMzOS_CFStructure provides the following indications:

Indication	Description
IBMzOS_CFStructureInstCreation	A 'lifecycle' indication that indicates that an instance of the IBMzOS_CFStructure class has been created.
IBMzOS_CFStructureInstDeletion	A 'lifecycle' indication that indicates that an instance of the IBMzOS_CFStructure class has been deleted.
IBMzOS_CFStructureInstModification	A 'lifecycle' indication that indicates that an instance of the IBMzOS_CFStructure class has been modified.

IBMzOS_CFStructureConnector

This class represents a zSeries Coupling Facility Structure Connector.

The following properties have been implemented for z/OS:

Property	Description
string Caption	A short description of the class.
string Description	A description of the class.
string ElementName	Name given to this instance of the class.

Property	Description
datetime InstallDate	A datetime value indicating when the object was installed. A lack of a value does not indicate that the object is not installed.
string Name	The Connector name.
uint16 OperationalStatus []	The current status of the CF connector: 0 Unknown 2 OK 6 Error 9 Stopping
string StatusDescriptions []	Strings describing the various Operational Status values.
string Status	A string indicating the current status
uint16 EnabledState	Indicates the Enabled or Disabled state.
string OtherEnabledState	String describing the Enabled State value
uint16 RequestedState	The last requested State
uint16 EnabledDefault	Indicates the default value for Enabled State
datetime TimeOfLastStateChange	The date and time Enabled State was last changed.
uint64 IdentityToken [key]	The generated identity value for sysplex cluster. (Part 1 of 2)
string IdentityName [key]	The generated identity value for sysplex cluster. (Part 2 of 2)
string ConnectorStructureName	The CFStructure name for the connection.
string ConnectorSystemName	OperatingSystem name for the system where the connector is running.
string ConnectorProcessName	Process name for the process in which the connector is running (for z/OS this is a jobpname).
uint64 ConnectorProcessID []	Unique process identification for the process in which the connector is running (for z/OS this is a ttoken).
uint32 State	Operational state of the CF connector: 0 Okay 2 Unknown 6 Error 9 Stopping
string ConnectorLevel	Connector-specified level information, or 0 if not provided by the connector.
boolean FailureIsolation	Indicator of whether or not the structure as currently allocated satisfies this connector's requirements for failure-isolation.
boolean Disposition	Indicator of the connector disposition. Defines whether the connection is persistent if the connection abnormally terminates. 0 Delete 1 Keep
boolean NonVolitalRequest	Indicator of whether the connector requested non-volatility.
uint8 ConnectorIdentifier	Connector Identifier.
string ConnectorVersion	Connector version number.

Property	Description
string ConnectorData	Connector data.
uint8 ConnectorInfoLevel	Connector Level of information.
uint8 ConnectorCFLevelRequired	Connector CF Level required.
boolean AllowRebuild	Indicates that the connector was connected with ALLOWREBUILD = YES
boolean AllowDupRebuild	Indicates that the connector was connected with ALLOWDUPBUILD = YES
boolean AllowAuto	Indicates that the connector was connected with ALLOWAUTO = YES
boolean AllowAlter	Indicates that the connector was connected with ALLOWALTER = YES
boolean Suspend	Indicates that the connector was connected with ALLOWALTER = YES, SUSPEND = YES
boolean AllowRatio	Indicates that the connector was connected with ALLOWALTER = YES, RATIO = YES
uint8 MinEntry	Indicates the value the connector specified for MINENTRY
uint8 MinElement	Indicates the value the connector specified for MINELEMENT
uint8 MinEMC	Indicates the value the connector specified for MINEMC

Methods of IBMzOS_CFStructureConnector

The class, IBMzOS_CFStructureConnector, provides the following methods:

Method	Description
uint32 StartRebuild()	Asynchronously rebuilds the structure into the same or a different CF than the one in which it is currently located. Only works if supported by exploiters. The Location parameter specifies the location where the new structure can be built. The LessConnAction parameter indicates whether the rebuild should be allowed to continue, in spite of a degradation in connectivity to the new structure. The rebuild will be performed asynchronously. The Return and Reason codes will indicate whether the operation was initiated successfully. A CFStructure InstModification indication will be generated asynchronously when the rebuild has completed. Coupling Facility Structure Operations should only be invoked from a single system in the sysplex.

Method	Description
uint32 StopRebuild()	Stops a Rebuild operation. A CFStructure InstModification indication will be generated asynchronously when the rebuild has been stopped. Coupling Facility Structure Operations should only be invoked from a single system in the sysplex.
uint32 StartDuplex()	Asynchronously establishes duplexing for the specified structure. Only works if supported by exploiters. The request to start duplexing will be performed asynchronously. The Return and Reason codes will indicate whether the operation was initiated successfully. A CFStructure InstModification indication will be generated asynchronously when the duplexing has started. Coupling Facility Structure Operations should only be invoked from a single system in the sysplex.
uint32 StopDuplex()	Stops duplexing. The required Keep parameter indicates which structure is to persist after duplexing has been stopped. The request to stop duplexing will be performed asynchronously. The Return and Reason codes will indicate whether the operation was initiated successfully. A CFStructure InstModification indication will be generated asynchronously when duplexing has been stopped. Coupling Facility Structure Operations should only be invoked from a single system in the sysplex.
uint32 Force()	Will asynchronously force the deallocation of a persistent structure. Force of a structure does not work if there are any active connectors to the structure, and may or may not work if there are failed connectors to the structure. The Force operation is performed asynchronously. The Return and Reason codes will indicate whether the operation was initiated successfully. A CFStructure InstModification indication will be generated asynchronously when the force has been completed.

Method	Description
uint32 ForceAll()	Will asynchronously force the deallocation of all failed persistent structures. The ForceAll operation is performed asynchronously. The Return and Reason codes will indicate whether the operation was initiated successfully. A CFStructure InstModification indication will be generated asynchronously when the force has been completed.

Indications for IBMzOS_CFStructureConnector

Class IBMzOS_CFStructureConnector provides the following indications:

Indication	Description
IBMzOS_CFStrConnectorInstCreation	A 'lifecycle' indication that indicates that an instance of the IBMzOS_CFStrConnector class has been created.
IBMzOS_CFStrConnectorInstDeletion	A 'lifecycle' indication that indicates that an instance of the IBMzOS_CFStrConnector class has been deleted.
IBMzOS_CFStrConnectorInstModification	A 'lifecycle' indication that indicates that an instance of the IBMzOS_CFStrConnector class has been modified.

IBMzOS_CouplingFunction

This class represents an abstraction of z/OS clustering capabilities. The clustering capabilities are referred to as coupling functions, each serving a unique purpose in a z/OS Systems Complex (Sysplex). Coupling functions are capabilities that are facilitated through the use of:

- Couple datasets, which serve as repositories.
- Coupling facilities, which are used by z/OS systems to cache data structures, serialization structures and provide signaling capabilities to z/OS systems participating in a Sysplex.
- Cross-System Coupling Facility (XCF) software, which is a component of z/OS that provides functions to support cooperation between authorized programs running within a Sysplex.

Coupling functions include such capabilities as basic Sysplex support and Coupling Facility Resource Manager (CFRM) support. There are other such coupling functions supported by z/OS, though not all of them may be externalized through CIM providers.

The following properties have been implemented for z/OS:

Property	Description
string Caption	A short description of the class.
string Description	A description of the class.

Property	Description
string ElementName	Name given to this instance of the class.
datetime InstallDate	A datetime value indicating when the object was installed. A lack of a value does not indicate that the object is not installed.
string Name [key]	Name of the coupling function
uint16 OperationalStatus []	The current status of the SysplexCoupleDataset: 0 = Unknown 2 = OK 6 = Error 9 = Stopping
string StatusDescriptions []	Strings describing the various Operational Status values.
string Status	A string indicating the current status
uint16 EnabledState	Indicates the Enabled or Disabled state.
string OtherEnabledState	String describing the Enabled State value.
uint16 RequestedState	The last requested State.
uint16 EnabledDefault	Indicates the default value for Enabled State.
datetime TimeOfLastStateChange	The date and time Enabled State was last changed.
uint32 Redundancy	This identifies the level of couple dataset redundancy currently active for the coupling function. 0 No couple datasets in use. The coupling function is not active. 1 Primary couple dataset in use. 2 Primary and alternate couple dataset are in use.
string ActivePolicyName	Specifies the name of the active policy for the coupling function. Instances of coupling functions such as SYSPLEX, which have no policy, will have a null string value.
uint64 TimeActivePolicyStarted	The local date and time that the active policy was started. Note: This property is in the local time of the operating system host servicing the request.
boolean IsActive	Identifies whether the coupling function is active. Coupling functions with no primary CDS are considered inactive. Coupling functions that support policies will be identified as active if they have a primary couple dataset in use, even if there is not active policy.

Methods of IBMzOS_CouplingFunction

The class, IBMzOS_CouplingFunction, provides the following methods:

Method	Description
uint32 StartPolicyMode()	This method activates (starts) the specified policy. The policy specified by the name parameter must be an administrative policy defined in the primary couple dataset currently in use by the coupling function.
uint32 StopPolicy()	This method inactivates the currently active policy. For Coupling Facilities (CFs) or structures that are actively being used, not all aspects of the policy may become inactive immediately. These changes will become pending until the resources in question are no longer being used by programs operating in the Sysplex.
uint32 DeletePolicy()	This method deletes the specified administrative policy. The policy specified by the name parameter must be an administrative policy defined in the primary couple dataset currently in use by the coupling function.
uint32 SwitchPrimary()	This method makes the current in-use alternate couple dataset the current primary couple dataset for the type represented by the coupling function instance. The current in-use primary couple dataset at the time this method is invoked, upon successful completion of the method, will no longer be recognized by XCF and the coupling function instance will be operating solely with a primary couple dataset. This method is similar to the z/OS operator command: SETXCF COUPLE,TYPE=__,PSWITCH
uint32 MakeAlternate()	This method makes the specified couple dataset the current in use alternate couple dataset for the type represented by the coupling function instance. The type of the specified couple dataset must be compatible with the coupling function instance for which the method was invoked. The specified couple dataset must be a newly formatted couple dataset, formatted specifically for use in the Sysplex in which the coupling function instance exists. The method will fail if the specified couple dataset is currently or was previously active in the Sysplex. The specified couple dataset may be one created using the Duplicate method or one created manually via the XCF couple dataset format utility (IXCL1DSU).

Method	Description
uint32 Duplicate()	This method duplicates the characteristics of the currently active primary couple dataset, for the type represented by the coupling function instance, to a new couple dataset. The name of the new couple dataset and the volume serial of the logical volume on which it will be allocated must be specified by the method invoker. The type of the couple dataset is determined by the coupling function instance.

Associations from IBMzOS_CouplingFunction

The IBMzOS_CouplingFunction class has the following associations:

- IBMzOS_UsesCouplingFunctions (see page 155)
- IBMzOS_UsesSysplexCoupleDatasets (see page 155)
- IBMzOS_UsesCFRMCoupleDatasets (see page 154)

IBMzOS_CoupleDataset

This class represents the methods and properties common to all specific types of z/OS couple datasets. Examples of z/OS couple datasets include z/OS System Complex (Sysplex) and Coupling Facility Resource Manager (CFRM) couple datasets.

The following properties have been implemented for z/OS:

Property	Description
string Caption	A short description of the class.
string Description	A description of the class.
string ElementName	Name given to this instance of the class.
datetime InstallDate	Not supported for z/OS.
string Name [key]	Name of Couple Dataset
uint16 OperationalStatus []	The current status of the SysplexCoupleDataset: 0 = Unknown 2 = OK 6 = Error 9 = Stopping
string StatusDescriptions []	Strings describing the various Operational Status values.
string Status	Not supported for z/OS.
string CSCreationClassName [key]	The scoping ComputerSystem's CreationClassName.
string CSName [key]	The scoping ComputerSystem's Name.
string FSCreationClassName [key]	The scoping FileSystem's CreationClassName.
string FSName [key]	The scoping FileSystem's name.
string CreationClassName [key]	CreationClassName indicates the name of the class or the subclass used in the creation of an instance. When used with the other key properties of this class, this property allows all instances of this class and its subclasses to be uniquely identified.

Property	Description
uint64 FileSize	Not supported for z/OS.
datetime CreationDate	Not supported for z/OS.
datetime LastModified	Not supported for z/OS.
datetime LastAccessed	Not supported for z/OS.
boolean Readable	Indicates that the File can be read.
boolean Writeable	Indicates that the File can be written.
boolean Executable	Indicates that the File is executable.
string CompressionMethod	Not supported for z/OS.
string EncryptionMethod	Not supported for z/OS.
uint64 InUseCount	Not supported for z/OS.
string SysplexName	This is the name of the z/OS Sysplex to which the couple dataset represented by an instance of this class belongs. Couple datasets are formatted for use in a particular Sysplex and cannot be used by a Sysplex other than the one for which they have been formatted.
string Volser	This is the volume serial of the logical volume on which the couple dataset is defined.
string DeviceNumber	This is the z/OS device number of the logical volume on which the couple dataset is defined. The device number is local to the z/OS system from which this instance was obtained. A logical volume may have different device numbers on different z/OS systems in the Sysplex, even though it is the same logical volume being shared by the different z/OS systems.
string NarrativeInfo	This property contains information used by the couple dataset owner to provide additional descriptive information about the couple dataset and its usage. This information includes formatting characteristics and any special functions or attributes that the couple dataset supports.
string Type	This property identifies the type of couple dataset the instance represents. Some examples of couple dataset types include CFRM and SYSPLEX. There are other types of couple datasets, although not all of them are externalized through CIM.
boolean IsPrimary	This property identifies whether the couple dataset represented by an instance is currently in use as the primary couple dataset for its type. A value of True indicates that this instance represents the couple dataset that is currently in use as the primary couple dataset of its type.

Property	Description
boolean IsAlternate	This property identifies whether the couple dataset represented by an instance is currently in use as the alternate couple dataset "for its type. A value of True indicates that this instance represents the couple dataset that is currently in use as the alternate couple dataset of its type.
uint32 MaximumNumberOfSystems	This property identifies the number of z/OS systems in the Sysplex that the couple dataset represented by this instance was formatted to support.
datetime FormatTime	This property identifies the local time that the couple dataset was formatted. Note: This property is in the local time of the operating system host servicing the request.
boolean IsSynchronized	This property applies only to instances representing couple datasets that are currently in use as the alternate couple dataset for their type. A value of True indicates that the couple dataset has been fully synchronized with the primary couple dataset of its type. A value of False indicates that the couple dataset is still in the process of synchronizing with the primary couple dataset of its type. An alternate couple dataset must be fully synchronized with the primary couple dataset of its type in order to provide failover capability in the event of an error affecting the primary couple dataset.
boolean ErrorState	This property identifies whether the couple dataset is in an error state. When True, the couple dataset has experienced a permanent error and is in the process of being removed from active use.
uint32 NumberOfStructures	This is the number of coupling facility (CF) structures that the CFRM couple dataset is formatted to support. It is the maximum number of structures that can be defined for use in a policy contained in this couple dataset.
uint32 NumberOfConnectors	Connectors are programs running under z/OS that establish a connection to a CF structure. This property identifies the number of connectors per structure that the couple dataset is formatted to support. It is the maximum number of concurrent connectors that can be supported for each structure defined in the couple dataset.
uint32 NumberOfCFs	This is the number of coupling facilities the couple dataset is formatted to support. It is the maximum number of CFs that can be defined for use in a CFRM policy contained in this couple dataset.
uint32 NumberOfPolicies	This is the number of administrative (inactive) policies that the couple dataset is formatted to support.

Methods of IBMzOS_CoupleDataset

The class, IBMzOS_CoupleDataset, provides the following methods:

Method	Description
uint32 SwitchPrimary()	This method switches the couple dataset represented by this instance as follows: If the instance represents a current in-use alternate couple dataset, it is switched to become the current primary couple dataset. If the alternate couple dataset is not fully synchronized or is in an error state, the method returns an error. If the instance represents a current in-use primary couple dataset, then it is switched out and the current in-use alternate couple dataset is switched to become the primary. If there is no current in-use alternate couple dataset or the in-use alternate couple dataset is not fully synchronized or in an error state, the method returns an error. This method functions like the z/OS operator command: SETXCF COUPLE,TYPE=___,PSWITCH

IBMzOS_SysplexCoupleDataset

This class represents the z/OS Systems Complex (Sysplex) couple datasets. A Sysplex couple dataset contains Sysplex-wide data about systems, groups, and members that use Cross-System Coupling Facility (XCF) services. All z/OS systems in a Sysplex must have connectivity to the Sysplex couple dataset.

A Sysplex couple dataset can be the primary, or optionally, the active alternate couple dataset supporting the Sysplex coupling function. Minimally, a Sysplex couple dataset must be in use as the active primary Sysplex couple dataset for the Sysplex function to be active.

IBMzOS_SysplexCoupleDataset is a subclass of IBMzOS_CoupleDataset, and as such it inherits all of its properties and methods.

IBMzOS_CFRMCoupleDataset

This class represents Coupling Facility Resource Manager (CFRM) couple datasets. A CFRM couple dataset contains CFRM policies, one of which can be active (started), defining how z/OS manages coupling facility resources.

A CFRM couple dataset can be the active primary, or optionally, the active alternate couple dataset supporting the CFRM coupling function. Minimally, a CFRM couple dataset must be in use as the active primary CFRM couple dataset for CFRM coupling function to be active.

IBMzOS_CFRMCoupleDataset is a subclass of IBMzOS_CoupleDataset, and as such it inherits all of its properties and methods.

IBMzOS_CFRMPolicy

This class represents administrative (inactive) Coupling Facility Resource Manager (CFRM) policies. CFRM policies are used to control Coupling Facility (CF) and CF

structure resources available to a z/OS Sysplex (Systems Complex). There can be only one active CFRM policy and some number of administrative (inactive) policies.

The following properties have been implemented for z/OS:

Property	Description
string Caption	A short description of the class.
string Description	A description of the class.
string ElementName	Name given to this instance of the class.
datetime InstallDate	A datetime value indicating when the object was installed. A lack of a value does not indicate that the object is not installed.
string Name [key]	Name of CFRM Policy
uint16 OperationalStatus []	The current status of the SysplexCoupleDataset: 0 = Unknown 2 = OK 6 = Error 9 = Stopping
string StatusDescriptions []	Strings describing the various Operational Status values.
string Status	A string indicating the current status
string EnabledState	Indicates the Enabled or Disabled state.
string OtherEnabledState	String describing the Enabled State value.
uint16 RequestedState	The last requested State.
uint16 EnabledDefault	Indicates the default value for Enabled State.
datetime TimeOfLastStateChange	The date and time Enabled State was last changed.
boolean Active	Indicates whether this policy is an active policy.
string PolicyText	This property contains the CFRM policy statements that define the Coupling Facilities (CFs) and CF structures that are eligible to be used by programs operating in the Sysplex when this policy is activated (started) via the <i>StartPolicy()</i> method. The CFRM policy, as defined by its <i>PolicyText</i> , governs many aspects of the use of CFs and CF structures by the Sysplex. For example, it governs CF structure placement, fixing, recovery and availability considerations.

IBMzOS_CFStrDependsOn association

This class associates an *IBMzOS_CFStructure* with an *IBMzOS_CFStructureConnector*.

Indications for IBMzOS_CFStrDependsOn

Class IBMzOS_CFStrDependsOn provides the following indications:

Indication	Description
IBMzOS_CFStrDependsOnInstCreation	A 'lifecycle' indication that indicates that an instance of the IBMzOS_CFStrDependsOn association class has been created.
IBMzOS_CFStrDependsOnInstDeletion	A 'lifecycle' indication that indicates that an instance of the IBMzOS_CFStrDependsOn association class has been deleted.

IBMzOS_CollectionOfCFs association

This class associates an *IBMzOS_Sysplex* with an *IBMzOS_CouplingFacility*.

Indications for IBMzOS_CollectionOfCFs

Class IBMzOS_CollectionOfCFs provides the following indications:

Indication	Description
IBMzOS_CollectionOfCFsInstCreation	A 'lifecycle' indication that indicates that an instance of the IBMzOS_CollectionOfCFs association class has been created.
IBMzOS_CollectionOfCFsInstDeletion	A 'lifecycle' indication that indicates that an instance of the IBMzOS_CollectionOfCFs association class has been deleted.

IBMzOS_CollectionOfSysplexNodes association

This class associates an *IBMzOS_Sysplex* with an *IBMzOS_SysplexNode*.

Indications for IBMzOS_CollectionOfSysplexNodes

Class IBMzOS_CollectionOfSysplexNodes provides the following indications:

Indication	Description
IBMzOS_CollectionOfSysplexNodesInstCreation	A 'lifecycle' indication that indicates that an instance of the IBMzOS_CollectionOfSysplexNodes association class has been created.
IBMzOS_CollectionOfSysplexNodesInstDeletion	A 'lifecycle' indication that indicates that an instance of the IBMzOS_CollectionOfSysplexNodes association class has been deleted.

IBMzOS_HostedCFStructures association

This class associates an *IBMzOS_CFStructure* with an *IBMzOS_CouplingFacility*.

Indications for IBMzOS_HostedCFStructure

Class IBMzOS_HostedCFStructure provides the following indications:

Indication	Description
IBMzOS_HostedCFStructureInstCreation	A 'lifecycle' indication that indicates that an instance of the IBMzOS_HostedCFStructure association class has been created.
IBMzOS_HostedCFStructureInstDeletion	A 'lifecycle' indication that indicates that an instance of the IBMzOS_HostedCFStructure association class has been deleted.

IBMzOS_HostedCFStructureConnectors association

This class associates an *IBMzOS_SysplexNode* with an *IBMzOS_CFStructureConnector*.

Indications for IBMzOS_HostedCFStructureConnectors

Class IBMzOS_HostedCFStructureConnectors provides the following indications:

Indication	Description
IBMzOS_HostedCFStrConnectorInstCreation	A 'lifecycle' indication that indicates that an instance of the IBMzOS_HostedCFStructureConnectors association class has been created.
IBMzOS_HostedCFStrConnectorInstDeletion	A 'lifecycle' indication that indicates that an instance of the IBMzOS_HostedCFStructureConnectors association class has been deleted.

IBMzOS_UsesCF association

This class associates an *IBMzOS_SysplexNode* with an *IBMzOS_CouplingFacility*.

Indications for IBMzOS_UsesCF

Class IBMzOS_UsesCF provides the following indications:

Indication	Description
IBMzOS_UsesCFsInstCreation	A 'lifecycle' indication that indicates that an instance of the IBMzOS_UsesCFs association class has been created.
IBMzOS_UsesCFsInstDeletion	A 'lifecycle' indication that indicates that an instance of the IBMzOS_UsesCFs association class has been deleted.

IBMzOS_UsesCFRMCoupleDatasets association

This class associates an instance of *IBMzOS_CouplingFunction* with instances of *IBMzOS_CFRMCoupleDataset* classes.

IBMzOS_UsesCFRMPolicies association

This class associates an instance of the *IBMzOS_CFRMCoupleDataset* class with instances of the *IBMzOS_CFRMPolicy* classes.

IBMzOS_UsesCouplingFunctions association

This class associates an instance of the *IBMzOS_Sysplex* class with instances of the *IBMzOS_CouplingFunction* classes.

IBMzOS_UsesSysplexCoupleDatasets association

This class associates an instance of the *IBMzOS_CouplingFunction* class with instances of the *IBMzOS_SysplexCoupleDataset* classes.

Chapter 6. Messages and logs

This chapter contains the following sections with information on which logs and messages the z/OS CIM server provides to users:

- “Logging”
- “CIM server messages”

Logging

The CIM server sends log messages to the z/OS system console as well as to the z/OS Communications Server's syslog daemon. For information how to configure logging for the CIM server please refer to “Configure logging” on page 45.

When using the system logger (syslog) daemon, TRACE level log messages can be captured that will not be sent to the z/OS system console. However, since a log level of TRACE will cause a significant amount of messages to be logged, it is not recommended to turn it on by default.

CIM server messages

Messages are written into the appropriate logs and also displayed at the z/OS console.

All messages while using the CIM server are part of the underlying OpenPegasus code. This section documents only those messages that are specific while using the CIM server on z/OS, together with explanation, system action, (system) programmer and user response.

All other OpenPegasus messages are wrapped by one of the following generic z/OS messages.

CFZ00001I	for INFORMATION log messages
CFZ00002W	for WARNING log messages
CFZ00004E	for SEVERE and FATAL log messages

z/OS-specific messages

| **CEZ10000E** Unable to obtain a passticket for
| GPMSEVER. RACF permissions
| probably missing

| **Explanation:** The Monitoring providers were unable to
| obtain a valid passticket for the application GPMSEVER
| (RMF Distributed Data Server).

| **System action:** The CIM request is not processed.

| **System programmer response:** Make sure the CIM
| client user has UPDATE access to the generation profile
| IRRPTAUTH.GPMSEVER.*.

| **User response:** Contact your system programmer.

| **CFZ02202W** Property value is not valid: *name=value*

| **Explanation:** The value that was specified for the
| configuration property is not valid. See "CIM server
| configuration properties" on page 42 for the correct
| values of configuration properties.

| **System action:** None.

| **System programmer response:** None.

| **User response:** Re-enter the command specifying a
| correct value for the configuration property.

| **CFZ02207W** The configuration property *name* is not
| dynamic.

| **Explanation:** The configuration property *name* cannot
| be changed dynamically for a running CIM server.
| Instead the change has to be made as a planned value
| to become effective after a CIM server restart. See
| "cimconfig" on page 55 or "MODIFY console
| command" on page 72 for details on how to change
| planned values.

| **System action:** None.

| **System programmer response:** None.

| **User response:** Change the planned configuration
| value and restart the CIM server.

CFZ02300I Configuration property *conf-property* is
not supported. Setting ignored.

Explanation: The mentioned configuration property is
no longer supported.

System action: The CIM server ignores this setting
and continues.

System programmer response: Please remove the
mentioned configuration property from the planned
configuration of the CIM server's
/etc/wbem/cimserver_planned.conf.

User response: None.

| **CFZ03029E** Unsupported *UserContext* value: "*value*".

| **Explanation:** A provider module was registered with a
| *UserContext* value of *value*, but that value is not
| supported by this version of the CIM server. Valid
| values are 2 for "Requestor" and 3 for "Designated
| User".

| **System action:** The provider module is not registered.

| **System programmer response:** Check the provider
| registration MOF and replace the invalid *UserContext*
| value with a value that is valid on z/OS.

| **User response:** Contact your system programmer.

| **CFZ03030E** Missing *DesignatedUserContext* property
| in PG_ProviderModule instance.

| **Explanation:** A provider module was registered with a
| *UserContext* value of 3 ("Designated User"). The user ID
| of the designated user has to be specified in
| *DesignatedUserContext*, but no value was found (see
| "PG_ProviderModule" on page 188).

| **System action:** The provider module is not registered.

| **System programmer response:** Check the provider
| registration MOF and add a valid user ID for the
| *DesignatedUserContext* property to all provider modules
| that are registered with a *UserContext* value of 3.

| **User response:** Contact your system programmer.

| **CFZ06201W** Command not recognized by CIM
server.

| **Explanation:** The command that was entered in the
| system console is not supported by the CIM server.

| **System action:** None.

| **System programmer response:** None.

| **User response:** None.

CFZ06202I STOP command received from z/OS
console, initiating shutdown.

Explanation: The CIM server received the STOP
command from the console.

System action: The CIM server is shutting down.

System programmer response: None.

User response: None.

CFZ06203E CIM server Console command thread
cannot be created: *error-text (errno*
error-number, reason code X'reason-code').

Explanation: The CIM server cannot start the thread

handling commands issued at the console. For a description of error *error-text* with error number *error-number* and the last four digits of the reason code *X'reason-code'*, see *z/OS UNIX System Services Messages and Codes*, or enter the reason code in the BPXMTEXT TSO command.

System action: None.

System programmer response: The CIM server cannot be stopped using the console command. To stop the CIM server, purge the address space or use a privileged UNIX user ID to issue the command `cimserver -s` from the UNIX System Services command prompt.

User response: Contact your system programmer.

CFZ06204E Console Communication Service failed:
error-text (errno error-number, reason code X'reason-code').

Explanation: The CIM server is connected to the system console by using the Console Communication Service. The CIM server received the unrecoverable error *error-text*. For a description of error *error-text* with *errno error-number* and the last four digits of the reason code *X'reason-code'*, see *z/OS UNIX System Services Messages and Codes*, or enter the reason code in the BPXMTEXT TSO command.

System action: CIM server shuts down.

System programmer response: *Errno error-number* and the last four digits of the reason code *X'reason-code'* point out the reason for the error. Check the console for more messages indicating the problem.

User response: None.

CFZ06205E CIM MODIFY command rejected due to syntax error.

Explanation: A MODIFY command was entered for the CIM server that could not be recognized due to invalid syntax.

System action: None.

System programmer response: None.

User response: Enter the command with the correct syntax.

CFZ06206I Syntax is: MODIFY
CFZCIM,APPL=CONFIG,
name=valueæ,PLANNED]

Explanation: This messages describes the expected format for CIM server MODIFY command.

System action: None.

System programmer response: None.

User response: None.

CFZ06207E Failed to update configuration value.

Explanation: The CIM server failed to update a configuration value that was entered through the system console.

System action: None.

System programmer response: Look for other messages indicating the problem.

User response: Look for other messages indicating the problem.

CFZ06208I Updated current value for name to value.

Explanation: A configuration value for a running CIM server has immediately been updated. The changed value will stay in effect as long as the CIM server is running. After a restart the value is reset to either the default or to the planned configuration value.

System action: The change requested by the MODIFY command is now in effect.

System programmer response: None.

User response: None.

CFZ06209I Updated planned value for name to value.

Explanation: A configuration value has been updated for the planned configuration of the CIM server. It will become active after the CIM server is restarted. This change is persistent until the planned value is changed again.

System action: The change requested by the MODIFY command becomes effective after the next CIM server restart.

System programmer response: None.

User response: None.

CFZ06210I This change will become effective after CIM server restart.

Explanation: The change requested by the MODIFY command will not be in effect until the CIM server is restarted.

System action: None.

System programmer response: None.

User response: None.

CFZ06211E MODIFY command failed: message

Explanation: A configuration update requested through the MODIFY command failed. The detailed cause is indicated by *message*.

System action: None.

| **System programmer response:** None.

| **User response:** None.

| **CFZ06212E** *name is not a valid configuration property.*

| **Explanation:** The configuration property *name* is not recognized by the CIM server as a valid configuration property.

| **System action:** None.

| **System programmer response:** None.

| **User response:** Use the correct name for the configuration property and re-enter the command.,

| **CFZ07805E** *Failed to bind socket on port port-number: error-text (error code error-code, reason code 0xreason-code).*

| **Explanation:** Before listening on network port *port-number* the CIM server failed to bind the socket with *error-code* and *0xreason-code*. It therefore will not be able to communicate over this network port. Probably the port is already in use by another program or has been reserved by the TCP/IP configuration.

System action: The CIM server does not start.

| **System programmer response:** Error code *error-code* and the last four digits of the reason code *0xreason-code* point out the reason for the error.

For a description of error *error-text* with error code *error-code* and the last four digits of the reason code *0xreason-code*, see *z/OS UNIX System Services Messages and Codes*, or enter the reason code in the BPXMTEXT TSO command.

| **User response:** None.

| **CFZ07806E** *Failed to set permission on local domain socket socket: error-text (error code error-code, reason code 0xreason-code).*

| **Explanation:** The CIM server is not able to set the permission on socket file *socket* for local communication.

| **System action:** The CIM server does not start.

| **System programmer response:** Error code *error-code* and the last four digits of the reason code *0xreason-code* point out the reason for the error. For a description of error *error-text* with error code *error-code* and the last four digits of the reason code *0xreason-code*, see *z/OS UNIX System Services Messages and Codes*, or enter the reason code in the BPXMTEXT TSO command.

| **User response:** None.

| **CFZ07807E** *Failed to listen on socket socket-number: error-text (error code error-code, reason code 0xreason-code).*

| **Explanation:** The CIM server failed to listen on socket *socket-number*. It therefore will not be able to communicate over this network port. Probably the port is already in use by another program or has been reserved by the TCP/IP configuration.

| **System action:** The CIM server does not start.

| **System programmer response:** Error code *error-code* and the last four digits of the reason code *0xreason-code* point out the reason for the error. For a description of error *error-text* with error code *error-code* and the last four digits of the reason code *0xreason-code*, see *z/OS UNIX System Services Messages and Codes*, or enter the reason code in the BPXMTEXT TSO command.

| **User response:** None.

CFZ09100I *TCP/IP temporary unavailable.*

Explanation: The TCP/IP stack used by the CIM server is not available.

System action: The CIM server is waiting for a restart of the TCP/IP stack. The CIM server will be not able to handle any commands and requests until the restart of the TCP/IP stack has completed. Currently processed requests are terminated.

System programmer response: Restart the TCP/IP stack the CIM server was using. If this stack is no longer used, restart the CIM server.

User response: None.

CFZ10024I *Unable to start the CIM server. CIM server is already running.*

Explanation: The CIM server detects that another instance of the CIM server is already running. There can be only one running CIM server.

System action: None.

System programmer response: Do not start the CIM server again. If you want to start a new CIM server on the system, use the stop command at the system console (/p cfzcm) or look for the CIM server running in the UNIX System Services (/d omvs,a=all) and cancel the process (/c cfzcm).

User response: None.

CFZ10025I *The CIM server is listening on HTTP port port-number.*

Explanation: The CIM server is starting up and will listen on port *port-number* for incoming requests from clients. For information about how to configure the CIM server's HTTP connections see "CIM server configuration" on page 41.

System action: None.

System programmer response: None.

User response: None.

CFZ10026I The CIM server is listening on HTTPS port *port-number*.

Explanation: The CIM server is starting up and will listen on port *port-number* for incoming requests from clients using SSL encryption. Please note that special TCP/IP configuration settings are required for enabling the CIM server to support SSL encryption for HTTPS. For information about how to configure the CIM server's HTTPS connections see "Configuring the CIM server HTTPS connection using AT-TLS" on page 27.

System action: None.

System programmer response: None.

User response: None.

CFZ10028I The CIM server is listening on the local connection socket.

Explanation: The CIM server is starting up and will listen for incoming requests from clients. For information about how to configure the CIM server's HTTP connections see "CIM server configuration" on page 41.

System action: None.

System programmer response: None.

User response: None.

CFZ10030I Started CIM server version *version*.

Explanation: The CIM server is now started and accepts CIM client requests.

System action: None.

System programmer response: None.

User response: None.

CFZ10031I CIM server - stopped.

Explanation: The CIM server is now stopped. CIM client requests are no longer accepted.

System action: None.

System programmer response: None.

User response: None.

CFZ10034E CIM server repository contains files with wrong tags. Unable to set file tags. Stopping CIM server startup.

Explanation: The CIM server repository contains files tagged with the wrong CCSID. The CIM server tried to

set the right CCSID (ISO8859-1) tag on this file, but was not successful.

System action: The CIM server stops.

System programmer response: Look for previously issued messages (CFZ10035E or equivalent LE messages) about access violations for path */var/wbem*. Grant the denied access right to the user ID running the CIM server. Restart the CIM server.

User response: None.

CFZ10035E Failed to change file tag for *file-name*. Error (*error-number*): *error-message*.

Explanation: The CIM server is not able to change the file tag for the file *file-name*. For the reason, see the system error number *error-number* and the system error message *error-message*.

System action: The CIM server stops.

System programmer response: Correct the reason for failing to change the file tag. The reason is indicated by the system error number *error-number* and the system error message *error-message*.

User response: None.

CFZ10036W CIM server repaired file tags for *number* repository files.

Explanation: The CIM server was able to restore the correct CCSID (ISO8859-1) file tag for a number of *number* repository files.

System action: None.

System programmer response: Repository file tags were missing or wrong. Please revise procedures handling files located in */var/wbem* to preserve file tags. If file tags are preserved, this message will not appear again.

User response: None.

CFZ10037E Failed to open repository directory *repository-directory*: *error-text* (error code *error-code*, reason code *0xreason-code*).

Explanation: The CIM server is not able to open the directory *repository-directory* containing the repository.

System action: The CIM server does not start.

System programmer response: Error code *error-code* and the last four digits of the reason code *0xreason-code* point out the reason for the error. For a description of error *error-text* with error code *error-code* and the last four digits of the reason code *0xreason-code*, see *z/OS UNIX System Services Messages and Codes*, or enter the reason code in the BPXMTEXT TSO command.

User response: None.

**CFZ12500E Not loading dynamic load library
library-name due to missing program
control flag.**

Explanation: The CIM server runs on a system with Enhanced Security and thus does not load dynamic libraries which are not audited by a system programmer.

System action: The system does not load the named dynamic library.

System programmer response: Set the program control flag on the dynamic library using the UNIX System Services command `extattr +p <filename>`.

User response: Contact a system programmer to audit the dynamic library and set the program control flag.

**CFZ12501E Security profile CIMSERV in CLASS
WBEM must be defined. Ending CIM
server.**

Explanation: The CIM server detected an incomplete security setup.

System action: The CIM server does not start.

System programmer response: Complete the security setup by defining the profile CIMSERV in class WBEM. Refer to "Customizing the security for the CIM server" on page 21 for further details.

User response: Contact your system programmer.

**CFZ12502E CIM server user ID requires either
READ access to BPX.SERVER or must
be UID 0. Ending CIM server.**

Explanation: The user ID that runs the CIM server must have READ access to the security profile BPX.SERVER, or, if BPX.SERVER is not defined on your system, must be a privileged user.

System action: The CIM server stops.

System programmer response: Permit the user ID to run the CIM server by either giving it READ access to profile BPX.SERVER, or, if not running in an Enhanced Security environment, set the UID to 0.

User response: Contact your system programmer.

**CFZ12503E CIM server address space dirty due to
loading from a not program controlled
load library. Ending CIM server.**

Explanation: The CIM server loaded a dynamic library that is not program controlled. Either the security setup is not complete or a dynamic library has been changed without a system programmer's audit.

System action: The CIM server stops.

Programmer response: Check all dynamic libraries for their program control flag and ensure that no library

changed. Make sure that the Language Environment libraries SCEERUN and SCEERUN2 are program controlled.

User response: Contact your system programmer.

**CFZ12504E CIM server does not have appropriate
privileges to check SAF security
environment. Ending CIM server.**

Explanation: The user ID that runs the CIM server must have READ access to the security profile BPX.SERVER, or, if BPX.SERVER is not defined on your system, must be privileged.

System action: The CIM server stops.

System programmer response: Permit the user ID to run the CIM server by either giving it READ access to profile BPX.SERVER, or, if not running in an Enhanced Security environment, set the UID to 0.

User response: Contact your system programmer.

**CFZ12505E CIM server user ID requires either
READ access to BPX.SERVER or must
be UID 0. Ending CIM server.**

Explanation: The user ID that runs the CIM server must have READ access to the security profile BPX.SERVER, or, if BPX.SERVER is not defined on your system, must be privileged.

System action: The CIM server stops.

System programmer response: Permit the user ID to run the CIM server by either giving it READ access to profile BPX.SERVER, or, if not running in an Enhanced Security environment, set the UID to 0.

User response: Contact your system programmer.

**CFZ12506E CIM server address space dirty due
loading from a not program controlled
load library. Ending CIM server.**

Explanation: The CIM server has loaded a dynamic library that is not program controlled. Either the security setup is not complete or a dynamic library has been changed without a system programmer's audit.

System action: The CIM server stops.

Programmer response: Check all dynamic libraries for their program control flag and ensure that no library has changed. Make sure the Language Environment libraries SCEERUN and SCEERUN2 are program controlled.

User response: Contact your system programmer.

CFZ12507W CIM server does not have surrogate for client user ID *user-ID*.

Explanation: A request sent from the user ID could not be processed. The CIM server does not have access to act as surrogate for the requesting user ID.

System action: The user request is ignored and an error message is sent to the client.

System programmer response: To permit the CIM server user ID to act as a surrogate for the client user, grant the user ID running the CIM server READ access to the RACF profile BPX.SRV.*user-ID* as described in “Switching identity (surrogate)” on page 27.

User response: Contact your system programmer.

CFZ12508W Failure *error-number* deleting thread security.

Explanation: The CIM server was not able to delete the thread level security built for a specific request.

System action: None.

System programmer response: None.

User response: None.

CFZ12509E The CIM server user ID requires either READ access to BPX.SERVER or must be UID 0. Stopping CIM server startup.

Explanation: The user ID that starts the CIM server must have READ access to the security profile BPX.SERVER, or, if BPX.SERVER is not defined on your system, must be a privileged user.

System action: The CIM server does not start.

System programmer response: Permit the user ID to run the CIM server by either giving it READ access to profile BPX.SERVER, or, if not running in an Enhanced Security environment, set the UID to 0.

User response: Contact your system programmer.

CFZ12510E CIM server address space dirty due to loading from a not program controlled load library. Stopping CIM server startup.

Explanation: The CIM server loaded a dynamic library that is not program controlled during startup. Probably the security setup is not complete or a dynamic library has been changed without a system programmer’s audit.

System action: The CIM server does not start.

Programmer response: Check all dynamic libraries for their program control flag and ensure that no library changed. Make sure the Language Environment libraries SCEERUN and SCEERUN2 are program controlled.

User response: Contact your system programmer.

CFZ12511E CIM server does not have appropriate privileges to check SAF security environment. Stopping CIM server startup.

Explanation: The user ID that starts the CIM server must have READ access to the security profile BPX.SERVER, or, if BPX.SERVER is not defined on your system, must be a privileged user.

System action: The CIM server does not start.

System programmer response: Permit the user ID to run the CIM server by either giving it READ access to profile BPX.SERVER, or, if not running in an Enhanced Security environment, set the UID to 0.

User response: Contact your system programmer.

CFZ12512E Security profile CIMSERV in CLASS WBEM must be defined. Stopping CIM server startup.

Explanation: The CIM server detected an incomplete security setup on startup.

System action: The CIM server does not start.

System programmer response: To complete the security setup, define the profile CIMSERV in class WBEM. Refer to “Customizing the security for the CIM server” on page 21 for further details.

User response: Contact your system programmer.

CFZ12513E The CIM server user ID requires CONTROL access to security profile CIMSERV in CLASS WBEM. Stopping CIM server startup.

Explanation: The CIM server user ID requires CONTROL access to security profile CIMSERV in CLASS WBEM.

System action: The CIM server does not start.

System programmer response: To permit the CIM server user ID to perform administrative CIM tasks, give it CONTROL permission to profile CIMSERV in class WBEM. Refer to “Customizing the security for the CIM server” on page 21 for further details.

User response: Contact your system programmer.

CFZ12514E Security profile *profile-name* in CLASS WBEM must be defined.

Explanation: A provider defined a security profile at registration that is not defined for RACF class WBEM.

System action: None.

System programmer response: Create the RACF profile in class WBEM and permit users who should have access to the provider. Verify if the security profile

is defined for RACF and make sure that the class WBEM has been refreshed. Verify if the provider really should be registered with the mentioned security profile and if it should be checked.

User response: Contact your system programmer.

CFZ12515W User *user-ID* not authorized to perform intrinsic CIM operation *operation* against provider *provider-name*. *access-type* access to *profile-name* in CLASS WBEM required.

Explanation: User *user-ID* is not authorized to perform CIM operation *operation* involving the provider *provider-name*. The user needs *access-type* access to SAF security profile *profile-name* that is defined in class WBEM.

System action: The request is not processed and an "Access Denied" notification is sent to the client.

System programmer response: Verify if the user should be permitted to perform the current request. If so, grant the user *access-type* access to the profile *profile-name*.

User response: None. Access has been denied to a user with insufficient authority.

CFZ12516E CIM server does not have appropriate privileges to check SAF security environment. Ending CIM server.

Explanation: The user ID that runs the CIM server must have READ access to the security profile BPX.SERVER, or, if BPX.SERVER is not defined on your system, must be a privileged user.

System action: The CIM server stops.

System programmer response: Permit the user ID to run the CIM server by either giving it READ access to profile BPX.SERVER, or, if not running in an Enhanced Security environment, set the UID to 0.

User response: None.

CFZ12517E Missing IdentityContainer (no username) in request.

Explanation: The security component of the CIM server detected an invalid operation context that does not contain a username.

System action: The request is not processed and an "Access Denied" notification is sent to the client.

System programmer response: None.

User response: None.

CFZ12519E An unexpected error occurs: *error-text* (error number *error-number*, reason code *0xreason-code*). Stopping CIM server startup.

Explanation: During startup, the CIM server received the unrecoverable error *error-text*. For a description of error *error-text* with error number *error-number* and the last four digits of the reason code *0xreason-code*, see *z/OS UNIX System Services Messages and Codes*, or enter the reason code in the BPXMTEXT TSO command.

System action: The CIM server does not start.

System programmer response: Error number *error-number* and the last four digits of the reason code *0xreason-code* point out the reason for the error. Check the console for more messages indicating the problem.

User response: Contact your system programmer.

CFZ12520E CIM server did not set *Must Stay Clean*. Stopping Provider Agent startup.

Explanation: The Provider Agent determined that the *Must Stay Clean* flag was not set. The Provider Agent startup is not processed by the CIM server.

System action: The Provider Agent does not start.

System programmer response: Ensure that the Provider Agent can only be started by the CIM server.

User response: Contact your system programmer.

CFZ12521E An unexpected error occurs: *error-text* (error number *error-number*, reason code *X'reason-code'*). Stopping Provider Agent startup.

Explanation: During startup, the Provider Agent received the unrecoverable error *error-text*. For a description of error *error-text* with error number *error-number* and the last four digits of the reason code *X'reason-code'*, see *z/OS UNIX System Services Messages and Codes*, or enter the reason code in the BPXMTEXT TSO command.

System action: The Provider Agent does not start.

System programmer response: Error number *error-number* and the last four digits of the reason code *X'reason-code'* point out the reason for the error. Check the console for more messages indicating the problem.

User response: Contact your system programmer.

CFZ12523E CIM Runtime Environment user ID requires either READ access to BPX.SERVER or has to be UID 0. Stopping Provider Agent startup.

Explanation: The user ID that runs the Provider Agent must have READ access to the security profile BPX.SERVER, or, if BPX.SERVER is not defined on your

system, must be a privileged user.

System action: The Provider Agent does not start.

System programmer response: Permit the user ID to run the CIM server by either giving it READ access to profile BPX.SERVER, or, if not running in an Enhanced Security environment, set the UID to 0.

User response: Contact your system programmer.

CFZ12524E Provider Agent address space dirty due to loading from a not program controlled load library. Stopping Provider Agent startup.

Explanation: The Provider Agent has loaded a dynamic library that is not program controlled. Either the security setup is not complete or a dynamic library has been changed without a system programmer's audit.

System action: The Provider Agent does not start.

System programmer response: Check all dynamic libraries for their program control flag and ensure that no library has changed. For details on program control look at *z/OS UNIX System Services Planning* and *z/OS Security Server RACF Security Administrator's Guide*.

User response: Contact your system programmer.

CFZ12525E CIM Runtime Environment does not have appropriate privileges to check SAF security environment. Stopping Provider Agent startup.

Explanation: The user ID that runs the Provider Agent must have READ access to the security profile BPX.SERVER, or, if BPX.SERVER is not defined on your system, must be a privileged user.

System action: The Provider Agent does not start.

System programmer response: Permit the user ID to run the Provider Agent by either giving it READ access to profile BPX.SERVER, or, if not running in an Enhanced Security environment, set the UID to 0.

User response: Contact your system programmer.

CFZ12526E Unsupported UserContext value: "value".

Explanation: A provider module was registered with a *UserContext* value of *value*, but that value is not supported by this version of the CIM server. Valid values are 2 ("Requestor") and 3 ("Designated User").

System action: The addressed provider module is not correctly registered. The request fails and an error is sent back to the requestor.

System programmer response: Identify the failing provider module, remove the provider using the cimprovider utility (see "cimprovider" on page 57) and

re-register the provider with a correct provider registration MOF.

User response: Contact your system programmer.

CFZ12527E Missing DesignatedUserContext property in PG_ProviderModule instance.

Explanation: A provider module was registered with a *UserContext* value of 3 ("Designated User"). The user ID of the designated user has to be specified in *DesignatedUserContext*, but no value was found (see "PG_ProviderModule" on page 188).

System action: The request that is directed against the provider module in error will fail and an error is sent back to the requestor.

System programmer response: Identify the failing provider module, remove the provider using the cimprovider utility (see "cimprovider" on page 57) and re-register the provider with a correct provider registration MOF.

User response: Contact your system programmer.

CFZ12528I Cannot switch to designated user user-ID. User is unknown to the security product, or has no OMVS segment.

Explanation: The CIM server failed to switch the security context to *user-ID* for a provider configured with a designated user context. The user *user-ID* defined for the provider's security context is not defined to the system or does not have an OMVS segment.

System action: The request fails and an authorization error is sent back to the requestor/client.

System programmer response: Check if the user *user-ID* is the right user ID to run with or check for the existence of the user *user-ID* within your security product with the appropriate OMVS segment. If the problem persists you may want to remove the failing provider using the cimprovider utility and re-register the provider with the correct designated user defined in the provider registration MOF.

User response: None.

CFZ12529E An unexpected error occurred when switching to user user-ID: error-text (error code error-code , reason code 0xreason-code).

Explanation: The CIM server failed to switch to *user-ID* for the designated user context of a provider.

System action: The request fails and an authorization error is sent back to the requestor/client.

System programmer response: Error code *error-code* and the last four digits of the reason code *0xreason-code* point out the reason for the error. For a description of

| error *error-text* with error code *error-code* and the last
| four digits of the reason code *0xreason-code*, see *z/OS*
| *UNIX System Services Messages and Codes*, or enter the
| reason code in the BPXMTEXT TSO command.

| **User response:** None.

CFZ12532I CIM server successfully registered to
ARM using element name
CFZ_SRV_ *system-name*.

Explanation: The CIM server successfully registered to
the Automatic Restart Manager.

System action: None.

System programmer response: None.

User response: None.

CFZ12533I CIM server failed to register with ARM
using element name
CFZ_SRV_ *system-name*: return code
X'*error-number*', reason code X'*reason-code*'.

Explanation: The CIM server failed to register with
the Automatic Restart Manager using the element name
CFZ_SRV_ *system-name*.

System action: None.

System programmer response: If you do not want to
use the Automatic Restart Manager, you can ignore this
message. If you want to use ARM, use X'*error-number*'
and X'*reason-code*' to look up the return and reason
codes for the IXCARM macro in the *z/OS MVS*
Programming: Sysplex Services Reference for the reason to
fail to register with ARM.

User response: None.

CFZ12534W Authorization failed: User ID *user-ID*
does not have CONTROL permission to
profile CIMSERV CL(WBEM).

Explanation: The user ID requesting an administrative
task, for example, cimconfig or cimprovider, does not
have the required permission.

System action: The request is not processed and an
"Access Denied" notification is sent to the client.

System programmer response: Permit the user to
perform administrative CIM tasks by giving him
CONTROL permission to profile CIMSERV in class
WBEM.

User response: Contact your system programmer.

CFZ12535W Authorization failed: User ID *user-ID*
misses UPDATE permission to profile
CIMSERV CL(WBEM) to execute a
writing CIM operation.

Explanation: A client with the named user ID has sent

a CIM request for a CIM write operation (SetProperty,
InvokeMethod, CreateInstance, ModifyInstance,
DeleteInstance) to the CIM server without having the
appropriate access rights.

System action: The request is not processed and an
"Access Denied" notification is sent to the client.

System programmer response: None.

User response: If you need to perform CIM write
operations, ask your system programmer to grant you
at least UPDATE access to profile CIMSERV CL(WBEM).

CFZ12540E ATTLS reset the connection due to
handshake failure. Connection closed.

Explanation: AT-TLS reset the connection with the
client due to a handshake failure.

System action: The connection is closed.

System programmer response: This message
documents an unsuccessful connect to AT-TLS. If this
prevents a connection from a client to the server, switch
on tracing at the AT-TLS policy to find the reason for
this closure.

User response: Contact your system programmer.

CFZ12541E An unexpected error occurs: *error-text*
(error number *error-number*, reason code
X'*reason-code*'). Connection closed.

Explanation: While querying the AT-TLS connection
using `ioctl()`, the CIM server received an unknown
error. For a description of error *error-text* with error
number *error-number* and the last four digits of the
reason code X'*reason-code*', see *z/OS UNIX System*
Services Messages and Codes, or enter the reason code in
the BPXMTEXT TSO command.

System action: The connection is closed.

System programmer response: Contact IBM support.

User response: Contact your system programmer.

CFZ12542E ATTLS policy is not active for the CIM
server HTTPS port. Communication not
secured. Connection closed.

Explanation: The CIM server is configured to use
HTTPS by defining the configuration property
`enableHttpsConnection`, but the AT-TLS policy is not
configured correctly for the CIM server.

System action: The connection is closed.

System programmer response: Please refer to
"Customizing the security for the CIM server" on page
21 for information on how to configure AT-TLS for the
CIM server.

User response: Contact your system programmer.

CFZ12543E ATTLS policy not valid for CIM server.
Set *ApplicationControlled* to OFF.
Connection closed.

Explanation: The value of the property *ApplicationControlled* defined in the AT-TLS policy for the CIM server is ON. Hence, the CIM server is only aware of AT-TLS but does not control it.

System action: The connection is closed.

System programmer response: Change the property *ApplicationControlled* to OFF in the AT-TLS policy defined for the CIM server. Refer to “Customizing the security for the CIM server” on page 21 for information on how to configure AT-TLS for the CIM server.

User response: None.

CFZ12544E ATTLS policy specifies the wrong *HandshakeRole* for the CIM server HTTPS port. Communication not secured. Connection closed.

Explanation: The property *HandshakeRole* defined in the inbound AT-TLS policy for the CIM server is not configured correctly.

System action: The connection is closed.

System programmer response: Change the property *HandshakeRole* to *ServerWithClientAuth* or to the server at the inbound AT-TLS policy defined for the CIM server. Refer to “Customizing the security for the CIM server” on page 21 for information on how to configure AT-TLS for the CIM server.

User response: None.

CFZ12545E Automatic repository upgrade failed at step *step-number*. Stopping CIM server startup.

Explanation: The CIM server failed to automatically migrate the old repository in */var/wbem* to the new schema level. No actual migration action was run, because the basic setup is not correct.

System action: The CIM server does not start.

System programmer response: To find out the reason for this error, check the previously issued message. Correct the basic setup and restart the CIM server.

User response: None.

CFZ12546E Automatic repository upgrade failed at step *step-number*. Recovery completed successfully. Stopping CIM server startup.

Explanation: The CIM server failed to automatically migrate the old repository in */var/wbem* to the new schema level. Migration started and ran to a certain point. Though they failed, the accomplished migration

actions were successfully rolled back.

System action: The CIM server does not start.

System programmer response: To find out the reason for this error, check the previously issued message. Correct the setup problem and restart the CIM server.

User response: Contact your system programmer.

CFZ12547F Automatic repository upgrade failed at step *step-number*. Recovery failed, manual intervention required. Stopping CIM server startup.

Explanation: The CIM server failed to automatically migrate the old repository in */var/wbem* to the new schema level. Migration started and ran into a critical break. The attempt to roll back the taken actions failed. Manual user intervention is required to roll back taken migration actions.

System action: The CIM server does not start.

System programmer response: To find out the reason for this error, check the previously issued message. Rollback the taken migration actions as described in that message. Fix the setup problem and restart the CIM server.

User response: None.

CFZ12548E Failed to initiate command: *command* with error: *error-number*.

Explanation: The CIM server failed to automatically migrate the old repository in */var/wbem* to the new schema level. Processing of the named command failed with error *error-number*.

System action: The CIM server will roll back already taken migration actions. The CIM server does not start.

System programmer response: Investigate why the named command cannot perform successfully. Fix the system setup and restart the CIM server.

You can find further details in *STDERR* and *STDOUT* of the job output.

User response: Contact your system programmer.

CFZ12549E Command *command* failed with status *status-code*.

Explanation: The CIM server failed to automatically migrate the old repository in */var/wbem* to the new schema level. The processing of the named command failed with status *status-code*.

System action: The CIM server will roll back the already taken migration actions. CIM server does not start.

System programmer response: Investigate why the named command cannot perform successfully. Fix the

system setup and restart the CIM server.

Further details can be found in STDERR and STDOUT of the job output.

User response: Contact your system programmer.

CFZ12550E Failed to rename directory
source-directory-name to
target-directory-name with error:
error-number.

Explanation: The CIM server failed to automatically migrate the old repository in */var/wbem* to the new schema level. Renaming of source directory to target directory failed.

System action: The CIM server will roll back already taken migration actions. CIM server does not start.

System programmer response: Investigate the reason of the renaming failure. Possible reasons are missing file access rights, a full file system or missing access right to run a program in an extra UNIX System Services address space.

User response: Contact your system programmer.

CFZ12551E Failed to create repository status files
with: *error-text.*

Explanation: The CIM server failed to write the repository status file while automatically migrating the old repository in */var/wbem* to the new schema level. The migration is nearly complete, but writing the repository status file failed. The repository status file serves to avoid repeated attempts to migrate the repository.

System action: A message is logged to the system console. The CIM server startup continues.

System programmer response: Either fix the reason for the failed write of the repository status file and stop and restart the CIM server, or copy the file supplied in */usr/lpp/wbem/* to */var/wbem*.

User response: Contact your system programmer.

CFZ12552I Starting automatic repository upgrade.

Explanation: The CIM server will start to migrate the old repository to the new schema level.

System action: The CIM server starts to migrate the repository.

System programmer response: None.

User response: None.

CFZ12554E Error during automatic repository upgrade. No reference repository found at *directory-name*.

Explanation: The CIM server could not locate the new repository at location *directory-name*. No actual migration action was run, because basic setup is not correct.

System action: The CIM server does not start.

System programmer response: Check the SMP/E installation. Directory and files should have been copied to the named location in the SMP/E APPLY step.

User response: Contact your system programmer.

CFZ12555E Rename of previous repository to *directory-name* failed.

Explanation: The CIM server failed to automatically migrate the old repository in */var/wbem* to the new schema level. Even though migration successfully created the new repository, renaming the old repository for backup failed.

System action: The CIM server will remove the new repository to roll back the taken migration actions. The CIM server does not start.

System programmer response: Investigate why the CIM server was unable to rename the directory */var/wbem/repository* to the directory *directory-name*. Probable causes are insufficient disk space or missing access rights.

User response: None.

CFZ12556E Rename of new repository to *directory-name* failed.

Explanation: The CIM server failed to automatically migrate the old repository in */var/wbem* to the new schema level. Even though migration successfully created the new repository and backed up the old repository, renaming the new repository to */var/wbem/repository* failed.

System action: The CIM server tries to roll back the taken migration actions and also removes the new repository and renames the backed up version to */var/wbem/repository*.

System programmer response: Investigate why the CIM server was unable to rename the directory. Probable reasons are insufficient disk space or missing access rights. If rollback actions fail (indicated by message CFZ12547E), manually remove the directory named */var/wbem/repository_new* and rename the latest backed up repository version to */var/wbem/repository*.

User response: None.

CFZ12557E Failure during automatic repository upgrade. Trying to recover.

Explanation: The CIM server failed to automatically migrate the old repository in */var/wbem* to the new schema level.

System action: The CIM server will try to roll back the taken migration actions.

System programmer response: Check former and further messages for details and possible required actions.

User response: None.

CFZ12558E Failed to remove incomplete new repository at *directory-name*.

Explanation: The CIM server failed to automatically migrate the old repository in */var/wbem* to the new schema level. Removing the new, migrated repository failed.

System action: The CIM server does not start.

System programmer response: Remove the directory */var/wbem/repository_new* and its subfolders and files. Check the system log for earlier messages for details on the actual migration step that failed. Fix the situation and restart the CIM server. The most common reason for this problem is insufficient disk space at */var/wbem*.

User response: None.

CFZ12559F Failed to restore previous repository on recovery. Manual rename of *source-directory-name* back to *target-directory-name* required!

Explanation: The CIM server tried to roll back the migration actions. Renaming the backed up copy of the old repository to target directory name failed.

System action: The CIM server does not start.

System programmer response: Rename the source directory to the target directory name. Investigate the reason for the failure of the automatic repository migration by checking the system log for former error messages. Fix the system setup and restart the CIM server.

User response: Contact your system programmer.

CFZ12560E Failed to create repository status file *directory-name*. Manual intervention required!

Explanation: The CIM server failed to write the repository status file while automatically migrating the old repository in */var/wbem* to the new schema level. The migration is nearly complete, but writing the repository status file failed. The repository status file serves to avoid repeated tries to migrate the repository.

System action: A message is logged to the system console. The CIM server startup continues.

System programmer response: Either fix the reason for the failed write of the repository status file and stop and restart the CIM server, or copy the file supplied in */usr/lpp/wbem/* to */var/wbem*.

User response: Contact your system programmer.

CFZ12561E Repository in directory *directory-name* is backlevel. Run migration job for repository upgrade.

Explanation: The CIM server failed to automatically migrate the old repository in */var/wbem* to the new schema level. No actual migration action was run, because the basic setup is not correct. The old repository found at *directory-name* is not a z/OS 1.8 level repository.

System action: The CIM server does not start.

System programmer response: Use migration job CFZRCUST to migrate the repository.

User response: Contact your system programmer.

CFZ12562I Previous repository was renamed to *directory-name* for backup and can be removed.

Explanation: The CIM server successfully migrated the old repository to the new schema level. A backup copy of the old repository is stored at *directory-name*. The copy should be backed up and then can be deleted to free up disk space.

System action: The CIM server startup continues.

System programmer response: You may want to backup the old repository, and delete the copy on hard disk.

User response: None.

CFZ12563I Automatic repository upgrade completed successfully.

Explanation: The CIM server successfully migrated the old repository to the new schema level.

System action: The CIM server startup continues.

System programmer response: None.

User response: None.

CFZ12564W Failed to obtain information on file *system path-name*. Error: *error-text*.

Explanation: The CIM server failed to determine information about the file system at *path-name*. The cause of the failure was error *error-text*.

System action: Automatic repository upgrade continues.

System programmer response: None.

User response: None.

CFZ12565W File system at *path-name* is smaller than the recommended 102400 KB (100MB).

Explanation: The file system available at *path-name* should be at least 100MB large or be able to extend to that size. The CIM server might run out of space when automatically upgrading the repository.

System action: Automatic repository upgrade continues.

System programmer response: Make sure that there is enough space for data to be stored in the file system at *path-name*. Recommended is a system specific data set with at least 100MB space mounted at */var/wbem*.

User response: None.

CFZ12566W Less free space than 61440 KB (60MB) available on file system *path-name*.

Explanation: The CIM server detected less than 60MB space available in the file system *path-name*. The CIM server might run out of space when automatically upgrading the repository.

System action: Automatic repository upgrade continues.

System programmer response: Make sure that there is enough space for data to be stored in the file system at *path-name*. Recommended is a system specific data set with at least 60MB space mounted at */var/wbem*.

User response: None.

CFZ12568E ATTLS is not active for TCP-IP stack the CIM server is using for HTTPS connections. Communication not secured. Connection closed.

Explanation: The CIM server is configured to use HTTPS by defining the configuration property *enableHttpsConnection*, but the Communication Server Policy Agent was not enabled on the stack the CIM server is using when AT-TLS policy mapping was performed for the connection.

System action: The connection is closed.

System programmer response: Ensure that Communication Server Policy Agent is configured for the TCP/IP stack the CIM server is listening. Please refer to “Customizing the security for the CIM server” on page 21 for information on how to configure AT-TLS for the CIM server.

User response: Contact your system programmer.

CFZ12569E There is no ATTLS policy found for the CIM server HTTPS connections. Communication not secured. Connection closed.

Explanation: The CIM server is configured to use HTTPS by defining the configuration property *enableHttpsConnection*, but the Communication Server Policy Agent did not find an AT-TLS policy for the CIM server when AT-TLS policy mapping was performed for the connection.

System action: The connection is closed.

System programmer response: Ensure that a Communication Server Policy Agent policy is defined for CIM Server. Please refer to “Customizing the security for the CIM server” on page 21 for information on how to configure AT-TLS for the CIM server.

User response: Contact your system programmer.

CFZ12570I Created directory */var/wbem*.

Explanation: CIM server successfully created the directory */var/wbem*.

System action: None.

System programmer response: None.

User response: None.

CFZ12571E Failed to create directory */var/wbem* with error: *error-message*. Stopping CIM server startup.

Explanation: CIM server failed to create the directory */var/wbem* with error *error-message*.

System action: The CIM server does not start.

System programmer response: Check the system setup for a system-specific data set mounted at path */var/wbem* with 100Mb space. Fix the problem and restart the CIM server.

User response: Contact your system programmer.

CFZ12572W Failed to read repository status file: *error-message*.

Explanation: CIM server failed to read information from the repository status file at */var/wbem*.

System action: CIM server startup proceeds and the repository is automatically migrated to the latest level available from */usr/lpp/wbem*.

System programmer response: Check the error condition described by *error-message* and fix the indicated problem in the system setup.

User response: Contact your system programmer.

| **CFZ12573W** Failed to read *file-name*: *error-message*.

| **Explanation:** CIM server failed to read information from the file *file-name* which is used to set up environment variables for the CFZCIM started task procedure.

| **System action:** CIM server startup proceeds.

| **System programmer response:** Make sure that the file *file-name* is readable for the CIM Server user if you wish to run the CIM server from the started task procedure CFZCIM.

| **User response:** Contact your system programmer.

| **CFZ12574W** File *file-name* contains quotes which should be removed. Removing quotes and stopping CIM server startup. Restart the CIM server.

| **Explanation:** CIM server found quote characters in file *file-name*. Quotes can cause environment variable setup problems.

| **System action:** CIM server tries to remove all quotes. The CIM server does not start.

| **System programmer response:** None.

| **User response:** Restart the CIM server.

| **CFZ12575E** Failed to open *file-name* for write with error: *error-message*.

| **Explanation:** CIM server failed to open *file-name* for writing. The reason is named in *error-message*. CIM server found quote characters in the environment variable setup file for the started task procedure. CIM server tried to open the environment variable setup file to remove all quotes.

| **System action:** CIM server does not start.

| **System programmer response:** Remove all quotes in file *file-name* manually or check the error condition described by *error-message* and fix the indicated problem in the system setup. Restart the CIM server.

| **User response:** Contact your system programmer.

| **CFZ12576F** Failed to write all data to *file-name* file.

| **Explanation:** CIM server failed to write to *file-name* for the reason named in *error-message*. CIM server found quote characters in the environment variable setup file for the started task procedure. CIM server tried to write the environment variable setup file with all quote removed, but the file was written partially.

| **System action:** CIM server does not start.

| **System programmer response:** Check the error condition described by *error-message* and fix the indicated problem in the system setup. Create a new environment variables setup file *file-name* using the

| default shipped in */usr/lpp/wbem/install*. Restart the CIM server.

| **User response:** Contact your system programmer.

| **CFZ12577I** Successfully removed all quotes from *file-name*.

| **Explanation:** CIM server removed all quote characters from file *file-name*. Quotes can cause environment variable setup problems for the started task procedure. To avoid issues caused by partially setup environment variables the CIM server is stopped and needs to be restarted.

| **System action:** CIM server does not start.

| **System programmer response:** Restart the CIM server.

| **User response:** Restart the CIM server.

| **CFZ12578W** Directory */var/wbem* does not exist. CIM server will create it.

| **Explanation:** On CIM server startup the automated migration procedure detected that path */var/wbem* does not exist.

| **System action:** CIM server creates the directory */var/wbem*.

| **System programmer response:** None.

| **User response:** None.

| **CFZ13006W** Request user ID *user-ID* doesn't have READ permission to profile CIMSERV CL(WBEM).

| **Explanation:** The user ID requesting a CIM operation using a remote connection is not permitted to use the CIM server.

| **System action:** The request is not processed and an "Access Denied" notification is sent to the client.

| **System programmer response:** Permit the user to perform CIM requests by giving the user ID READ access to profile CIMSERV CL(WBEM).

| **User response:** Contact your system programmer to permit your user ID to perform CIM requests. Repeat your request.

| **CFZ13007W** Request user ID *user-ID* doesn't have READ permission to profile CIMSERV CL(WBEM).

| **Explanation:** The user ID requesting a CIM operation using a remote connection is not permitted to use the CIM server.

| **System action:** The CIM request is not processed and an "Access Denied" notification is sent to the client.

| **System programmer response:** Permit the user to

| perform CIM requests by giving the user ID READ
| access to profile CIMSERV CL(WBEM).

| **User response:** Contact your system programmer to
| permit your user ID to perform CIM requests. Repeat
| your request.

CFZ13607E CIM server cannot execute
Out-Of-Process Provider Agent: *error-text*
(error number *error-number*, reason code
X'reason-code').

Explanation: The CIM server failed to process the
Out-Of-Process Provider Agent caused by the problem
error-text. For further details, see the description of
error number *error-number* and the last four digits of
the reason code *X'reason-code'* in *z/OS UNIX System*
Services Messages and Codes.

System action: None.

System programmer response: Stop the CIM server.
Error number *error-number* and the last four digits of
the reason code *X'reason-code'* point out the reason for
the error. Check the console for more messages
indicating the problem.

User response: Contact your system programmer.

| **CFZ17200W** Authentication failed for user *user-ID*.

| **Explanation:** The authentication for user *user-ID*
| against the z/OS system failed. Either the user ID or
| password contained in a request was invalid or
| revoked, or the user ID has not been authorized to use
| CIM.

| **System action:** The CIM request is denied.

| **System programmer response:** None.

| **User response:** Check that you are using a valid user
| ID and password and that the user ID has been
| authorized to use CIM. If the problem persists, contact
| the system programmer of the target system to check
| for more detailed authentication error messages on the
| system console.

CFZ17201W Authentication failed for user *user-ID*
because
enableRemotePrivilegedUserAccess is not
set to true.

Explanation: The CIM server refused login for user
user-ID, because *user-ID* is a superuser (UID=0), and the
current CIM server configuration prohibits superuser
logins (the configuration option
enableRemotePrivilegedUserAccess is false).

System action: The CIM request is denied.

System programmer response: To allow superuser
login to the CIM server set the
enableRemotePrivilegedUserAccess configuration option to

true, as described in “Advanced configuration
properties” on page 43.

User response: Either use a non-superuser user ID for
login to the CIM server, or contact your system
administrator to enable superuser login for the CIM
server.

CFZ17202W Request user ID *user-ID* doesn't have
READ permission to profile CIMSERV
CL(WBEM).

Explanation: The user ID requesting a CIM operation
using a remote connection is not permitted to use the
CIM server.

System action: The request is not processed and an
“Access Denied” notification is sent to the client.

System programmer response: Permit the user to
perform CIM requests by giving the user READ access to
profile CIMSERV CL(WBEM).

User response: Contact your system programmer to
permit your user ID to perform CIM requests and
afterwards repeat your request.

CFZ17203W Request user ID *user-ID* misses
password.

Explanation: A request was sent to the CIM server
with user *user-ID* but no password was specified.

System action: The request is rejected as
unauthorized.

System programmer response: None.

User response: Specify a password with your request.

CFZ17204I CIM server authentication is using
application ID OMVSAPPL.

Explanation: The CIM server is using the application
ID 'OMVSAPPL' for authentication.

System action: Application ID 'OMVSAPPL' is used
for authentication.

System programmer response: If the usage of
application ID 'OMVSAPPL' is intended, no action has
to be taken.

Otherwise, if you want to use the application ID
'CFZAPPL',

1. Set the configuration property *enableCFZAPPLID* to
true (see “Advanced configuration properties” on
page 43)
2. Restart the CIM server

User response: None.

CFZ17400W Request user ID *user-ID* does not have READ permission to profile CIMSERV CL(WBEM).

Explanation: The user ID requesting a CIM operation using a local connection is not permitted to use the CIM server.

System action: The request is not processed and an "Access Denied" notification is sent to the client.

System programmer response: Permit the user to perform CIM requests by giving the user READ access to profile CIMSERV CL(WBEM).

User response: Contact your system programmer to permit your user ID to perform CIM requests and afterwards repeat your request.

CFZ17600E Change owner action of security token file failed, which is required for local authentication.

Explanation: The CIM server cannot change the ownership of a file to the user requesting local authentication. The file is located at */tmp* and the file name matches the pattern: *cimclient_<USERID>_**. The file is only valid for a short time. The server should remove this file automatically. It can be deleted.

System action: The request is not processed and an "Access Denied" notification is sent to the client.

System programmer response: Either define CHOWN.UNRESTRICTED in RACF, or grant the CIM server runtime environment user ID READ access to the SUPERUSER.FILES.CHOWN resource in the UNIXPRIV RACF class. For details refer to "Configuring the CIM server's resource authorization model" on page 23.

User response: Contact your system programmer.

CFZ17805I Audit logging is enabled.

Explanation: Audit logging is enabled.

System action: The CIM server starts writing SMF 86 records. These records are only recorded if the SMF configuration contains record 86 and the security is set up accordingly. For details see "Audit logging with SMF record 86" on page 47.

System programmer response: None.

User response: None.

CFZ17806I Audit logging is disabled.

Explanation: Audit logging is disabled.

System action: The CIM server stops writing SMF 86 records.

System programmer response: None.

User response: None.

CFZ18202E CIM server registration with internal SLP failed.

Explanation: The CIM Server failed to register itself as a service for the Service Location Protocol (SLP). Clients will not be able to detect the CIM server on the local networking using the SLP protocol.

System action: None.

System programmer response: Check the system log for further messages indicating CIM server configuration problems or general communication problems. This message usually indicates an issue with the CIM server setup.

User response: None.

CFZ18204I SLP registration initiated.

Explanation: The CIM server has successfully registered itself as a service for the Service Location Protocol (SLP). Clients using the SLP protocol can now detect this CIM server on the local network.

System action: None.

System programmer response: None.

User response: None.

CFZ18603E Could not get CLASSPATH from environment.

Explanation: Initialization of the Java Virtual Machine failed due to environment variable CLASSPATH not being set. The CIM client request cannot be answered as JMPI (Java Managed Provider Interface) providers do not run without a correctly set CLASSPATH.

System action: None.

System programmer response: Set the CLASSPATH as described by the provider.

User response: Contact your system programmer.

IWMCP001E Internal error.

Explanation: An unspecified internal error occurred. The requested operation could not be completed.

System action: No action was performed.

System programmer response: None.

User response: No action required. The function may be successful if invoked again.

IWMCP002E Severe internal error. Explanation: An unspecified internal error occurred. The requested operation might have been partly or completely processed. System action: Operation was partly or fully completed. System programmer response: None. User response: Check the system state. If the operation was not fully completed, the function may be successful if invoked again.	User response: Contact the System Programmer to get the required authorization.
IWMCP003E Memory shortage. Explanation: Storage is not available for the requested operation. The requested operation could not be performed. System action: No action was performed. System programmer response: None. User response: There is a storage shortage. The function may work successfully later on.	
IWMCP004E Module IWMP2PCS missing. Explanation: Unsupported operating system environment. The WLM CIM provider requires z/OS V1R10 or higher. It cannot be used on z/OS V1R9 or lower. System action: No action was performed. System programmer response: Install WLM CIM provider on z/OS V1R10 or higher. User response: None.	
IWMCP005E Invalid or missing parameter. Explanation: One or several CIM provider method parameters are not valid. System action: No action was performed. System programmer response: None. User response: Check the parameters passed to CIM provider methods.	
IWMCP006E Insufficient access rights. Explanation: The caller is not authorized to perform the requested operation. The RACF facility class is active and a profile has been defined for the MVSADMIN.WLM.POLICY RACF facility class profile to which the caller does not have sufficient read or update access. System action: No action was performed. System programmer response: Grant user appropriate access for RACF profile MVSADMIN.WLM.POLICY.	

Appendix A. Sample CIM request and response

The provided sample shows an XML request for an EnumerateInstances request for class IBMzOS_ComputerSystem and the related XML response. See the *CIM Operations over HTTP* and *Representation of CIM in XML* specifications available from the DMTF Published Documents website.

Request:

```
<?xml version="1.0" encoding="utf-8" ?>
<CIM CIMVERSION="2.0" DTDVERSION="2.0">
  <MESSAGE ID="4711" PROTOCOLVERSION="1.0">
    <SIMPLEREQ>
      <IMETHODCALL NAME="EnumerateInstances">
        <LOCALNAMESPACEPATH>
          <NAMESPACE NAME="root"></NAMESPACE>
          <NAMESPACE NAME="cimv2"></NAMESPACE>
        </LOCALNAMESPACEPATH>
        <IPARAMVALUE NAME="ClassName">
          <CLASSNAME NAME="IBMzOS_ComputerSystem"/>
        </IPARAMVALUE>
        <IPARAMVALUE NAME="DeepInheritance">
          <VALUE>TRUE</VALUE>
        </IPARAMVALUE>
        <IPARAMVALUE NAME="LocalOnly">
          <VALUE>FALSE</VALUE>
        </IPARAMVALUE>
        <IPARAMVALUE NAME="IncludeQualifiers">
          <VALUE>FALSE</VALUE>
        </IPARAMVALUE>
        <IPARAMVALUE NAME="IncludeClassOrigin">
          <VALUE>TRUE</VALUE>
        </IPARAMVALUE>
      </IMETHODCALL>
    </SIMPLEREQ>
  </MESSAGE>
</CIM>
```

Response:

```
<?xml version="1.0" encoding="utf-8" ?>
<CIM CIMVERSION="2.0" DTDVERSION="2.0">
  <MESSAGE ID="4711" PROTOCOLVERSION="1.0">
    <SIMPLERSP>
      <IMETHODRESPONSE NAME="EnumerateInstances">
        <IRETURNVALUE>
          <VALUE.NAMEDINSTANCE>
            <INSTANCENAME CLASSNAME="IBMzOS_ComputerSystem">
              <KEYBINDING NAME="CreationClassName">
                <KEYVALUE VALUETYPE="string">IBMzOS_ComputerSystem</KEYVALUE>
              </KEYBINDING>
              <KEYBINDING NAME="Name">
                <KEYVALUE VALUETYPE="string">BOEMT11.boeb.de.ibm.com</KEYVALUE>
              </KEYBINDING>
            </INSTANCENAME>
            <INSTANCE CLASSNAME="IBMzOS_ComputerSystem" >
              <PROPERTY NAME="Caption" TYPE="string">
                <VALUE>IBM z/OS Computer System</VALUE>
              </PROPERTY>
              <PROPERTY NAME="Description" TYPE="string">
                <VALUE>This is an IBMzOS_ComputerSystem</VALUE>
              </PROPERTY>
              <PROPERTY NAME="ElementName" TYPE="string">
```

```

        <VALUE>BOEMT11.boeb.de.ibm.com</VALUE>
    </PROPERTY>
    <PROPERTY NAME="InstallDate" TYPE="datetime">
    </PROPERTY>
    <PROPERTY.ARRAY NAME="OperationalStatus" TYPE="uint16">
        <VALUE.ARRAY>
            <VALUE>0</VALUE>
        </VALUE.ARRAY>
    </PROPERTY.ARRAY>
    <PROPERTY.ARRAY NAME="StatusDescriptions" TYPE="string">
    </PROPERTY.ARRAY>
    <PROPERTY NAME="Status" TYPE="string">
    </PROPERTY>
    <PROPERTY NAME="EnabledState" TYPE="uint16">
        <VALUE>2</VALUE>
    </PROPERTY>
    <PROPERTY NAME="OtherEnabledState" TYPE="string">
        <VALUE>NULL</VALUE>
    </PROPERTY>
    <PROPERTY NAME="RequestedState" TYPE="uint16">
        <VALUE>2</VALUE>
    </PROPERTY>
    <PROPERTY NAME="EnabledDefault" TYPE="uint16">
        <VALUE>2</VALUE>
    </PROPERTY>
    <PROPERTY NAME="TimeOfLastStateChange" TYPE="datetime">
    </PROPERTY>
    <PROPERTY NAME="CreationClassName" TYPE="string">
        <VALUE>IBMzOS_ComputerSystem</VALUE>
    </PROPERTY>
    <PROPERTY NAME="Name" TYPE="string">
        <VALUE>BOEMT11.boeb.de.ibm.com</VALUE>
    </PROPERTY>
    <PROPERTY NAME="PrimaryOwnerName" TYPE="string">
    </PROPERTY>
    <PROPERTY NAME="PrimaryOwnerContact" TYPE="string">
    </PROPERTY>
    <PROPERTY.ARRAY NAME="Roles" TYPE="string">
        <VALUE.ARRAY>
            <VALUE>Unknown</VALUE>
        </VALUE.ARRAY>
    </PROPERTY.ARRAY>
    <PROPERTY NAME="NameFormat" TYPE="string">
        <VALUE>IP</VALUE>
    </PROPERTY>
    <PROPERTY.ARRAY NAME="OtherIdentifyingInfo" TYPE="string">
    </PROPERTY.ARRAY>
    <PROPERTY.ARRAY NAME="IdentifyingDescriptions" TYPE="string">
    </PROPERTY.ARRAY>
    <PROPERTY.ARRAY NAME="Dedicated" TYPE="uint16">
        <VALUE.ARRAY>
            <VALUE>0</VALUE>
        </VALUE.ARRAY>
    </PROPERTY.ARRAY>
    <PROPERTY.ARRAY NAME="OtherDedicatedDescriptions" TYPE="string">
    </PROPERTY.ARRAY>
    <PROPERTY NAME="ResetCapability" TYPE="uint16">
        <VALUE>5</VALUE>
    </PROPERTY>
    <PROPERTY.ARRAY NAME="PowerManagementCapabilities" TYPE="uint16">
    </PROPERTY.ARRAY>
    <PROPERTY NAME="MachineType" TYPE="string">
        <VALUE>2084</VALUE>
    </PROPERTY>
    <PROPERTY NAME="Model" TYPE="string">
        <VALUE>314</VALUE>
    </PROPERTY>

```

```

        <PROPERTY NAME="SerialNumber" TYPE="string">
            <VALUE>0000000000016F7A</VALUE>
        </PROPERTY>
        <PROPERTY NAME="LPARName" TYPE="string">
            <VALUE></VALUE>
        </PROPERTY>
        <PROPERTY NAME="VMGuestID" TYPE="string">
            <VALUE>MT11</VALUE>
        </PROPERTY>
        <PROPERTY NAME="CPUID" TYPE="string">
            <VALUE>026F7A2084</VALUE>
        </PROPERTY>
    </INSTANCE>
</VALUE.NAMEDINSTANCE>
</IRETURNVALUE>
</IMETHODRESPONSE>
</SIMPLERSP>
</MESSAGE>
</CIM>

```

Appendix B. Impacts of the Out-of-Process support for providers

This appendix discusses several aspects that administrators or system programmers need to consider when using the *Out-of-Process* support for providers. The following topics are addressed:

- “Provider management and registration”
- “Provider-based authorization model” on page 36
- “Tracing providers running Out-of-Process”
- “Performance implications” on page 180

Provider management and registration

To manage and register your providers, use classes *PG_Provider* and *PG_ProviderModule* from the Open Group. In addition, there is the following z/OS-specific extension for class *PG_ProviderModule* that you can specify in a provider’s registration MOF file:

boolean ShareAS

Description: When the *ShareAS* property is set to ‘false’, the provider module will run in its own copy of a Provider Agent process. No other provider module will be loaded into this process. Setting the *ShareAS* property to ‘false’ has a major impact on the performance, so it should be set to true unless there is an urgent need for a provider module to be protected from other provider modules running in the same process. Therefore the default of this property is ‘true’.

Setting *ShareAS* to ‘false’ is only honored by the CIM server, if it is running with the configuration property for *forceProviderProcesses* set to ‘true’.

Example of a provider registration MOF file with properties specified for the *Out-of-Process* support:

```
instance of PG_ProviderModule
{
    Name = "OSBase_TestClassProviderModule";
    //The library name on disk
    Location = "cmpiOSBase_TestClassProvider";
    Vendor = "IBM";
    Version = "2.0.0";
    InterfaceType = "CMPI";
    InterfaceVersion = "2.0.0";
    ShareAs = false;
};
```

Tracing providers running Out-of-Process

When tracing is enabled in the CIM server, it is also enabled in the Provider Agent processes. For reasons of trace data integrity and regarding performance aspects, a separate trace file is used for each Provider Agent process.

Each Provider Agent is uniquely identified by the name of the shared Provider Agent executable. Each non-shared instance of a Provider Agent corresponds with a single provider module, the module/executable name uniquely identifies the

process. The tracer is updated to allow an extra name to be specified, and this name is used as an extension to the trace file name specified by the *traceFilePath* configuration property. For example, if the *traceFilePath* is defined as */tmp/cimserver.trc*, the non-shared Provider Agent for the ***OperatingSystemModule*** would direct its trace output to the file */tmp/cimserver.trc.OperatingSystemModule*.

Performance implications

Exploitation of the *Out-of-Process* provider support has some impact on the path length executed for each single CIM request. Most of this path length increase occurs because of the serialization/de-serialization of request and response messages flowing between the CIM server and the Provider Agent address spaces. Also, the path length for provider up-calls back into the CIM server through the *CIMOMHandle* will increase significantly. Therefore, provider developers should use up-calls with care.

Appendix C. Developing CMPI providers for z/OS

The system-specific management data for the CIM Schema and system-specific Schema extension classes are provided through management instrumentation. While some management instrumentation is already provided by the z/OS CIM (see Chapter 5, “z/OS Management Instrumentation for CIM,” on page 79), it is also possible to develop additional management instrumentation for other z/OS resources which are not accessible through the existing z/OS management instrumentation.

Management instrumentation is implemented by developing a provider. A provider is a dynamic load library (DLL) that implements a given interface and contains the program code used by the CIM server to interact with the system resource described by a certain CIM class, for example *CIM_Processor*. Providers are registered with the CIM server for a defined CIM class, allowing the CIM server to route all client requests directed against this class to the provider for interacting with the resource. A provider logically acts as an extension of the CIM server for interfacing directly with the managed resources.

Providers are the de facto standard concept for developing management instrumentation, though this purpose of providers is not explicitly mentioned by the various CIM and WBEM standards available from the DMTF. The *Common Manageability Programming Interface* (CMPI) technical standard was defined by *The Open Group* to allow for developing providers independently from a specific CIM server implementation.

Figure 14 shows the CMPI provider interfaces:

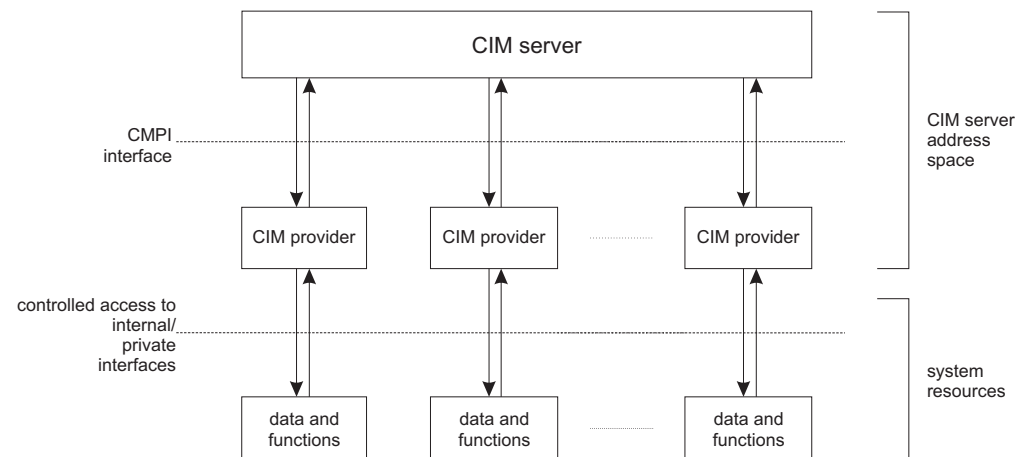


Figure 14. CMPI provider interfaces

CMPI is a C-based programming interface for providers designed for binary compatibility. All management instrumentation included with the z/OS CIM server was developed following the CMPI standard and CMPI is the only supported provider programming interface for the z/OS CIM server. Documentation about the CMPI Technical Standard is available from *The Open Group* and is not repeated in any documentation available for z/OS. Developers of management instrumentation for z/OS need to be familiar with the CMPI and CIM/WBEM

standards. The information contained here explains the specific aspects that need to be considered for developing CMPI Providers for z/OS.

To be able to develop a CMPI provider for z/OS, a set of C header files is required that define the CMPI interface (see “Obtaining the required header files”).

Due to legal implications with the OpenSource nature of these files, they are not provided together with z/OS CIM, but must be obtained from their original location at *The Open Group* instead. “Obtaining the required header files” explains how to accomplish this.

Due to the CMPI interface design, you need not link a CMPI provider to any library of the z/OS CIM server. Only the header files are needed for developing a CMPI provider library.

Obtaining the required header files

Before you can start to develop a provider DLL, you must obtain the following C header files from the OpenPegasus project through the internet:

	cmpidt.h	Data type definitions
	cmpift.h	Function signature definitions in the form of function tables
	cmpimacs.h	CMPI convenience macros (optional)

These files are available in the *OpenPegasus CVS Repository*. Users familiar with *CVS* can check out these files using a *CVS* client on any platform by following the instructions on <http://www.openpegasus.org/> in the *CVS Overview* section. The required files are located in directory `pegasus/src/Pegasus/Provider/CMPI`. To get the correct version of the files, they need to be checked out with the `RELEASE_2_7_1` tag.

Users who are not familiar with using *CVS*, are recommended to obtain the files through a web browser starting at <http://www.openpegasus.org/>. From the main page switch to the *Web CVS* section from where you can navigate to the required CMPI files by clicking on the following directory names:

pegasus (see Figure 15 on page 183) → **src** → **Pegasus** → **Provider** → **CMPI**

OpenPegasus CVS Repository

File
 Attic/ [show contents]
 atlas/
 cvstest/
 deleteme/
 dmtdocs/
 junk/
 pegasus/
 pegasus-JavaCIMClient/
 pegasus_unsupported/
 xxxx/

Figure 15. OpenPegasus CVS Repository

Once you have successfully navigated to the CMPI directory, the required header files are found at the end of the list of displayed files. To get the correct version of the files, the tag `RELEASE_2_7_1` needs to be selected from the list at the bottom of the file list.

To download the files, first click on the version number displayed in the column after each file name and then click on **download** on the next screen where the content of the file is displayed. Once you have successfully downloaded the files, transfer them to the z/OS system on which the provider DLL(s) will be developed, ideally to a ZFS directory. Please note that when transferring files from the workstation to a z/OS system, they should be converted from ASCII to EBCDIC encoding.

There are also a couple of samples for CMPI providers available on the **OpenPegasus CVS Repository**. They can be obtained the same way as the header files by navigating to the `pegasus/src/Providers/sample/CMPI` directory.

General aspects for developing a provider

Before you can start to develop a CMPI provider, you first need to have the CIM class model containing descriptions for the resource to be instrumented in the form of a CIM class. Such a class should follow the standards from **DMTF Standards and Initiatives**, and in particular it should be consistent with the CIM Schema supported by the CIM server. Usually, a CIM class for which a provider is written, is derived from one of the classes in the CIM Schema provided by the DMTF, and named with a vendor-specific class name prefix. For example, the prefix `IBMzOS_` is used for all classes provided by IBM for the z/OS operating system. This naming scheme also helps to prevent conflicts with the resources that have already been instrumented for CIM by IBM or other vendors.

Note: In general it is not recommended to create new providers for resources that have already been instrumented by IBM.

Provider initialization and function signatures

The nature of a CMPI provider does not require static linking to any of the CIM server's libraries. Instead, for each provider function group a single initialization routine (factory) entry point must be exposed following a defined naming scheme, so that the CIM server can call this entry point by name once it has dynamically loaded a provider DLL. The CIM server will attempt to determine the function groups supported by a provider and the respective entry points by verifying the existence of the according provider factory entry points.

The signature for the factory functions looks like this:

```
CMPI<mi-type>MI * <mi-name>_Create_<mi-type>MI(CMPIBroker*,
                                                CMPIContext*,
                                                CMPIStatus*);
```

<mi-type> refers to the function group of the provider, and <mi-name> refers to the actual provider name as specified during provider registration.

Important:

The actual signature of this function has an additional '_' after '_Create', which is not described as such in the initial version of the *CMPI Technical Standard*, but is changed in a Corrigendum to match the existing implementations of the CMPI interface.

The factory function must return a pointer to a valid CMPI<mi-type>MI structure where the major component of this structure is the table holding the function pointers, and thus enabling access to the individual provider group functions for the CIM server. An example of such a function pointer is the pointer to the *enumerateInstances* function in the CMPIInstanceMI structure.

The function groups for CMPI providers are *Instance*, *Association*, *Property*, *Method* or *Indication*, where type *Property* is not supported by the z/OS CIM server.

In file *cmpimacs.h*, a set of C preprocessor macros are defined that you may use for the provider initialization code and through which the required code for the <mi-name>_Create_<mi-type>MI function is generated in a convenient way. These macros are called *CM<mi-type>MIStub* and they are used in many of the examples referenced in "Samples" on page 191.

For further details please read chapter *MI Factories* of the *CMPI Technical Standard Document* provided by *The Open Group*.

For each of the CMPI provider function groups, a set of C functions must be implemented as described in chapter *MI Function Signatures* of the *CMPI Technical Standard Document*.

Instance provider functions

Instance providers are the most common kind of management instrumentation. They implement the basic access to the resources described in a CIM class. With an instance provider it is possible to create, enumerate, modify, delete, query or simply retrieve system resources:

- cleanup(...)

- enumInstanceNames(...)
- enumInstances(...)
- getInstance(...)
- createInstance(...)
- modifyInstance(...)
- deleteInstance(...)
- execQuery(...)

Method provider functions

Method providers are needed to implement the methods defined for a CIM class.

- cleanup(...)
- invokeMethod(...)

Association provider functions

Association providers are needed to implement the relationships between system resources as defined by the association classes.

- cleanup(...)
- Associators(...)
- AssociatorNames(...)
- References(...)
- ReferenceNames(...)

Indication provider functions

Event or indication providers must be implemented for event subscription and notification:

- cleanup(...)
- AuthorizeFilter(...)
- MustPoll(...)
- ActivateFilter(...)
- DeActivateFilter(...)
- EnableIndications(...)
- DisableIndications(...)

Please note that the function *MustPoll* is not supported for z/OS.

Security aspects

When developing a CMPI provider for z/OS, consider the security context in which the provider runs. Besides the levels of security provided by the z/OS CIM server for authentication and authorization, a provider is processed in the context of a user ID:

Requestor's user ID

By default, a provider is processed in the context of the requestor's user ID for all invocations that are caused by an external CIM operation. This means that the provider runs under the identity of the requestor's user ID, and resource access authorization occurs against this user ID. See the usage notes for the *pthread_security_np* call in chapter *Callable services descriptions* in *z/OS UNIX System Services Programming: Assembler Callable Services Reference* for additional information.

Designated user ID

Alternatively, you can provide a designated user ID that runs the provider.

Specify the designated user ID during provider registration using the *UserContext* and *DesignatedUserContext* properties of the *PG_ProviderModule* class.

When a provider is registered with a designated user ID, the CIM server processes all requests under the designated user ID, regardless which client user ID has issued the request.

The user ID of the requestor is still available for the provider and should be used for further authorization checking in order to prevent unauthorized access to a resource. You have to specify similar security definitions for the designated user ID as for regular client users, as described in “Switching identity (surrogate)” on page 27.

Data conversion ASCII / EBCDIC

The z/OS CIM server executes in the Enhanced ASCII mode. This means that all string data within the CIM server’s address space is represented in ASCII rather than EBCDIC encoding. For a provider this means that all string data exchanged with the CIM server is expected to be in ASCII, or more precisely, in UTF8 format. Since the native data of z/OS resources is usually represented in EBCDIC, the provider code needs to convert this data before it can return it to the CIM server through the CMPI interface, or when it receives data from the CIM server through the CMPI interface.

Therefore it is also recommended to compile the provider’s C code using the ASCII option of the z/OS XL C/C++ compiler. Please note that using the ASCII option also requires the XPLINK compile and link option.

See *Appendix B* in the *z/OS XL C/C++ Run-Time Library Reference* for additional information about the Enhanced ASCII support. Also see the *z/OS XL C/C++ Guide* and the *z/OS XL C/C++ Programming Guide* for details about the ASCII compiler option.

Registering a provider with the CIM server

In order for the CIM server to be able to recognize a provider, you first need to store the provider in the CIM server’s provider directory and then register it with the CIM server for the respective CIM class.

The providers belonging to the z/OS operating system are located in the `/usr/lpp/wbem/provider` hierarchical file system directory. It is not recommended to store non-IBM providers in this directory as well. Either store them in their own separate hierarchical file system directory and create a symbolic link into the CIM server owned provider directory, or extend the CIM server’s search list for provider directories by setting the *providerDir* configuration property. See section “Advanced configuration properties” on page 43 for details on how to set this property.

On systems where “program control” is enabled, you must also flag the provider DLL as program controlled. You achieve this using the `extattr UNIX System Services` command (`extattr +p <providerfile>`). Details about enabling shared libraries for program control can be found in *Defining modules to program control* in *z/OS UNIX System Services Planning*.

Once the provider DLL has been made physically accessible to the CIM server, it needs to be registered via a special MOF file using the cimmo command. A provider registration MOF file contains instances of the CIM classes from the provider registration schema, namely of classes *PG_ProviderModule*, *PG_Provider* and *PG_ProviderCapabilities* as shown in the following provider registration schema:

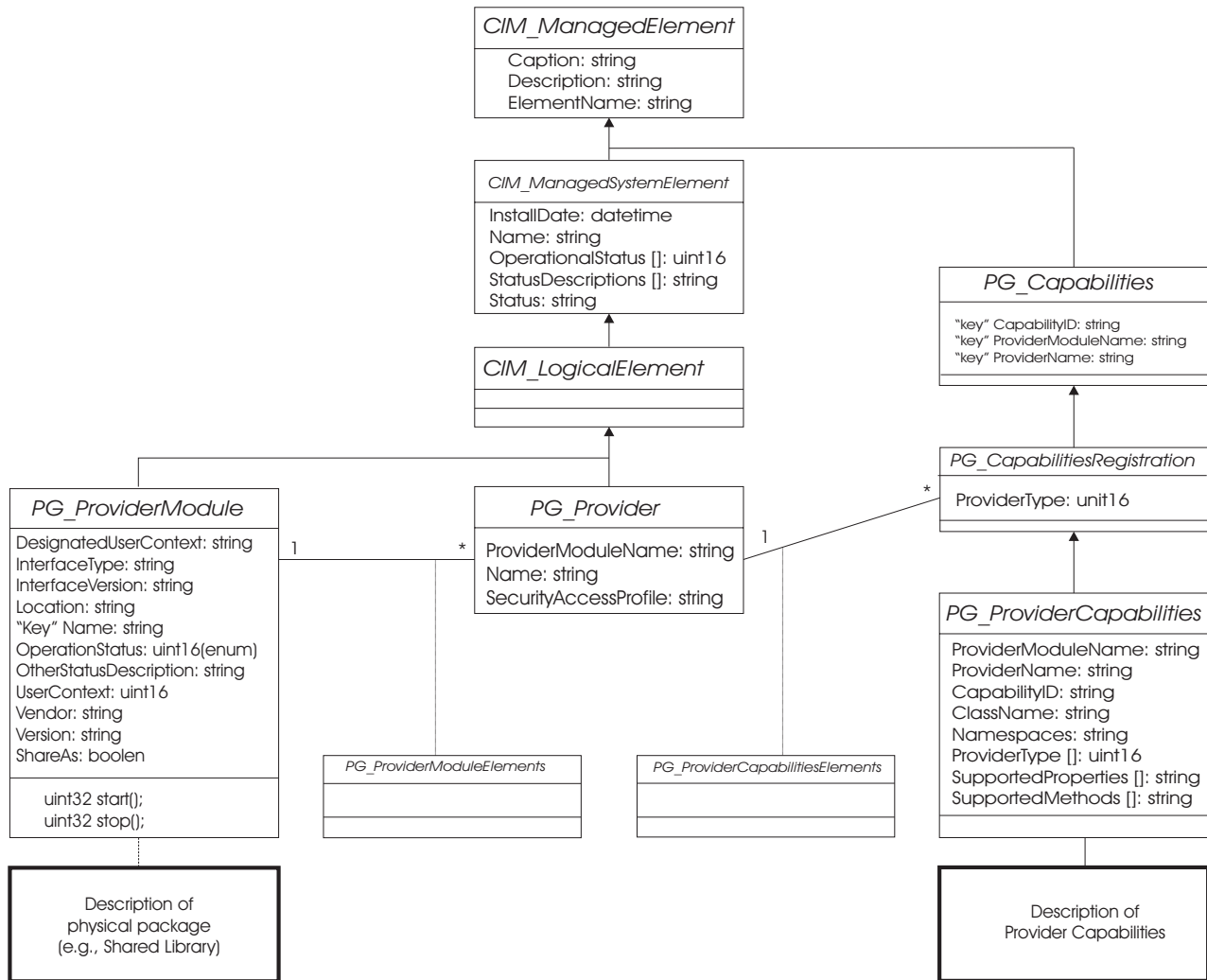


Figure 16. CIM classes from the provider registration schema

The instances of these classes contain all the information that the CIM Server needs to know about a provider, for example its physical packaging structure, supported CIM classes and namespaces, as well as the set of supported provider operations.

PG_Provider

Class *PG_Provider* is the logical representation of a CIM provider. Its only properties are the name of the provider, the name of the provider module in which the code of the provider physically resides and the name of a SAF security profile to be checked before a client is granted access to the provider.

The following table shows the relevant properties of class *PG_Provider*:

Table 8. Properties of class *PG_Provider*

Property name	Property type	Description
ProviderModuleName	string	The name of the provider module containing the code for this provider. This name needs to match the value of the <i>Name</i> property of the corresponding instance of class <i>PG_ProviderModule</i> .
Name	string	The name of the provider. This name is used to identify a specific provider within a provider module (DLL) and specifies the prefix of a provider's <i>_Create_<mi-type>MI()</i> initialization function.
SecurityAccessProfile	string	This property defines the name of a z/OS security server's profile in the CIM server's WBEM class that will be checked for a requestor's access before a request is routed to this provider. Depending on the type of the CIM operation, a different level of access to the security profile is required as listed in Table 3 on page 26. This is not a required property and can be omitted from the provider registration MOF.

Example of an instance of class PG_Provider in MOF syntax:

```
instance of PG_Provider
{
    //The provider module as defined in PG_ProviderModule
    ProviderModuleName = "TestClassProviderModule";
    // The provider name as referenced in the code
    Name = "TestClassProvider";
};
```

PG_ProviderModule

Class *PG_ProviderModule* represents the physical packaging of one or more providers in a DLL or shared library.

The following table shows the relevant properties of class *PG_ProviderModule*:

Table 9. Properties of class *PG_ProviderModule*

Property name	Property type	Description
Name	string	The logical name of the provider module.
Vendor	string	The name of the provider module vendor, for example, IBM.
Version	string	The provider module version.
InterfaceType	string	The interface type implemented by the provider. Must be CMPI for z/OS.
InterfaceVersion	string	The interface version number implemented by the provider. Must be 2.0.0 for CMPI on z/OS.

Table 9. Properties of class *PG_ProviderModule* (continued)

Property name	Property type	Description
Location	string	The name of the DLL or shared library in the hierarchical file system without a path name. The name specified for <i>Location</i> is automatically prefixed with <i>lib</i> and extended with <i>.so</i> by the CIM server: <i>lib<Location>.so</i>
ShareAS	boolean	Setting the <i>ShareAS</i> property to false causes the provider module to run in its own copy of a Provider Agent process. No other provider module will be loaded into this process. Setting the <i>ShareAS</i> property to false has a major impact on the performance, so you should not set it to 'false' unless there is an urgent need for a provider module to be protected from other provider modules. The default setting of <i>ShareAS</i> is true. Setting <i>ShareAS</i> to false is only honored by the CIM server, if it is running with the configuration property <i>forceProviderProcesses</i> set to true.
UserContext	uint16	Defines the user context in which this provider module is invoked. Values: 2 (Requestor), default The provider is invoked in the security context of the user requesting an operation. 3 (Designated User) The provider is invoked in the security context of the user ID specified by the <i>DesignatedUserContext</i> property. See "Running providers in a designated user context" on page 37 for a general description on running a provider module with a designated user context.
DesignatedUserContext	string	Specifies the user ID providing the context in which this provider module is invoked (regardless of which user requests an operation). Values: NULL when <i>UserContext</i> = 2 non-NULL value when <i>UserContext</i> = 3 See "Running providers in a designated user context" on page 37 for a general description on running a provider module with a designated user context.

Example of an instance of class PG_ProviderModule in MOF syntax:

```
instance of PG_ProviderModule
{
    Name = "TestClassProviderModule";
    //The library name on disk
    Location = "TestClassProvider";
}
```

```

// (will be extended to libTestClassProvider.so)
Vendor = "IBM";
Version = "1.0.0";
InterfaceType = "CMPI";
InterfaceVersion = "2.0.0";
ShareAS = true;
UserContext = 2;
};

```

PG_ProviderCapabilities

Class *PG_ProviderCapabilities* describes the specific capabilities of a provider. Multiple instances of *PG_ProviderCapabilities* can be created for each provider allowing the same provider to be registered, for example, for multiple CIM classes.

The following table shows the relevant properties of class *PG_ProviderCapabilities*:

Table 10. Properties of class *PG_ProviderModule*

Property name	Property type	Description
ProviderModuleName	string	The name of the provider module as specified in the corresponding instances of classes <i>PG_Provider</i> and <i>PG_ProviderModule</i> .
ProviderName	string	The name of the provider as specified in the corresponding instance of class <i>PG_Provider</i> .
CapabilityID	string	A value that uniquely identifies this <i>Capabilities</i> instance within the set of <i>Capabilities</i> for the designated provider.
ProviderType	uint16[]	Enumerates the kind of provider capabilities (=supported operations) defined for the associated provider: 2 Instance 3 Association 4 Indication 5 Method 6 IndicationConsumer (not supported for z/OS) 7 InstanceQuery
ClassName	string	Describes the CIM class for which the associated provider supplies instances, associations or indications information.
Namespaces	string[]	Describes the namespaces that are supported by the provider for this CIM class.
SupportedProperties	string[]	Lists the properties supported by this provider. If this array is empty, the provider must support all of the properties defined in the class.
SupportedMethods	string[]	Lists the methods supported by this provider. If this array is empty, the provider must support all the methods defined in the class.

Example of an instance of class PG_ProviderCapabilities in MOF syntax:

```

instance of PG_ProviderCapabilities
{
    //The provider module as defined in PG_ProviderModule
    ProviderModuleName = "TestClassProviderModule";
    //The provider name as defined in PG_Provider

```

```

    ProviderName = "TestClassProvider";
    CapabilityID = "1";
    //Name of the CIM class as defined in the mof
    ClassName = "IBMzOS_TestClassB";
    Namespaces = {"root/cimv2","root/test"};
    ProviderType = { 2, 5 }; // Instance, Method
    SupportedProperties = NULL; // All properties
    SupportedMethods = NULL; // All methods
};

instance of PG_ProviderCapabilities
{
    //The provider module as defined in PG_ProviderModule
    ProviderModuleName = "TestClassProviderModule";
    //The provider name as defined in PG_Provider
    ProviderName = "TestClassProvider";
    CapabilityID = "2";
    //Name of the CIM class as defined in the mof
    ClassName = "IBMzOS_TestIndication";
    Namespaces = {"root/cimv2"};
    ProviderType = { 4 }; // Indication
    SupportedProperties = NULL; // All properties
    SupportedMethods = NULL; // All methods
};

```

Provider registration processing

Once the provider registration MOF file has been created with the instances of classes *PG_Provider*, *PG_ProviderModule* and *PG_ProviderCapabilities*, the content of this MOF file can be loaded into the CIM server's root/PG_InterOp namespace using the `cimmof` command described in "cimmof/cimmofl" on page 53.

Example:

```
cimmof -n root/PG_InterOp TestProviderRegistration.mof
```

Samples

Examples for CMPI providers can be found on the *OpenPegasus CVS Repository*, located in the `pegasus/src/Providers/sample/CMPI` directory. You can access them in the same ways as described in "Obtaining the required header files" on page 182. Please note that these examples have been enabled for z/OS only in an *OpenPegasus* build environment and will need some minor adoptions for a custom build environment.

Additional examples are available from the *SBLIM OpenSource* project and can be downloaded from <http://www.sblim.org>. Although the CIM providers from SBLIM apply to Linux platforms only, they are good examples for how to write CIM providers in general. The SBLIM project also provides a number of useful tools and documents related to provider development.

Appendix D. Cluster / CoupleDataset / JES2-JES3Jobs Provider Setup

For using the Job and Cluster providers, some additional setup has to be done.

- ___ 1. Configure the Common Event Adapter (CEA):
 - ___ a. Define additional parameters in PARMLIB (see “Required PARMLIB updates”)
 - ___ b. Prepare RACF for CEA (see “Required RACF setup”)
- ___ 2. When running in a sysplex, format the sysplex couple dataset to allow it to be cluster capable (see “Sysplex couple dataset formatting” on page 196).

Required PARMLIB updates

The following PARMLIB parameters have to be defined to enable the Job and Cluster providers:

MAXCAD limit

This parameter defaults to 50. If the installation sets a lower limit, it may be necessary to increase this setting to accommodate the Common Event Adapter (CEA) Common Area Data Space (CADS).

APF Authorize SYS1.MIGLIB

The following must be added to the installation’s PROGxx member in PARMLIB to enable the CFRM-related CIM providers to function:

```
APF ADD DSNAME(SYS1.MIGLIB) VOLUME(*****)
```

REXX Alternate Library

The Couple Dataset providers require the use of compiled REXX execs provided as part of the z/OS 1.9 SYSREXX support. These execs require the use of the REXX alternate library. The following addition to the installation’s PROGxx member in PARMLIB is one way to accomplish this.

```
LNKLST ADD,NAME(LNKLST00),DSN(REXX.V1R3M0.SEAGALT),ATTOP
```

Required RACF setup

The following sample TSO CLIST performs the necessary RACF setup to permit CEA to use Automatic Restart Manager (ARM), and to permit CEA to operate in UNIX System Services (USS) with the Cluster, Couple Dataset and JES2/JES3 Jobs CIM providers.

Table 11. Sample TSO CLIST

```

PROC 0
/*****
/*    LICENSED MATERIALS - PROPERTY OF IBM.                                */
/*                                                                              */
/*    5694-A01                                                                */
/*                                                                              */
/*    (C) COPYRIGHT IBM CORP. 2004, 2007 ALL RIGHTS RESERVED                */
/*                                                                              */
/*    US GOVERNMENT USERS RESTRICTED RIGHTS - USE,                        */
/*    DUPLICATION OR DISCLOSURE RESTRICTED BY GSA ADP                      */
/*    SCHEDULE CONTRACT WITH IBM CORP.                                     */
/*                                                                              */
/*****
/*
* This is a sample TSO CLIST which provides all the RACF commands
* needed to prepare for using z/OS CIM providers.
*
* This CLIST assumes that the CIM administrator, CFZADM, and a
* CIM user, CFZUSR1, have already been defined.
*
* ENSURE YOU EXECUTE THIS CLIST FROM A RACF SPECIAL USER ID!
* =====
* CAUTION: This is neither a JCL procedure nor a complete CLIST.
* All of the commands have been commented out to prevent
* accidental execution. Customers must uncomment the necessary
* lines before running this CLIST to perform the respective
* commands. In addition, site dependent customization might
* need to be performed to this CLIST before execution.
*
* NOTE: It is entirely possible that some of these
* commands will return a non-zero return code since the specific
* RACF function may already be set up.
*****/

/*-----
* Step 1 - Setup the Common Event Adapter (CEA) Support
*         Define the CEA id with UNIX System Services support.
*-----*/
/* ADDUSER CEA DFLTGRP(SYS1) OMVS(UID(<new_uid>) HOME('/') FILEPROCMAX(1024)) +
* SPECIAL
* RDEFINE STARTED CEA.** STDATA(USER(CEA) GROUP(SYS1) TRACE)
*/

/*-----
* Step 2 - Setup CEA for Multi-Layer Security (MLS)
*         This step is only necessary if the installation is
*         already using MLS. Otherwise, it can be skipped.
*-----*/
/* ALTUSER CEA SECLABEL(SYSMULTI)
* PERMIT SYSMULTI CLASS(SECLABEL) ACCESS(ALTER) ID(CEA)
* PERMIT SYSHIGH  CLASS(SECLABEL) ACCESS(ALTER) ID(CEA)
* PERMIT SYSLOW   CLASS(SECLABEL) ACCESS(ALTER) ID(CEA)
* ADDSD 'CEA.*' SECLABEL(SYSNONE)
*
* SETROPTS GENERIC(STARTED) REFRESH
* SETROPTS RACLIST(STARTED) REFRESH
* SETR CLASSACT(SECLABEL)
* SETR RACLIST(SECLABEL) REFRESH
*/

```

Table 11. Sample TSO CLIST (continued)

```

/*-----
* Step 3 – Setup for CIM Providers use CEA to connect/subscribe to
*           to events, to perform JES operations and to access JES
*           job properties.
*
* Security profiles must be defined by the security administrator
* in the SERVAUTH class. The profiles will cover resource names of
* the form: CEA.operationtype.eventname where 'operationtype'
* corresponds with "connect" and "subscribe" requests, and 'eventname'
* corresponds to the name of the specific event being subscribed by the
* CIM provider software. Each portion of the resource name will allow
* wildcarding ('*' notation) for generic permissions.
*
* For the Cluster and JES Jobs CIM providers, IBM recommends
* specifying a profile of CEA.*
*
* If more restrictive security settings are desired, the following
* may be used:
*
* + The CIM provider's MVS userid requires authority to connect
*   to CEA:
*   Define the SERVAUTH profile CEA.CONNECT and grant read access
*   to the ID or Group.
*
* + The CIM provider's MVS userid requires authority to be notified
*   of provider-specified events:
*   Define the SERVAUTH profile CEA.SUBSCRIBE.* and grant read
*   access to the ID or Group.
*
* + The CIM provider's MVS userid requires authority to perform JES
*   operations:
*   Define the SERVAUTH profile CEA.CEADOCMD and grant read
*   access to the ID or Group.
*
* + The CIM provider's MVS userid requires authority to retrieve
*   properties for JES jobs:
*   Define the SERVAUTH profile CEA.CEAGETPS and grant read
*   access to the ID or Group.
*-----*/

/* Allow GENERIC (wildcard) naming conventions for the
*   SERVAUTH class:                                     */
/* SETROPTS GENERIC(SERVAUTH)
*/

/* Define the CEA resource profiles required to connect/subscribe to
*   events and to perform operations/retrieve properties for JES.
*/
/*   RDEFINE SERVAUTH CEA.CONNECT          UACC(NONE)
*   RDEFINE SERVAUTH CEA.SUBSCRIBE.*      UACC(NONE)
*   RDEFINE SERVAUTH CEA.CEAGETPS         UACC(NONE)
*   RDEFINE SERVAUTH CEA.CEADOCMD         UACC(NONE)
*/

```

Table 11. Sample TSO CLIST (continued)

```

/* Grant the CIM administrator userid, CFZADM, authority to the
 * following and to grant access to the same for other CIM users.
 * + connect to CEA
 * + subscribe to events
 * + perform JES operations
 * + obtain job properties */
/* PERMIT CEA.CONNECT CLASS(SERVAUTH) ID(CFZADM) ACCESS(UPDATE)
 * PERMIT CEA.SUBSCRIBE.* CLASS(SERVAUTH) ID(CFZADM) ACCESS(UPDATE)
 * PERMIT CEA.CEAGETPS CLASS(SERVAUTH) ID(CFZADM) ACCESS(UPDATE)
 * PERMIT CEA.CEADOCMD CLASS(SERVAUTH) ID(CFZADM) ACCESS(UPDATE)
 */

/* Grant the CIM user id, CFZUSR1, authority to the following:
 * + connect to CEA
 * + subscribe to events
 * + perform JES operations
 * + obtain job properties */
/* PERMIT CEA.CONNECT CLASS(SERVAUTH) ID(CFZUSR1) ACCESS(READ)
 * PERMIT CEA.SUBSCRIBE.* CLASS(SERVAUTH) ID(CFZUSR1) ACCESS(READ)
 * PERMIT CEA.CEAGETPS CLASS(SERVAUTH) ID(CFZUSR1) ACCESS(UPDATE)
 * PERMIT CEA.CEADOCMD CLASS(SERVAUTH) ID(CFZUSR1) ACCESS(UPDATE)
 */

/* Activate authority checking for the SERVAUTH class: */
/* SETROPTS RACLIST(SERVAUTH) CLASSACT(SERVAUTH)
 */

/* If the SERVAUTH class is already active, issue:
 */
/* SETROPTS RACLIST(SERVAUTH) REFRESH
 */

/*-----
 * Step 4 – Setup for Cluster/Couple Dataset Providers
 * Read access for CIM users and update access for CIM
 * administrators.
 *-----*/
/* RDEF FACILITY MRCLASS.CLUSTER UACC(READ)
 * PERMIT MRCLASS.CLUSTER CLASS(FACILITY) ID(CFZUSR1) ACCESS(READ)
 * PERMIT MRCLASS.CLUSTER CLASS(FACILITY) ID(CFZADM) ACCESS(UPDATE)
 *
 * Read access required to enumerate instances/get instance for
 * IBMzOS_CFRMPolicy.
 * PERMIT MVSADMIN.XCF.CFRM CLASS(FACILITY) ID(CFZUSR1) ACCESS(READ)
 *
 * Alter access required to invoke the deletePolicy method.
 * PERMIT MVSADMIN.XCF.CFRM CLASS(FACILITY) ID(CFZADM) ACCESS(ALTER)
 * SETR CLASSACT(FACILITY) RACLIST(FACILITY)
 */

```

Sysplex couple dataset formatting

To format the sysplex couple dataset, use the IXCL1DSU format utility specifying ITEM NAME(CLUSTER) NUMBER(1).

The following table shows a sample JCL formatting the sysplex couple dataset for enabling cluster functions. The IXCSYSPF member has been updated to indicate the new CLUSTER keyword.

Table 12. Sample sysplex couple dataset formatting JCL

```

IXCSYSPF JOB
*
* SAMPLE JCL TO FORMAT THE PRIMARY AND/OR ALTERNATE COUPLE DATA SETS
* - SYSPLEX COUPLE DATA SETS
*
* 1. SYSPLEX NAME IS REQUIRED AND IS 1-8 CHARACTERS
* 2. SYSPRINT DD IS A REQUIRED DD STATEMENT FOR FORMAT UTILITY
*   MESSAGES
* 3. SYSIN DD IS A REQUIRED DD STATEMENT FOR FORMAT UTILITY CONTROL STATEMENTS
*
//STEP1 EXEC PGM=IXCL1DSU
//STEPLIB DD DSN=SYS1.MIGLIB,DISP=SHR
//SYSPRINT DD SYSOUT=A
//SYSIN DD *
        DEFINEDS SYSPLEX(PLEX1)
                DSN(SYS1.XCF.CDS01) VOLSER(CDSPK1)
                MAXSYSTEM(8)
                CATALOG
        DATA TYPE(SYSPLEX)
                ITEM NAME(GROUP) NUMBER(50)
                ITEM NAME(MEMBER) NUMBER(120)
                ITEM NAME(GRS) NUMBER(1)
                ITEM NAME(CLUSTER) NUMBER(1)
        DEFINEDS SYSPLEX(PLEX1)
                DSN(SYS1.XCF.CDS02) VOLSER(CDSPK1)
                MAXSYSTEM(8)
                CATALOG
        DATA TYPE(SYSPLEX)
                ITEM NAME(GROUP) NUMBER(50)
                ITEM NAME(MEMBER) NUMBER(120)
                ITEM NAME(GRS) NUMBER(1)
                ITEM NAME(CLUSTER) NUMBER(1)
/*

```


Appendix E. WLM classes

IBMzOS_WLM

This class represents the z/OS Workload Manager. READ access to the RACF facility class MVSADMIN.WLM.POLICY is required to access this class.

The following properties have been implemented:

Property	Description
string Caption	A short description of the class
string Description	A description of the class
string ElementName	Name given to this instance of the class
datetime InstallDate	Not supported
uint16 OperationalStatus[]	The current status of WLM: [2] [OK]
string StatusDescriptions[]	Not supported
string Status	Not supported
uint16 HealthState	The health status of WLM: 5 OK
uint16 EnabledState	Indicates the Enabled or Disabled state: 2 Enabled
string OtherEnabledState	Not supported
uint16 RequestedState	The last requested state: 2 Enabled
uint16 EnabledDefault	Indicates the default value for Enabled State: 2 Enabled
datetime TimeOfLastStateChange	Not supported
string SystemCreationClassName [key]	The scoping system's CreationClassName
string SystemName [key]	The name of the scoping system
string CreationClassName [key]	Indicates the name of the class used in the creation of an instance
string Name [key]	Name of z/OS Workload Management service
string PrimaryOwnerName	Not supported
string PrimaryOwnerContact	Not supported
boolean Started	Indicates if z/OS WLM runs
string ActiveServicePolicy	Name of WLM service policy activated for the sysplex
string PolicyDescription	Description of the WLM service policy activated for the sysplex
datetime PolicyActivationTimestamp	The time the WLM service policy has been activated
string PolicyActivationUser	Userid that activated the WLM service policy
string PolicyActivationSystem	System from which the WLM service policy activation was triggered

Property	Description
string RelatedServiceDefinition	Name of the service definition the WLM service policy was activated from
datetime ServiceDefinitionInstallationTimestamp	Time the service definition was installed
string ServiceDefinitionInstallationUser	User that installed the service definition
string ServiceDefinitionInstallationSystem	System from which the service definition installation was triggered
uint8 ServiceDefinitionFunctionalityLevel	Functionality level of the service definition
string EmbeddedEWLMPolicy	Name of the EWLM policy embedded in the active WLM service policy
datetime EWLMDMPolicyActivationTimestamp	Time the EWLM Domain Manager has triggered the activation of the EWLM policy that is activated on this system
datetime EWLMPolicyActivationTimestamp	Time the EWLM Managed Server has activated the EWLM policy that is activated on this system
datetime EWLMManagementActivationTimestamp	Time when management towards EWLM goals has been activated on this system
boolean PolicyActivationInProgress	Indicates whether a WLM policy activation is currently in progress
boolean AbnormalSystemConfiguration	Indicates an abnormal system configuration
string PolicyActivatingSystem	If a WLM policy activation is currently in progress, the name of the system where the policy activation was triggered
uint8 WLMVersion	WLM version
uint16 CDSFormat	WLM Couple Dataset format
string SysplexMembersSystemName[]	Name of systems in sysplex
uint8 SysplexMembersWLMMode[]	Workload management mode of systems in sysplex: 0 Undefined 1 Compatibility Mode 2 Goal Mode 3 EWLM Mode
uint8 SysplexMembersWLMStatus[]	Workload management status of systems in sysplex: 0 Undefined 1 Initializing 2 Active 3 Active, Not Running with Active Policy 4 Quiesce in Progress 5 Cleanup Initiated by System 6 WLM Inactive, Cleanup Complete 7 Unknown 8 System Inactive, Cleanup Pending 9 System Inactive, Cleanup Complete 10 Unknown
string SysplexMembersActivePolicy[]	Name of WLM service policy active on systems in sysplex
datetime SysplexMembersPolicyActivationTimestamp[]	Time the WLM service policy was activated on systems in sysplex

Property	Description
string SysplexMembersCleaningSystem[]	If WLM state is 'Cleanup Initiated by System', the name of the system performing the cleanup
string CouplingFacilityStructureNames[]	Name of the WLM coupling facility structures
uint8 CouplingFacilityStructureStatus[]	Status of the WLM coupling facility structures: 0 Disconnected 1 Connected

Methods of IBMzOS_WLM

Class IBMzOS_WLM provides the following methods:

Method	Description
uint32 RequestStateChange()	Not supported
uint32 StartService()	Not supported
uint32 StopService()	Not supported
uint32 ActivateServicePolicy()	Activate a service policy contained in the WLM service definition installed in the WLM couple dataset. UPDATE access to the RACF facility class MVSADMIN.WLM.POLICY is required to successfully invoke this method. Successful execution of this method is indicated by an IBMzOS_WLMPolicyActivationIndication indication.
uint32 InstallServiceDefinition()	Install the passed service definition to the WLM couple dataset. UPDATE access to the RACF facility class MVSADMIN.WLM.POLICY is required to successfully invoke this method.
uint32 ExtractServiceDefinition()	Extract the service definition from the WLM couple dataset. READ access to the RACF facility class MVSADMIN.WLM.POLICY is required to successfully invoke this method.
uint32 UploadServiceDefinition()	Save service definition in XML format in a sequential dataset.
uint32 DownloadServiceDefinition()	Download a service definition that is stored in XML format in a sequential dataset.

Indications for IBMzOS_WLM

Class IWMzOS_WLM provides the following indications:

Indication	Description
IBMzOS_WLMPolicyActivationIndication	A 'process' indication that indicates that a service policy has been activated in the sysplex. This event occurs on each system in the sysplex.

	IBMzOS_WLMOS association
	This class associates an IBMzOS_WLM with an IBMzOS_ComputerSystem.

Appendix F. Reason codes

The following list of reason codes may be returned by the methods in the Jobs providers. The first four digits (X'xxxx') may be any value.

Table 13. Jobs providers' reason codes

Reason code (hex)	Description	User action	IBM Service Information
X'xxxx0100'	Common Event Adapter (CEA) communication unavailable.	Ensure CEA is active; Call IBM Service.	CEAUNAVAIL
X'xxxx0117'	Instrumentation is unable to accommodate additional CIM indication providers.	Remove unused/unnecessary indication provider connections from the instrumentation. Call IBM Service if this is a consistent problem.	CEAMAXCLIENTSCONNECTED
X'xxxx011F'	z/OS System Operator forced the unsubscribe of the event.	Resubscribe to the event.	CEASYSOPFORCEUNSUBSCRIBE
X'xxxx0121'	Common Event Adapter (CEA) is no longer able to communicate with CIM indication providers.	Adjust CEA by transitioning the component from minimum mode to full mode. Operator must use F CEA,MODE=FULL	CEAFORCEMINMODE
X'xxxx0126'	Instrumentation is unable to accept any more subscriptions to indication events.	Remove unused/unnecessary indication event subscriptions	CEAMAXPGMSUBSCRIBED
X'xxxx0300'	Internal CIM error.	Call IBM Service.	CEAREQUESTNOTRECOGNIZED
X'xxxx0301'	Internal CIM error.	Call IBM Service.	CEAREQUESTNOTIMPLEMENTED
X'xxxx0302'	Internal CIM error.	Call IBM Service.	CEAPROPERTYSTRUCTBADPTR
X'xxxx0303'	Internal CIM error.	Call IBM Service.	CEAPROPERTYSTRUCTBADEYE
X'xxxx0304'	Internal CIM error.	Call IBM Service.	CEAPROPERTYSTRUCTBADVERSION
X'xxxx0305'	Internal CIM error.	Call IBM Service.	CEAPROPERTYBADRESOURCE
X'xxxx0306'	Internal CIM error.	Call IBM Service.	CEAPROPERTYNOMATCH
X'xxxx0307'	Internal CIM error.	Call IBM Service.	CEAPROPERTYSTRUCTEMPTY
X'xxxx0308'	Internal CEA error.	Call IBM Service.	CEAENVBAD
X'xxxx0309'	Internal CIM error.	Call IBM Service.	CEAFILTERSTRUCTBADEYE
X'xxxx030A'	Internal CIM error.	Call IBM Service.	CEAFILTERSTRUCTBADVERSION
X'xxxx030B'	Internal CIM error.	Call IBM Service.	CEAFILTERBADRESOURCE
X'xxxx030C'	Internal CIM error.	Call IBM Service.	CEAFILTERNOMATCH
X'xxxx030D'	Internal CIM error.	Call IBM Service.	CEABADPARMPTR

Table 13. Jobs providers' reason codes (continued)

Reason code (hex)	Description	User action	IBM Service Information
X'xxxx030E'	Internal CEA error.	Call IBM Service.	CEABADSSISUBSYSTEM
X'xxxx030F'	Internal CEA error.	Call IBM Service.	CEABADSSICALL
X'xxxx0310'	Internal CEA error.	Ensure JES2/JES3 is active. Ensure that ExtendedSubsystem is available. Call IBM Service.	CEANOSSI
X'xxxx0311'	Internal CEA error.	Call IBM Service.	CEABADSSIENV
X'xxxx0312'	Internal CEA error.	Look for SDUMP. Call IBM Service.	CEAENVBADSSI
X'xxxx0314'	Internal CEA error.	Look for SDUMP. Check storage indicators (monitors). Call IBM Service if external symptom not resolved.	CEAUNABLETOALLOCATE
X'xxxx0315'	Internal CEA error.	Call IBM Service.	CEANOTJOBSTERSEELEMENT
X'xxxx0316'	Internal CEA error.	SSI Abend. Look for SDUMP. Call IBM Service.	CEAJOBCHAINBROKEN
X'xxxx0317'	Internal CEA error.	Look for SDUMP. Call IBM Service.	CEABADDATENV
X'xxxx0318'	Internal CEA error.	Look for SDUMP. Call IBM Service.	CEASYSOUTCHAINBROKEN
X'xxxx0319'	Internal CEA error.	Look for SDUMP. Call IBM Service.	CEANOTSYSOUTHRELEMENT
X'xxxx031A'	Internal CEA error.	Call IBM Service.	CEABADFREFPTR
X'xxxx031B'	Internal CEA error.	Call IBM Service.	CEABADFREFBLK
X'xxxx031C'	Internal CEA error.	Call IBM Service.	CEABADFREFENV
X'xxxx031D'	Internal CEA error.	Call IBM Service.	CEAUNABLETOFREE
X'xxxx031E'	Internal CEA error.	Call IBM Service.	CEABADIEFQRY
X'xxxx031F'	Internal CEA error.	Look for SDUMP. Call IBM Service.	CEASSCHAINBROKEN
X'xxxx0320'	Internal CEA error.	Look for SDUMP. Call IBM Service.	CEAENVBADJSQY
X'xxxx0321'	Internal CEA error.	Call IBM Service.	CEABADFILTEROPER
X'xxxx0322'	Internal CEA error.	Call IBM Service.	CEABADS54SUBSYSTEM
X'xxxx0323'	Internal CEA error.	Call IBM Service.	CEABADS54CALL
X'xxxx0324'	Internal CEA error.	SSI not activated. Call IBM Service.	CEANOS54
X'xxxx0325'	Internal CEA error.	Call IBM Service.	CEABADS54ENV
X'xxxx0327'	Internal CEA error.	Call IBM Service.	CEABADS54STOR
X'xxxx0328'	Internal CIM error.	Call IBM Service.	CEATIMEOUTMAXIMUMEXCEEDED
X'xxxx0329'	Internal CEA error.	Call IBM Service.	CEANEEDSYSOUTFILTER
X'xxxx032A'	Internal CIM error.	Call IBM Service.	CEABUFFERTOOLARGE

Table 13. Jobs providers' reason codes (continued)

Reason code (hex)	Description	User action	IBM Service Information
X'xxxx032B'	Internal CEA error.	Call IBM Service.	CEACMDSDIAGRCSET
X'xxxx032C'	Internal CEA error.	Ensure SYSREXX is active/operational using the F AXR,DISPLAY command. Call IBM Service if AXREXX appears active.	CEACMD SAXREXXRCSET
X'xxxx032D'	Client not authorized for instrumentation	Ensure user has access to instrumentation facilities.	CEANOINSTRAUTH
X'xxxx032E'	Internal CIM error.	Call IBM Service.	CEATOOMUCHDATA
X'xxxx032F'	Internal CEA error.	Call IBM Service.	CEAFILTERNOTSUPPORTED
X'xxxx0330'	Internal CEA error.	Call IBM Service.	CEAPRIMARYTYPEMISMATCH
X'xxxx0331'	Internal CEA error.	Call IBM Service.	CEABADSUBSYSTEM
X'xxxx0332'	Internal CEA error.	Call IBM Service.	CEAUNABLETOALLOCATE2
X'xxxx0333'	Internal CEA error.	Call IBM Service.	CEABADBUFFER
X'xxxx0334'	Internal CIM error.	Call IBM Service.	CEATIMEOUTLESSTHANMINIMUM
X'xxxx0335'	Internal CIM error.	Call IBM Service.	CEACMDSSYNTAXERROR
X'xxxx0336'	The CIM provider request was cancelled in-process.	Retry the command request. If it does not work, call IBM Service.	CEACMDSHALTERERROR
X'xxxx0337'	Internal CIM error.	Call IBM Service.	CEACMDSUNINITERROR
X'xxxx0338'	Internal CEA error.	Call IBM Service.	CEAFILTERBADCOMBO
X'xxxx0339'	Underlying command did not complete in the time specified.	Increase timeout value in the CIM method request and retry request.	CEACMDSTIMEDOUT

Accessibility

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use software products successfully. The major accessibility features in z/OS enable users to:

- Use assistive technologies such as screen readers and screen magnifier software
- Operate specific or equivalent features using only the keyboard
- Customize display attributes such as color, contrast, and font size

Using assistive technologies

Assistive technology products, such as screen readers, function with the user interfaces found in z/OS. Consult the assistive technology documentation for specific information when using such products to access z/OS interfaces.

Keyboard navigation of the user interface

Users can access z/OS user interfaces using TSO/E or ISPF. Refer to *z/OS TSO/E Primer*, *z/OS TSO/E User's Guide*, and *z/OS ISPF User's Guide Vol I* for information about accessing TSO/E and ISPF interfaces. These guides describe how to use TSO/E and ISPF, including the use of keyboard shortcuts or function keys (PF keys). Each guide includes the default settings for the PF keys and explains how to modify their functions.

z/OS information

z/OS information is accessible using screen readers with the BookServer/Library Server versions of z/OS books in the Internet library at:

<http://www.ibm.com/systems/z/os/zos/bkserv/>

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

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

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

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

IBM Corporation
Mail Station P300
2455 South Road
Poughkeepsie New York 12601-5400
U.S.A.

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

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS"

WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Programming Interface Information

This book is intended to help the customer to use the Common Information Model to write system management applications for z/OS systems.

The book also documents intended Programming Interfaces that allow the customer to write programs to obtain the services of CIM.

Trademarks

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

IBM
The IBM logo
ibm.com
@server
MVS
RACF
RMF
S/390
System z
z/OS
zSeries

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel, Intel Inside (logos), MMX and Pentium are trademarks of Intel Corporation in the United States, other countries, or both.

Other company, product, and service names may be trademarks or service marks of others.

Terms and conditions for downloading and printing publications

Permissions for the use of the publications you have selected for download are granted subject to the following terms and conditions and your indication of acceptance thereof.

Personal Use: You may reproduce these Publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative works of these Publications, or any portion thereof, without the express consent of IBM.

Commercial Use: You may reproduce, distribute and display these Publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these Publications, or reproduce, distribute or display these Publications or any portion thereof outside your enterprise, without the express consent of IBM.

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the Publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the Publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations. IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

All material copyrighted by IBM Corporation.

By downloading or printing a publication from this site, you have indicated your agreement with these terms and conditions.

Policy for unsupported hardware

Various z/OS elements, such as JES, DFSMS, HCD and MVS, contain code that supports specific hardware servers or devices. In some cases this device-related element support remains in the product, even after the hardware devices pass their announced 'end of support' date. z/OS may continue to service element code; however, it will not provide service related to unsupported hardware devices. Software problems related to these devices will not be accepted for service, and current service activity will cease if a problem is determined to be associated with 'out-of-support' devices. In such cases, fixes will not be issued.

Index

Special characters

@server CIM 3

A

accessibility 207
ARM
 security 49
ARM support of the CIM server 48
ASCII 75, 186
ASCII-EBCDIC conversion 75
association classes 66, 67, 79, 82
association provider functions 185
associations 79
AT-TLS 27
authentication 11, 26
 basic 26
authorization 11, 23
automatic restart 48
automatic restart using ARM 48

B

backup the CIM server configuration 48
Base classes 79
 OS management 83, 89
BaseBoard class 79
basic authentication 26
BPX.DAEMON 25
BPX.SRV 23

C

CDT 22
CDTINFO 22
CEA (Common Event Adapter) 203
CFZCIM 34
 start 40
 started task 39
 stop 40
CFZRCUST 16
CIM ix, 1
 introduction 1
 user access 26
CIM client application 2
CIM Event Model 4
CIM indication provider 6
CIM indications
 hierarchy 6
 introduction 4
 subscriptions 8
CIM listener 6
CIM operations 82
CIM Operations over HTTP 2
CIM Query Language 5
CIM server 2
 automatic restart 48
 configuration 41
 configure logging 45

CIM server (*continued*)
 configure tracing 44
 customizing 34
 how to work with 39
 setting up 21
 setting up security 21
 starting and stopping 39
CIM server configuration
 backup 48
CIM server runtime environment
 security 11
CIM_AlertIndication 8
CIM_ClassIndication 7
CIM_ComputerSystem 84
CIM_ComputerSystemPackage
 association 91
CIM_EthernetPort 99
CIM_HostedFileSystem 96
CIM_Indication 6
 properties 7
CIM_IndicationFilter 5
 properties 9
CIM_IndicationSubscription 5
 properties 10
CIM_InstIndication 6, 8
CIM_InstModification 8
CIM_IPProtocolEndpoint 100
CIM_ListenerDestinationCIMXML 5, 6, 8
 properties 10
CIM_LocalFileSystem 96
CIM_LogicalDisk 94
CIM_OperatingSystem 85
CIM_OSProcess 88
CIM_PortImplementsEndpoint 100
CIM_Process 86
CIM_ProcessClassIndication 7
CIM_ProcessIndication 8
CIM_Processor 92
CIM_RemoteFileSystem 97
CIM_RunningOS 86
CIM_SNMPTrapIndication 8
CIM_SystemDevice association 92, 101
cimcli 59
cimconfig 55
 options 42
CIMIVP
 sample client application 74
cimmof 53
cimmofl 53
cimprovider 57
CIMSERV 26
CIMSERV profile 22
cimserver command 39
cimserver.env 34, 35
cimsub 69
class descriptor table 22
class SURROGAT 23
client application 2
client monitoring application 74

cluster classes
 OS management 125
Cluster classes 80
CMPI 181
 header files 182
CMPI provider
 data conversion ASCII/EBCDIC 186
 function signatures 184
 initialization 184
 samples 191
 security 185
command
 cimconfig
 startup options 42
 cimserver 39
command-line utilities 52
commands 52
Common Event Adapter (CEA) 203
Common Information Model ix, 1
 introduction 1
Common Manageability Programming
 Interface 181
Communications Server 82
configuration properties
 changing current 41
 changing planned 41
configure logging 45
configure tracing 44
CQL 5
customizing
 CFZCIM 34
 CIM server 34
 profile.add 34
 started task 34
 UNIX System Services shell 34

D

daemon 42
designated user context 37, 185
designated user ID 37, 185
disability 207
Distributed Management Task Force 1
DMTF 1
dynamic CDT 22

E

EBCDIC 186
EBCDIC conversion 75
enableAssociationTraversal 43
enableAuditLog 43
enableCFZAPPLID 43
enableHttpConnection 42
enableHttpsConnection 42
enableIndicationService 43
enableRemotePrivilegedUserAccess 43
Enhanced ASCII 75
Enhanced Security model 13, 23
EnumerateInstanceNames 82

- EnumerateInstances 82
- environment variables 35
- error messages 75
- event 4
 - subscription 4
- Event Model 4
- event provider function 185
- extattr 25

F

- file system classes
 - OS management 96
- File System classes 79
- filter condition 4
- forceProviderProcesses 43

G

- GetInstance 82

H

- header files
 - CMPI 182
- HTTP 26
- httpPort 42
- HTTPS 26, 27
- httpsPort 42

I

- IBM_BaseBoard 89
- IBMzOS_BaseBoard 90
- IBMzOS_CFRMCoupleDataset 151
- IBMzOS_CFRMPolicy 151
- IBMzOS_CFStrDependsOn
 - association 152
- IBMzOS_CFStructure 134
 - indications 141
- IBMzOS_CFStructureConnector 141
 - indications 145
- IBMzOS_CollectionOfCFs
 - indications 153
- IBMzOS_CollectionOfCFs
 - association 153
- IBMzOS_CollectionOfSysplexNodes
 - indications 153
- IBMzOS_CollectionOfSysplexNodes
 - association 153
- IBMzOS_ComputerSystem 84
- IBMzOS_CoupleDataset 148
- IBMzOS_CouplingFacility 131
 - indications 133
- IBMzOS_CouplingFunction 145
- IBMzOS_CSBaseBoard 91
- IBMzOS_EthernetPort 100
- IBMzOS_HostedCFStrConnectors
 - association 154
- IBMzOS_HostedCFStructureConnectors
 - indications 154
- IBMzOS_HostedCFStructures
 - indications 154
- IBMzOS_HostedCFStructures
 - association 153

- IBMzOS_IPProtocolEndpoint 100
- IBMzOS_JES2Job 101
- IBMzOS_JES2SysoutDataset 120
- IBMzOS_JES3Job 112
- IBMzOS_JES3SysoutDataset 123
- IBMzOS_Job 101
- IBMzOS_JobsManagementSettings 124
- IBMzOS_LogicalDisk 95
- IBMzOS_LogicalDiskDevice
 - association 95
- IBMzOS_NFS 97
- IBMzOS_OperatingSystem 85
- IBMzOS_OSProcess 88
- IBMzOS_Process 86
- IBMzOS_RunningOS 86
- IBMzOS_SFMAAttributes 127
- IBMzOS_Subsystem 123
- IBMzOS_SubsystemJES2Jobs
 - association 124
- IBMzOS_SubsystemJES3Jobs
 - association 124
- IBMzOS_SysoutDataset 120
- IBMzOS_Sysplex 125
 - indications 128
 - methods 127
- IBMzOS_SysplexCoupleDataset 151
- IBMzOS_SysplexNode 129
 - indications 131
- IBMzOS_UnixLocalFileSystem 97
- IBMzOS_UnixProcess 87
- IBMzOS_UsesCF association 154
- IBMzOS_UsesCFRMCoupleDatasets
 - association 154
- IBMzOS_UsesCFRMPolicies
 - association 154
- IBMzOS_UsesCouplingFunctions
 - association 155
- IBMzOS_UsesJES2SysoutDatasets
 - association 124
- IBMzOS_UsesJES3SysoutDatasets
 - association 124
- IBMzOS_UsesSysplexCoupleDatasets
 - association 155
- inband view 89, 90
- indication
 - consuming 4
 - creation 4
 - definition 4
 - filter condition 4
 - processing 4
 - subscription 4
- indication consumer 4
- indication listener 4
- indication provider 4
- indication provider function 185
- indication stream 8, 9
- installation 15
- installation directories 15
- instance provider
 - functions 184

J

- job classes
 - OS management 101
- Job classes 80

K

- keyboard 207

L

- LIBPATH 35
- logging 45, 157
- logical disk classes
 - OS management 93
- logLevel 43

M

- managed object format 52
- management instrumentation
 - for additional z/OS resources 181
- messageDir 43
- messages 158
- method provider
 - functions 185
- MOF 52
- MOF file
 - provider registration 37
- monitoring provider 35
- must-stay-clean 24

N

- network classes
 - OS management 98
- Network classes 80

O

- Open Group 2
- OpenPegasus 2
- OS (operating system) 2
- OS management Base classes 79, 83
- OS management BaseBoard class 79, 89
- OS management cluster classes 125
- OS management Cluster classes 80
- OS management file system classes 96
- OS management File System classes 79
- OS management job classes 101
- OS management Job classes 80
- OS management logical disk classes 93
- OS management network classes 98
- OS management Network classes 80
- OS management Processor class 79
- OS management processor classes 91
- OSBASE_TRACE
 - trace level 35
- OSBASE_TRACE_FILE 35
- Out-of-Process 40
 - tracing providers 179

P

- PATH 35
- PEGASUS_HOME 35
- PG_Provider 187
- PG_ProviderCapabilities 190
- PG_ProviderModule 188
- PreviousInstance 8

- priority for z/OS CIM 36
- privileged user 23
- problem determination 74
- Processor class 79
- processor classes
 - OS management 91
- profile
 - CIMSERV 22, 26
- profile.add 34
- program control 25
- provider 2
 - registering 186
- Provider Agent 40, 41
- provider registration MOF file 37
- provider registration processing 191
- provider registration schema 187
- providerDir 43
- providers
 - CMPI 181

R

- RACF 21
- registering
 - indication provider 5
- registering a provider 186
- repository 44
 - maintaining 73
- repositoryDir 44
- repositoryIsDefaultInstanceProvider 44
- Resource Access Control Facility 21
- RMF monitoring provider 35
- RMF_CIM_BENCH 36
- RMF_CIM_HOST 35
- RMF_CIM_PORT 35
- RMF_CIM_TRACE 35
- RMF_CIM_TRACE_FILE 36
- RMF_INDICATION_POLLING 36
- RMF_INDICATION_RESTTIME 36
- RunAuthorized 40
- runtime environment security
 - CIM server 11

S

- sample client application 74
- SBLIM 191
- security 11
 - setting up for CIM server 21
 - using ARM 49
- service class for CIM priority 36
- setting up
 - CIM server 21
- setting up security
 - CIM server 21
- setup
 - z/OS based authorization 23
- ShareAs 41
- shortcut keys 207
- shutdownTimeout 44
- slp 44
- socketWriteTimeout 44
- SourceInstance 8
- started task
 - CFZCIM 39
- starting the CIM server 39

- STDERR 34
- STDOUT 34
- stopping the CIM server 39
- subscription 4
- SURROGAT class 23
- syslog 45
- system logger 45

T

- trace file 35
- traceComponents 44
- traceFilePath 44
- traceLevel 44
- tracing 44
- tracing providers 179
- troubleshooting 74

U

- user access to CIM 26
- user context 37, 185

W

- WBEM ix, 1
- wbemexec 53
- Web Base Enterprise Management ix
- Web Based Enterprise Management 1
- WLM service class for CIM priority 36

Z

- z/OS based authorization
 - setup 23

Readers' Comments — We'd Like to Hear from You

z/OS

Common Information Model User's Guide

Version 1 Release 10

Publication No. SC33-7998-05

We appreciate your comments about this publication. Please comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this book. The comments you send should pertain to only the information in this manual or product and the way in which the information is presented.

For technical questions and information about products and prices, please contact your IBM branch office, your IBM business partner, or your authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you. IBM or any other organizations will only use the personal information that you supply to contact you about the issues that you state on this form.

Comments:

Thank you for your support.

Submit your comments using one of these channels:

- Send your comments to the address on the reverse side of this form.
- Send a fax to the following number: FAX (Germany): 07031+16-3456
FAX (Other Countries): (+49)+7031-16-3456
- Send your comments via e-mail to: s390id@de.ibm.com

If you would like a response from IBM, please fill in the following information:

Name

Address

Company or Organization

Phone No.

E-mail address



Cut or Fold
Along Line

Fold and Tape

Please do not staple

Fold and Tape



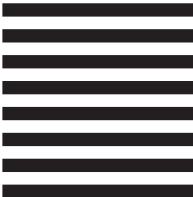
NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

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



Fold and Tape

Please do not staple

Fold and Tape

Cut or Fold
Along Line



Program Number: 5694-A01

Printed in USA

SC33-7998-05



Spine information:



z/OS

z/OS V1R10.0 Common Information Model
User's Guide

Version 1
Release 10

SC33-7998-05