Oracle Communications IP Service Activator™
Version 5.2.4

OSS Integration Manager Guide

Third Edition
December 2008

Contents

Preface

About this document

The *OSS Integration Manager Guide* is for designers writing interfaces to Service Activator. It provides details of the API, including command syntax and detailed examples.

It consists of the following chapters:

- Chapter 1: *Introduction to the OSS Integration Manager* provides a brief introduction to the OIM including installing and running.
- Chapter 2: *The Command Language* provides details of the format and syntax of all the commands available within OIM.
- Chapter 3: *The External Object Model* provides reference information for all the objects that can be accessed from the OIM API including their attributes and relationships between objects.
- Chapter 4: *Examples of Using OIM* provides practical examples of using OIM to perform a number of functions within Service Activator, including creating MPLS VPNs and creating and applying rules.
- Chapter 5: *Error Handling* provides details of error messages that can be reported from OIM.
- Appendix A: *Command Grammar* provides a formal definition of the command grammar.
- Appendix B: *Command Correlation* shows the correlation of commands with the objects to which they apply.

Command syntax

This section describes the conventions used in this guide when representing the syntax of commands.

Commands appear in a contrasting monospaced typeface, for example:

```
commit mode=queue name="createVPN"
```

Variables are indicated by italicized text:

```
delete object-path
```

Optional commands or parameters are indicated by square brackets:

```
find [object-path] search-string [attributes]
```

Where a set of alternatives are available, one of which must be selected, the options are enclosed in braces and separated by the pipe character:

```
subscribe type={object-type | all}
```

Before contacting Oracle Global Customer Support (GCS)

If you have an issue or question, Oracle recommends reviewing the product documentation and articles on MetaLink in the Top Technical Documents section to see if you can find a solution. MetaLink is located at http://metalink.oracle.com.

In addition to MetaLink, product documentation can also be found on the product CDs and in the product set on Oracle E-Delivery.

Within the product documentation, the following publications may contain problem resolutions, work-arounds and troubleshooting information:

- Release Notes
- Oracle Installation and User's Guide
- README files

Contacting Oracle Global Customer Support (GCS)

You can submit, update, and review service requests (SRs) of all severities on MetaLink, which is available 24 hours a day, 7 days a week. For technical issues of an urgent nature, you may call Oracle Global Customer Support (GCS) directly.

Oracle prefers that you use MetaLink to log your SR electronically, but if you need to contact GCS by telephone regarding a new SR, a support engineer will take down the information about your technical issue and then assign the SR to a technical engineer. A technical support representative for the Oracle and/or former MetaSolv products will then contact you.

Note that logging a new SR in a language other than English is only supported during your local country business hours. Outside of your local country business hours, technical issues are supported in English only. All SRs not logged in English outside of your local country business hours will be received the next business day. In order to obtain the broadest access to skilled technical support, Oracle advises you to log new SRs in English.

Oracle GCS can be reached locally in each country. Refer to the Oracle website for the support contact information in your country. The Oracle support website is located at http://www.oracle.com/support/contact.html.

Downloading products and documentation

To download the Oracle and/or former MetaSolv products and documentation, go to the Oracle E-Delivery site, located at http://edelivery.oracle.com.

You can purchase a hard copy of Oracle product documentation on the Oracle store site, located at http://oraclestore.oracle.com.

For a complete selection of Oracle documentation, go to the Oracle documentation site, located at http://www.oracle.com/technology/documentation.

Downloading a media pack

To download a media pack from Oracle E-Delivery

1. Go to http://edelivery.oracle.com.
2. Select the appropriate language and click **Continue**.
3. Enter the appropriate **Export Validation** information, accept the license agreements and click **Continue**.
4. For **Product Pack**, select **Oracle Communications Applications**.
5. For **Platform**, select the appropriate platform for your installation.
6. Click **Go**.
7. Select the appropriate media pack and click **Continue**.
8. Click **Download** for the items you wish to download.
9. Follow the installation documentation for each component you wish to install.

Service Activator publications

The Service Activator documentation suite includes a full range of publications. Refer to the Service Activator *Release Notes* for more information.

Service Activator configuration policy online documentation

An online reference is provided which documents configuration policies that are provided with the Service Activator core. This reference is distributed with the IP Service Activator Software Development Kit.

It is accessed through:

<ServiceActivatorHome>\ipsaSDK\doc\schemaDoc\configPolicyIndex.html

Refer to the *Software Development Kit Installation and Setup Guide* for installation instructions and more information.

Chapter 1

Introduction to the OSS Integration Manager

This chapter provides a brief introduction to the OSS Integration Manager and its capabilities. It includes the following:

- A summary of what you can do with the OIM, and some security restrictions
- The OIM executables; how to install them and run them
- An introduction to the command set and the External Object Model
- An outline of connecting an OIM client to the Naming Service

What you can do with OIM

The OIM provides an API to Operational Support Systems (OSSs), enabling Service Activator to be integrated with third-party software, such as billing, monitoring and fault management systems. The OIM enables automated or programmatic control of Service Activator's service provisioning facility.

Using OIM you can:

- Browse the managed network
- Provision services
- Report on faults and other events

Browsing the managed network

A set of commands allows external systems to access part of the Service Activator object model. You can:

- Examine managed networks and VPNs
- View details of the services applied
- Check the status of managed devices or configured services
- Find objects by name or attribute

Provisioning services

The OIM allows user applications to make both physical and logical changes to the topology. You can:

- Create or discover new devices, including details of interfaces, sub-interfaces and PVCs
- Set up customers and sites, link sites to network elements and create VPNs
- Create policy rules to control Quality of Service (QoS) and to control access, and apply them to specific points in the network topology
- Create standard and MQC Per Hop Behavior (PHB) groups and apply them to specific points in the network topology in order to control queuing or traffic shaping mechanisms
- Change certain attributes associated with a system, topology or policy object
- Propagate changes to the network, that is, update all devices/interfaces with configuration changes that have been specified
- Undo specific changes where you have ownership of the changes

Event reporting

Faults and events occurring in any part of the managed network can be reported to external systems. By using the OIM in conjunction with the Event Handler, you can:

- Subscribe to specific types of events such as faults, system messages, status changes or changes in the attributes of objects
- Specify the parts of the network to be monitored, such as a specific VPN or subnet
- Deliver data in the form of SNMP traps, as CORBA calls or to Micromuse Netcool

For more detailed information about the Event Handler, see the *Event Handler Guide*.

Security considerations

Use of the OIM requires a login and password, which must be set up from the Service Activator user interface. The user's rights to view, create, and modify objects are controlled by the user group and the permissions set up on objects. For information, see the *Network Discovery and Basic Setup* guide.

In addition, to ensure system security, some actions and some parts of the object model are not available using the OIM. You cannot:

- Access individual Service Activator system components, or execute core system operation commands, such as shutdown
- Define PHB groups
- Change or alter the visibility of audit logs

Multiple client OIM access

Service Activator supports multiple OIMs running on one Service Activator system. You can connect multiple clients to each OIM instance.

Each OIM instance runs a copy of the master object model. Each client that is connected to an OIM instance can traverse and read this model concurrently with other clients. If a client performs any write operation on the model (i.e. create/ modify/delete/use/unuse type commands, as opposed to setpath/getpath/ getattributes/getparents type commands), these operations are stored in the OIM server in a transaction queue for that particular client.

At any point, the client can either issue an abort (which empties the queue) or a commit (which performs the queued operations on the model). If the client disconnects, this is treated as an abort (after an appropriate timeout).

When the queued operations begin to be applied to the model, it is locked for all other clients. Should other clients attempt to access the model, they will block. This situation resolves in one of two ways:

- The transaction is successfully applied into the model, model validation succeeds and the changes are successfully registered with the main object model in the policy server.
- One of the above steps triggers or encounters an error. The client is notified of transaction success/failure and the model re-opened to all clients.

Installing OIM

The OIM is an optional Service Activator component, which consists of the following:

- The Integration Manager. The executable (**integration_manager.exe**) is installed in the **Program** directory on Windows systems and the **rbin** directory on Solaris systems
- The OIM command line interface (CLI). This optional executable (**integration_manager.cli**) allows you to run the OIM commands interactively.
- The OIM IDL file. This file (i**ntegration_manager.idl**) defines the OIM module structure in IDL (Interface Definition Language) format. This file is located in the **Source** directory on Windows systems and the **src** directory on Solaris systems.

Note that a Component Manager must also be installed on the same host system.

> Note that if you install the OIM on a Windows platform the CLI is only installed if you select the Extra Applications component.

For full details of installation, see the *Setup Guide*.

Running the OIM

Running the Integration Manager

To specify command-line parameters to OIM you need to do the following:

- On Solaris platforms, edit the Integration Manager entry in the configuration file **cman.cfg** in the Service Activator **Config** directory.
- On Windows platform you can use the Configuration GUI tool to run Interation Manager by following the steps given below:

- Ensure that you have installed OSS Integration.
- Check the **OSS Integration Manager** checkbox in the Configuration GUI window, in the Config Parameters panel, under Component Manager Entries.
- Click **Commit to host**.

Note: For further information on Configuration GUI, please see the chapter 'Using the Configuration GUI' in the Admin Guide.

Running the command-line interface

While generally applications are written in the form of scripts, the command-line interface is provided to run OIM commands interactively, which can be useful for testing purposes.

Run the following from the **Program** directory:

```
integration_manager_cli.exe
```

The following command line parameters can be set for the CLI:

Parameter	Description
`-NoLogin`	Use this option to prevent the login prompt – useful if the CLI is to be used to run scripts.
`-ComponentName` *`filename`*	Use this option to specify the OIM component name if it is anything other than `"master_integration_manager"`.
`-ComponentLocation` *`hostname`*	The name of the host on which the OIM is installed.
`-ORBgiopMaxMsgSize` *`size`*	Maximum CORBA message size in bytes. Defaults to 4294967295. See *Maximum CORBA message size* below.

You can create a simple text file of the OIM commands, then redirect the script into the CLI by using the following syntax:

```
integration_manager_cli -NoLogin < script.txt
```

In this case, the first line of the script should be:

```
login name=user_name password=password
```

Running OIM on Solaris

- Run the ./integration_manager_cli command in the ServiceActivator BIN directory. The path to that directory is /opt/OracleCommunications/ServiceActivator/bin.
- You will be prompted for login and password. The login and password are same as your IPSA login and password.

Once you enter the login information, you will see the Interation Manager prompt.

Maximum CORBA message size

In some situations, the default CORBA message size is not large enough – for example, if an XML export is performed, or working with a large database. A message that exceeds the CORBA message size causes the OIM to fail. We recommend that any CORBA connection is initialized with the value 4294967295. This applies to both the connection between the OIM component and the policy server, and between an OIM client (such as the OIM command-line interface) and the OIM component. To initialize the OIM component and the OIM command-line interface with the required message size, use the command-line parameter:

```
-ORBgiopMaxMsgSize 4294967295
```

Any script that runs as a client of the OIM component should also initialize the CORBA connection with this recommended message size. The method used is dependent on the language in which the script is written.

Using the OIM

In order to write applications to interface to Service Activator, systems developers need the following:

- Details of the OIM command set and syntax
- Details of the Service Activator objects that can be viewed, created and modified via the API

The command set

The OIM command set comprises a number of generic commands that enable third-party systems to browse the object model and provision services. Full details are given in *The Command Language* on page 9.

OIM uses a small set of generic commands, rather than a large number of commands specific to objects. This means the command set is independent of the Service Activator object model as well as any information model used by an external

system. Changes can be made to object attributes or new objects added to the model without affecting backwards compatibility.

Any user familiar with UNIX or scripting language commands will find the OIM straightforward to use. All commands have the same basic format:

command [*object_path*] [*parameters*]

Commands that execute changes to the database and network are aggregated into transactions, which work in the same way as using transactions from the Service Activator user interface. This logical approach helps ensure straightforward scripting.

The External Object Model

Service Activator maintains an Object Model of the physical managed network. This object model is managed by the policy server and stored in the central database. It holds information about all classes of objects, including their attributes, the actions that can be performed on them and the relationships between them. The object model is also known as the common object model, to distinguish it from local data models maintained on host machines running the user interface.

The External Object Model (EOM) is a subset of the common object model, including those objects which are accessible from external systems via the API.

Like the common object model, the EOM is divided into three major categories:

- **Topology** – the Topology Model contains objects that represent the physical network, such as VPNs, devices, interfaces and PVC objects.
- **Policy** – the Policy Model contains objects for creating and applying policies and services to the network devices.
- **System** – the System Model contains objects that represent the Service Activator system components and associated system management objects.

Full details are given in *The External Object Model* on page 63.

Connecting an OIM client to the Naming Service

The following is an example of Java code to connect to the Naming Service:

```
// setting up the CORBA details
org.omg.CORBA.Object objRef =
orc.resolve_initial_references("NameService");
NamingContext nameServiceContext = NamingContextHelper.narrow(objRef);
NameComponent nc0 = new NameComponent("orchestream.com", "vendor");
```

```
NameComponent nc1 = new NameComponent("provider",    "application");
NameComponent nc2 = new NameComponent("integration_manager",
"component_type");
NameComponent nc3 = new NameComponent("master_integration_manager",
"integration_manager");


NameComponent path[] = {nc0, nc1, nc2, nc3};


org.omg.CORBA.Object sessionManagerRef =
nameServiceContext.resolve(path);
OimSessionManager sessionManager =
OimSessionManagerHelper.narrow(sessionManagerRef);


org.omg.CORBA.Object sessionRef = sessionManager.NewSession("Java");
OimSessionInstance session =
OimSessionInstanceHelper.narrow(sessionRef);
```

The complete path of the Integration Manager component in the Naming Service is:

```
"orchestream.com",  "vendor"
"provider",         "application"
"integration_manager","component_type"
"<ComponentName>",  "integration_manager"
```

Chapter 2

The Command Language

This chapter provides details of the commands available within OIM. It includes the following:

- Description of the command grammar
- A summary of the available commands
- Full details of each command, including the syntax, details of parameters and any exceptions returned

Command grammar

All commands take the following basic form:

command [*object-path*] [*attributes*]

All commands operate on objects within the EOM, identified by a unique name or a unique ID number. For full details of the objects that can be accessed, see *The External Object Model* on page 63.

A formal definition of the language grammar is given in *Command Grammar* on page 385.

Object path

The *object_path* identifies the object that the command is to operate on. Not all commands require an object path. If the command requires an object path and it is omitted then the current object is assumed as the path.

Note that the `link` and `unlink` commands operate on two objects, a parent and a child, so the paths of both objects must be specified.

The *object_path* can take one of three forms:

- An absolute object path, starting with the root object. The '/' character given at the start of the path indicates the root object. For example:

  ```
  /System:"System"/EventSubscription:"DeviceFaults"
  ```

- A relative object path from the current object. For example:

  ```
  network:"Acme"/device:"Cisco7505"
  ```

- An object's unique ID number, enclosed in square brackets. For example:

  ```
  [675]
  ```

The following symbols may be used in the path definition:

Symbol	Description
.	Current path
..	Path to parent object
/	Object separator
Object name	Set the path to the location of the specified object
Object ID	Set the path to the location of the object with the specified ID

During object creation, unique object name verification is performed within its parent object. If duplicate names are discovered, an error message similar to the following will appear:

```
CommandExecutionError.ERR_ObjectModelViolation: Object [1802], a
Interface, has multiple children of type SubInterface with the same
name: Serial1/0.100, [4129].
```

You can create multiple objects with the same name in an object model as long as they belong to different parent objects. Uniquely named folders, under a parent object, can each store objects with the same name. The stored objects can have the same name since they belong to different parent objects.

Attributes

Not all commands require attributes. When set, they take the form:

```
attribute="value"
```

Quotation marks are not strictly required unless there are spaces within the value. If necessary, multiple attributes can be specified on a single line.

Data types

Attributes entered in command lines can be one of the following data types:

Type	Description
String	A string containing any alphanumeric characters. Maximum string length is generally 127 characters; where the maximum length of a string is shorter or longer, its length is specified in this document.
Object ID	The unique ID of each object, in the form [*nnn*] where *nnn* is an unsigned integer (Note that when attributes are returned, object IDs appear as they are in integer form)
DateTime	YYYY/MM/DD HH:MM:SS The date defaults to today

Type	Description
IpAddress	*nnn.nnn.nnn.nnn* where *nnn* <= 255
Boolean [string]	True or False
Enum	String representing one member from a set of possible enumerated values. Enum stands for enumerated type. Each attribute of type Enum is described with an enumeration of valid arguments. This attribute can be entered either by its numeric value or by the corresponding string.
U32	Unsigned 32-bit integer
Bitmap	Some attributes have binary values that are additive. When several choices in a range can be set to true, then the attribute value is a binary sum that uniquely represents those several choices.

Case sensitivity

Commands and parameters are not case-sensitive, with the exception of user passwords which must be entered exactly as originally set.

Overview of the commands

OIM commands are grouped into the following modules:

- Access Control Module – commands that allow the user to securely authenticate to Service Activator
- Transaction Module – commands that aggregate and execute changes to the database and network into transactions
- Declarative Module – commands that modify, link, create, delete or otherwise change objects within the topology or policy tree
- Management Module – commands that discover and manage the network
- Navigation Module – commands that navigate or search the object model
- Utility Module – commands that facilitate the use of OIM. Commands in the Utility Module are executed immediately

Transactions

Unless specified in their individual descriptions, commands that belong to the Declarative and Management modules are held in a default transaction queue. The commands in the transactions can be executed with a `commit` command, or the queue can be removed by an `abort` command. Commands held in the current transaction queue can be listed using the `listTrans` command.

Navigation commands may be interspersed with Declarative and Management commands. Navigation commands are executed immediately.

For more details about transaction processing within Service Activator see the *Network Discovery and Basic Setup* guide.

Error reporting

Details of errors are given in *Error Handling* on page 381.

The Access Control Module

The Access Control Module commands allow the user to securely log into and authenticate to the system by providing a user name and a password. The user can exit by logging off at any time.

Users must already have been set up within Service Activator.

The Access Control Module provides the following commands:

Command	Summary
`login`	Logs the user into Service Activator
`logout`	Logs the user out of Service Activator
`getAccess`	Get the user's access rights for all object types or a specific object type

The login command

Description

Logs the user into the system, using a log-in name and password previously set up in Service Activator. You can use the `login` command in scripts to execute

commands on the OIM client without logging into it. The log in information is provided in these scripts.

Syntax

`login name=`*`username`* `password=`*`password`*

Path

None

Attributes

Attribute	Type	Description
name	String	User name
password	String	Password

Return

None

Errors

The LoginException error may be returned for the following reasons:

- The user has entered an invalid combination of username and password.
- The user has repeatedly failed to log into the system.
- Concurrent logins are not permitted.
- The password has expired.

Example

```
login name="user1" password="mypass26"
```

The changePassword command

Description

Enables the user to set or modify their own password.

Syntax:

```
changepassword
```

The user is prompted to enter a password. Any characters entered are displayed on screen in encrypted mode.

Path

not applicable

Attributes

none

Return

none

Examples

login: user_name

password: *******

/>changepassword

password: *******

confirmation password: *******

The logout command

Description

Logs the user off the system.

Syntax

```
logout
```

Path

None

Attributes

None

Return

None

Errors

None

Example

```
logout
```

The getAccess command

Description

Returns the current user's access permissions for a single object, an object type or all object types.

Syntax

`getAccess [`*`object-path`*`] [type=`*`objecttype`*`]`

Path

The object for which access rights are required. This is ignored if a `type` attribute is specified.

Attributes

Attribute	Type	Description
type	String	Object type name, or "all" to retrieve the information for all object types.

Return

The access rights for the specified object(s). The access rights are:

- read
- create (includes delete)
- link (includes unlink)
- modify
- execute

In each case the user access is True or False.

Examples

To return access rights for the current object:

```
getAccess
   read=true create=false link=false modify=false execute=false
```

To return access rights for the object with ID no 250:

```
getAccess [250]
   read=true create=true link=true modify=false execute=false
```

To return access rights for all Device objects:

```
getAccess type=Device
   Device read=true create=true link=true modify=true execute=true
```

To return access rights for all objects:

```
getAccess type=all
   Component read=true create=false link=false modify=false
   execute=false
   Options read=true create=false link=false modify=true execute=false
   Topology read=true create=false link=false modify=true
   execute=false
   etc ...
```

The Transaction Module

The Transaction Module commands allow the user to save commands in the form of transactions and control how and when the transactions are executed.

Commands in the Declarative and Management modules are grouped into transactions. Other commands are executed immediately.

The Transaction Module provides the following commands:

Command	Summary
`abort`	Stops the current transaction.
`commit`	Saves the changes made in the current transaction to the database.
`events`	Returns the changes that have occurred through transactions since the last time the `events` command was called.
`exporttransaction`	Returns the changes that have occurred through transactions since the last time the `events` command was called.
`listTrans`	Lists commands in the current transaction.

Command	Summary
`merge`	Applies changes saved in a Transaction object to the current object model.
`rollback`	Discards changes stored in a Transaction object in the object mode.
`schedule`	Schedules a Transaction object to be applied at a given date and time.

For more details about transaction processing see the *Network Discovery and Basic Setup* guide.

The abort command

Description

Stops the current transaction and discards the queued commands.

Syntax

```
abort
```

Path

None

Attributes

None

Return

None

Errors

None

Example

```
abort
```

The commit command

Description

Saves the changes within the current transaction in the object model, and sends them to the server.

A transaction can be:

- Committed immediately. The common object model is updated with the transaction's changes and any configuration changes are propagated to the network.
- Queued. The transaction is saved to the common object model's transaction store with a status of Pending. The object model changes associated with that transaction are not executed and no configuration changes are propagated to the network.
- Scheduled. The transaction is saved and will be committed at a specified future date or time.

In all cases, the `commit` command creates a TransactionEntry object.

Atomic Transactions

Atomic transactions change the model for successful execution of the transaction. Normally, all commands in a transaction must be executed successfully for the transaction as a whole to be completed i.e. if any one command in a transaction fails, the entire transaction fails.

Use atomic transactions to break a transaction into smaller 'sub-transactions' which can succeed or fail independently without causing the entire transaction to fail.

These 'sub-transactions' are grouped using the commit type `atomic`. Then entire transaction, which includes the atomic sub-transactions, is then committed using the commit type `atomicTransaction`.

Notice the error in the first command of the following example:

```
create Customre:"Bob"
create Customer:"Bob"/Site:"NOC"
commit mode=atomic
create Customer:"John Doe"
create Customer:"John Doe"/Site:"NOC"
commit mode=atomic
commit mode=atomicTransaction
```

The first command fails, causing the first atomic transaction, which attempts to create customer Bob, to fail. However, the second atomic transaction to create

customer John Doe and its NOC succeeds. The post transaction activities which apply to the second atomic transaction are still completed.

Note that if a command in a transaction fails, and it isn't in an atomic transaction, the whole transaction fails, regardless of whether or not it contained atomic transactions.

Syntax

`commit [mode=`*`commit-mode`*`] [name=`*`transaction-name`*`] [schedule=`*`datetime`*`]`

Path

None

Attributes

Attribute	Type	Description
mode	Enum	The type of commit to perform. Values are: ■ immediate ■ queue ■ schedule ■ atomic ■ atomicTransaction The default value is immediate.
name	String	The name of the TransactionEntry object that is created. Optional for an immediate commit, mandatory for a queued or scheduled commit.
schedule	DateTime	The date and time that the transaction is to be applied. Mandatory if `mode` is schedule.
waitforcompletion	Boolean	When set to True, the `commit` command returns only if the transaction's changes have been written to the database. The default is False.

Return

None

Errors

The CommandExecutionError may be returned for the following reasons:

- Commit Failure: an element of the commit transaction was not successful
- No name attribute was specified to a queued or scheduled transaction
- No schedule time was specified for a scheduled transaction

Example

To commit the current transaction:

```
commit
```

To schedule the current transaction to be executed at a specific time:

```
commit mode=schedule name="Scheduled Transaction" schedule="2002/11/15
20:00:00"
```

To queue a current transaction:

```
commit mode=queue name="createVPN"
```

The events command

Description

Returns the changes that have occurred (through transactions) since the last time the `events` command was called.

The OIM starts to record events when the `events` command is first entered. When the `events` command is next called all the recorded events are returned.

The command can be used in two ways:

- If no path is specified, the command lists the changes that have occurred since the last time the command was called.
- If a path is specified, the command lists the transactions within the identified transaction object.

Syntax

`events [`*`object-path`*`] [block=true]`

Path

The path to the TransactionEntry object. A path does not have to be specified.

Parameters

Attribute	Type	Description
block	Boolean	When set to True, if no changes have occurred since the last time the events command was called, the command pauses its execution until an event occurs.

Return

Returns the commands within the committed transaction.

Errors

- The CommandExecutionError may be returned if the object specified in the path is not a TransactionEntry object.
- The OimSystemException command may be returned if the OIM session is invalid, or if OIM is shutting down.

Examples

To return information about a committed transaction:

```
events [2037]
   create Customer:"Cust3" Id="1056" Remarks="" Reference=""
   create TransactionEntry:"20020130145347" Id="1057"
   Description="1 action" State="Committed" Username="OIM"
   Schedule="2002/01/30 14:53:47"
   link [293] [1056]
   link [2] [1057]
```

To run the events command in blocking mode, in order to pause the execution of the command until an event occurs:

```
events block=true
```

This eliminates the process of having to repeatedly enter the `events` command until an event occurs. This automatically returns details of the transaction that has occurred, for example:

```
   create Device:"C7507-1" Id=146 DeviceType="cisco 7500"
   link [120] [146]
   modify [146] Remarks="Gateway device"
```

The exporttransaction command

Description

This command outputs the list of operations part of the transaction. The output is formatted as follows for our three formats of output; normal, concrete and fault operations:

Operations:

OID: <main_object_ID> OClass: <main_object_class_type> OName: <main_object_name> Operation: <type_of_operation> PClass: <parent_object_class> PName: <parent_object_name> PId: <parent_object_id> Parameters: <Attribute1=Value1>, <Attribute2=Value2>…

Concrete Operations:

CID: <concrete_ID> CClass: <Concrete_object_class_type> Operation: <type_of_operation> PClass: PName: PId: Parameters: <Attibute1=Value1>…

Fault Operations:

FID: <object_ID> Operation: <type_of_operation> PClass: PName: PId: Parameters: <Attribute1=Value1>, <Attribute2=Value2>…

Note: PClass, PName ,and PId are only used for "link" type of operations to describe the parent object to which the main object is to be linked as a result of the link operation.

Syntax

`exporttransaction [`*`TransactionEntry-object-id`*`]`

Attributes

The "TransactionEntry" object ID for which we want to get the list of operations.

Return

List of operations for the transaction as described in the description section above.

Errors

- Transaction entry ID not specified: CommandExecutionError.ERR_ExportTransIdRequired: The command needs a transaction entry id.
- Transaction entry not found: CommandExecutionError.ERR_CannotResolveToObject: Cannot Resolve to object: [<object_id>]

Example

In the following example, the transaction that is the object of "exporttransaction" was disabling a concrete that resulted in faults being generated.

```
exporttransaction [17414]
Concrete Operations:
          CID: 17269      CClass: ConcreteGenericRule     Operation:
   Modify       PClass:      PName:  PId:    Parameters: Enabled=0
Fault Operations:
         FID: 17413     Operation: Create      PClass:        PName:
   PId:    Parameters: DefaultText=This user-created interface has had
   its creation policy concrete disabled - the interface may not exist
   on the device, MajorCode=100, MinorCode=2346, Parameters=,
   Severity=2, TimeStamp=Thu Jan 31 13:55:45 2008
        FID: 17413     Operation: Link PClass: System  PName: 2
   PId: 2  Parameters:
         FID: 17413      Operation: Link PClass: SubInterface
   PName: 17267    PId: 17267   Parameters:
```

The listTrans command

Description

Lists all the commands entered in the current transaction.

Note that if you run `listTrans` command just after committing a set of commands, `listTrans` returns nothing.

Syntax

```
listTrans
```

Path

None

Attributes

None

Return

Returns the list of commands in the format:

`command` *`object-id`* [*`object-id`*] *`parameters`*

Errors

None

Example

```
listTrans
```

returns:

```
create /policy:"policy"/domain:"d"/customer:"YDEurope"
```

The merge command

Description

Merges the content of an existing TransactionEntry object to the current local object model. For queued (Pending) transactions, this is required before the transaction can be committed.

During the merge, Service Activator tests the validity of the transaction against the local object model. If there is a conflict, Service Activator abandons the merge and reports an error – this occurs, for example, if you attempt to merge a transaction that creates an object that already exists.

Merge is a queued command, that is, it is not executed immediately like the other Transaction Module commands.

Syntax

```
merge [object-path]
```

Path

The path of the TransactionEntry object to be merged. If no path is given, the current object is assumed.

Attributes

None

Return

None

Errors

The CommandExecutionError may be returned for the following reasons:

- The object specified in the path is not a TransactionEntry object

- The transaction in the TransactionEntry object could not be merged with the local object model.

Example

To merge the transaction MyQueuedChanges with the local object model:

```
merge /System:"System"/TransactionEntry:"MyQueuedChanges"
```

The rollback command

Description

Rolls back the changes specified in a TransactionEntry object from the current object model.

When you roll back a transaction, its changes are removed from the object model and, where configuration has been installed on network devices, the configuration is removed.

The ability to roll back a committed transaction depends on whether transactions that were subsequently committed are dependent on that transaction's changes. For example, a transaction that created a VPN to which sites and interfaces have subsequently been linked cannot be rolled back. It is not necessary to have merged the content of the TransactionEntry object first.

Rollback is a queued command, that is, it is not executed immediately like the other Transaction Module commands.

Syntax

```
rollback [object-path]
```

Path

The path of the TransactionEntry object to be rolled back. If no path is given, the current object is assumed.

Attributes

None

Return

None

Errors

The CommandExecutionError may be returned for the following reasons:

- The object specified in the path is not a TransactionEntry object

- The transaction stored in the TransactionEntry object could not be rolled back in the local object model.

Example

```
rollback /System:"System"/TransactionEntry:"MyQueuedChanges"
```

The schedule command

Description

Changes a queued (Pending) TransactionEntry object into a scheduled one, or changes the scheduled time for a scheduled TransactionEntry object.

Schedule is a queued command, that is, it is not executed immediately like the other Transaction Module commands.

Syntax

```
schedule [object-path] schedule=schedule-date
```

Path

The path to the TransactionEntry object to be scheduled. If no path is given, the current object is assumed.

Attributes

Attribute	Type	Description
schedule	DateTime	The date and time when the transaction is to be applied (YYYY/MM/DD HH:MM:00) **Note:** This parameter must be given in GMT. If the time stated is in the future in local time, but in the past in GMT, the command is executed immediately. Must be in quotes.

Return

None

Errors

The CommandExecutionError may be returned for the following reasons:

- The object specified in the path is not a TransactionEntry object
- The TransactionEntry object's state is not Pending or Scheduled.

Example

```
schedule TransactionEntry:"MyQueuedChanges" schedule="2001/6/15
20:00:00"
```

The Declarative Module

The Declarative Module provides commands that operate on and potentially change objects within the policy and topology trees.

The following commands are available:

Command	Summary
PreserveMissingInterfaces	Restores to 'found' state, the interfaces which are 'not found'.
ApplyRoleAssignmentRules	Applies roles to interfaces and devices in the Network object specified.
copy	Copies the specified object.
create	Creates and names an object.
delete	Deletes the specified object.
link	Links a child object to a parent object.
unlink	Unlinks a child object from a parent object.
modify	Modifies the attributes associated with a specified object.
subscribe/ unsubscribe	Sets the system to ignore certain types of objects during the lifetime of a session.
use	Creates an instance of a global template to be used by the specified object.
unuse	Removes the instance of a global template applied on the specified object.

The PreserveMissingInterfaces command

Description

Restores to 'found' state, the interfaces which are 'not found'. Neither the GUI nor the OSS Integration Manager Interface allow you to manage a device with 'no found' interfaces. These have to either be discovered to be found or must be deleted individually. Refer to "Preserving Missing Interfaces" in the *Online Help*.

Syntax

```
preservemissinginterfaces <object_id>
```

Where

<object_id> is the identifier for the device, interface, or sub-interface.

Path

The path should be mentioned with device object_id separated by a space.

Return

None

Errors

```
CommandExecutionError.ERR_CommitFailure: Commit failed on command 1
preservemissinginterfaces
```

```
CommandExecutionError.ERR_IncorrectObjectForCommand: The command
PreserveMissingInterfaces may not be applied to a Network
```

Example

```
Policy:"Policy"/Domain:"Oracle_Corporation"/
Network:"Oracle_Corporation">ls

[3773] ParameterSetInstance:""

[3701] Device:"rot7507-4"

/Policy:"Policy"/Domain:"Oracle_Corporation"/
Network:"Oracle_Corporation">preservemissinginterfaces [3701]
```

The ApplyRoleAssignmentRules command

Description

Applies role assignment rules to devices and interfaces in the current network.

Syntax

```
ApplyRoleAssignmentRules
```

Path

None. The command is applied to the current path.

Return

None

Errors

If applied at objects other than a Network object, you will receive the following error or a similar error:

```
CommandExecutionError.ERR_IncorrectObjectForCommand: The command
ApplyRoleAssignmentRules may not be applied to a Domain.
```

Example

```
cd /Policy:"Policy"/Domain:"MyDomain"/Network:"MyDomain"

ApplyRoleAssignmentRules

commit
```

The copy command

Description

Creates a copy of the object and of its direct children, and links the newly-created object to a specified parent.

> This command can only be used with policy rules, that is, RuleClassification, RuleAccess and RulePolicing objects.

Syntax

copy *source-object-path dest-object-path*

Path

Paths for both source and destination objects must be provided. They must be separated by a space.

Attributes

None

Return

None

Errors

None

Example

To copy a rule from one point in the topology to another:

```
copy domain:"europe"/Customer/RuleClassification:"CustomerRule"
domain:"europe"/Network:"England"/RuleClassification:"NetRule"
```

The create command

Description

Creates and names a new object. If attributes can be defined they override the default attribute settings.

The object is linked to the parent object indicated in the path statement.

Syntax

`create` *`object-path`* [*`attributes`*]

Path

The path to the new object, ending with the type and name of the object to be created.

Attributes

Optionally, a list of attribute=value pairs, which override the default attributes or supply required attributes.

Return

None

Errors

The CommandExecutionError may be returned for the following reasons:

- The user does not have permission to perform this command
- The object cannot be created at this position

Example

To create a new customer called Acme Telecom in the Europe domain:

```
create Domain:"europe"/Customer:"Acme Telecom" Remarks="Important
Customer"
```

Note

You can set the attributes of the object using either using the `create` command or the `modify` command. For example:

```
create Customer:foo Remarks="A Customer" Reference="A Reference"
```

or:

```
create Customer:foo
```

```
Modify Customer:foo Remarks="A Customer" Reference="A Reference"
```

However, attributes with an access type of CreateOnly can only be set at the time the object is created. For example, if creating a virtual device:

```
create Device:"NLV2000" IsVirtual=False DeviceType:"Cisco2000"
```

`IsVirtual` is mandatory, `DeviceType` is optional. The IsVirtual attribute does not appear in the object and cannot be set later.

The delete command

Description

Removes a specified object. All children of the object are unlinked, and any orphaned children are deleted.

Syntax

`delete` *`object-path`*

Path

The path of the object to be deleted or its ID.

Attributes

None

Return

None

Errors

The CommandExecutionError may be returned for the following reasons:

- The user does not have permission to perform this command.

Examples

To delete a customer called Acme Telecom in the Europe domain:

```
delete Domain:"europe"/Customer:"Acme Telecom"
```

To delete the object with an ID of 400:

```
delete [400]
```

The link command

Description

Links two objects in a parent/child relationship. The parent and child objects to be linked are specified in the path.

Syntax

`link` *`parent-object-path child-object-path`*

Path

Paths for both parent and child objects must be provided. They must be separated by a space.

Attributes

None

Return

None

Errors

The CommandExecutionError may be returned for the following reasons:

- The object types indicated cannot be linked

- The user does not have permission to perform this command
- The child object is already a child of the parent

Example

To link a child object with ID 475 to a parent object with ID 600:

```
link [600] [475]
```

To link the site "Paris" to the customer "Acme Telecom":

```
link customer:"Acme Telecom" site:"Paris"
```

The unlink command

Description

Unlinks a specified child object from its parent object.

Syntax

`unlink` *`parent-object-path child-object-path`*

Path

Paths for both parent and child objects must be provided. They must be separated by a space.

Attributes

None

Return

None

Errors

The CommandExecutionError may be returned for the following reasons:

- The object types indicated cannot be unlinked
- The two objects indicated are not linked

Example

To unlink a child object with ID 400 from a parent object with ID 300:

```
unlink [300] [400]
```

The modify command

Description

Modifies the attributes of the specified object.

Syntax

`modify` *`object-path parameters`*

Path

Path of the object to be modified or its object ID.

Attributes

A list of attribute=value pairs, which override the default attributes.

Attributes may be listed in any order.

Note that an error in one attribute results in all attributes being rejected.

> There is no overflow checking for attributes that are of type U32. If an entered value is negative or too large, then the attribute is changed to the overflowed value.

Return

None

Errors

The CommandExecutionError may be returned for the following reasons:

- The object or attributes are read only
- The named attribute does not exist on the object

Example

To change the comment and name attributes of an object with ID 200.

```
modify [200] comment="this is a comment" name=NewName
```

The subscribe/unsubscribe commands

Description

The subscribe/unsubscribe commands can be used to ignore certain types of objects during the lifetime of a session. For example:

- They can be used when browsing through the EOM using the CLI, to reduce the amount of information displayed on screen. For example, policy elements or topology elements can be hidden.
- They can be used if you want to run `xmlExport` for specific objects only.
- They can be used when waiting for a transaction to happen (`events block=true`), but you want to report on certain objects only, such as ConcreteObjects.
- They can be used to preserve backward compatibility. Certain types can be shown or hidden that were not available in previous versions of OIM.

Subscribe/unsubscribe are queued commands.

Syntax

`subscribe type={`*`object-type`*`|all}`

`unsubscribe type={`*`object-type`*`|all}`

Attributes

Attribute	Type	Description
type	Enum	`all` to specify all objects. *object-type* to specify the object type to subscribe to/ unsubscribe from.

Return

None

Errors

None

Example

A common query is to wait for policy elements to be applied in the policy server. For example, if you commit a transaction that creates an access rule, you may want to be notified only when the concrete rule objects are created:

```
create RuleAccess:myRule
unsubscribe type=all
subscribe type=ConcreteObject
subscribe type=System
commit
events block=true
```

This makes sure that only events concerning concrete objects are reported. For more information on events command, see *The events command* on page 21.

The use command

Description

Used to indicate the use of a policy element (such as a PHB group) by a policy target (such as a device). This is achieved by creating an intermediary Instance object.

For example, a target can be configured to use a PHB group by creating a PHBGroupInstance object, which is the child of both the target object and the PHBGroup object.

At present, this command is used only for associating PHB groups and ParameterSetInstance objects with target objects.

Syntax

`use` *`target-object-path policy-element-path`* [*`attributes`*]

Path

Two paths must be provided, separated by a space:

`target-object-path` Path of the policy target object that is to use the policy element.

`policy-element-path` Path of the policy element object to be used.

Attributes

Attributes applied to the Instance object. See *The getParameters command* on page 54.

Return

None

Errors

The CommandExecutionError may be returned for the following reasons:

- One of the object types indicated cannot be a target, or cannot be applied

Example

To specify that the PHB Group "WRR" is applied to the XYZ network:

```
use network:"XYZNetwork" PHBGroup:"WRR"
```

This creates a PHBGroupInstance object, child of both the XYZNetwork network and the WRR PHB group.

The unuse command

Description

Removes the link between a policy element by a policy target by destroying the Instance object created by the `use` command.

Syntax

`unuse` *`target-object-path policy-element-path`*

Path

`target-object-path` Path of the policy target object that is using the policy element.

`policy-element-path` Path of the policy element object used.

Attributes

None

Return

None

Errors

The CommandExecutionError may be returned for the following reasons:

- The object type indicated cannot be unused

Example

To specify that the PHB Group "WRR" is no longer applied to the XYZ network:

```
unuse network:"XYZNetwork" PHBGroup:"WRR"
```

The Management Module

The Management Module provides commands for discovering and managing devices and subscribing to events.

The following commands are available:

Command	Summary
discover	Starts a device discovery operation.
manage	Changes the state of an object to Managed.
unmanage	Changes the state of an object to Unmanaged.
resetCapability	Resets the discovered capabilities for Device and Interface objects.

The discover command

Description

Begins a discovery operation. The `discover` command can be used to discover specific devices in the physical network and create new device objects to represent them, or to get the attributes of an existing device object from the physical device.

The discover command can be used in one of two ways:

- An IP address and subnet mask or a DNS name can be given and the corresponding device(s) are found and linked to the network object specified in the path.
- The path of an existing object can be given, and the object is rediscovered and its attributes updated if necessary.

The `discover` command is executed immediately rather than being queued to a transaction.

The `discover` command can be applied only to Devices and Networks (as modelled by Service Activator).

Syntax

`discover` *`object-path`* [*`parameters`*]

Path

- For a new discovery, the path or ID of the object that the discovered objects are to be linked to (must be a network object)

- For a rediscovery, the path or ID of the object to be rediscovered

Attributes

- If the path is an existing object then no parameters are required and the specified object is rediscovered.
- If the path is an existing network, then other parameters are required, and a discovery operation is performed, linking the discovered objects to the path.

Attribute	Type	Description
Type	Enum	The type of discovery to perform. Can take the following values: ■ 0 = Refresh – Performs a rediscover of all discoverable objects in the domain ■ 1 = Discover – Performs a "normal" SNMP discovery (this is the default). ■ 3 = LocalSegment – performs an SNMP discovery of the routers on the local segment. **Note:** an SNMP agent must be running on the server machine ■ 4 = GetCapabilities – Fetches the capabilities of the discoverable objects. **Note:** Does not rediscover capabilities where they have already been fetched. **Note:** GetCapabilities will only return results when applied against Network objects, not individual Devices. ■ 5 = Stop – Stops any current discovery process
DnsName	String	DNS name of the object to be discovered. Either this or IpAddress must be supplied when Type=Discover.
IpAddress	IpAddress	IP address of the object or subnet to be discovered. Either this or DnsName must be supplied when Type=Discover.
Mask	U32	The subnet mask of the object to be discovered. Defaults to 32 if the IP address is a device, 24 if it is a subnet.
SnmpProfileName	String	The SNMP Profile to use for discovery. Note: An error message is returned if you specify any of these parameters: ReadCommunity, Retries, Timeout, or SnmpVn, in the same command statement as SnmpProfileName.

Attribute	Type	Description
ReadCommunity	String	The SNMP Read community of the object(s) to be discovered. Default value is "public".
Retries	U32	The number of retries performed for each object. Default value is 2.
Timeout	U32	The time (in seconds) before a discovery attempt times out. Default value is 3.
SnmpVersion	Enum	SNMP version: ■ V1V2c = both SNMP versions (default) ■ V1 = Version 1 ■ V2 = Version 2c
AccessStyle	String	The means of accessing and authenticating with the object - required when fetching the capabilities of each object. Valid values are: ■ None – capabilities are not fetched ■ NamedUser ■ Anonymous ■ TACACS ■ SNMPv1 ■ SNMPv2c ■ SSH ■ RSA_SSH ■ passwordOnly If no AccessStyle parameter is set, then the discover is performed with the settings that were set previously.
InheritsSecurity	Boolean	If True, security settings are inherited from the network object. If False, security settings must be set for the command.

Attribute	Type	Description
WriteCommunity	String	The SNMP Write community to use when fetching capabilities. Required if AccessStyle set to NamedUser, SNMPv1, or SNMPv2c.
UserName	String	The user name for login when fetching capabilities. Required if AccessStyle set to NamedUser or SSH.
LoginPassword	String	The Login password to use when fetching capabilities. Required if AccessStyle set to NamedUser, Anonymous, SSH or passwordOnly.
EnablePassword	String	The Enable password to use when fetching capabilities. Required if AccessStyle set to NamedUser, Anonymous, or SSH.
Filename	String	Path of topology import file. Not used; file import is not currently supported.
RsaPrivateKey	String	Name of private key file. Required only when AccessStyle is RSA_SSH.

Return

None

Errors

The CommandExecutionError may be returned for the following reasons:

- Incorrect combination of attributes.
- Path specified is not a discoverable object or an object a to which a discoverable object can be linked.

Examples

To discover a device as a child object of the root network, assuming the root network ID has an object ID of 250:

```
discover [250] ipaddress=10.0.0.30
```

To discover all devices on a subnet:

```
discover [250] ipaddress=10.0.0.0
```

To discover a device and get the capabilities, using anonymous login:

```
discover [250] ipaddress=10.0.0.30 accessstyle=anonymous
loginpassword=apasswd enable password=anotherpassword
```

To discover a device, inheriting security settings from the network object:

```
discover policy:"Policy"/domain:"DDTest"/network:"DDTest"
ipaddress=192.168.27.1 accessstyle="TACACS" inheritssecurity=True
```

To rediscover an existing device with the object ID of 500:

```
discover [500]
```

To discover a device using an SNMP Profile:

```
discover ipaddress=192.168.27.1 SnmpProfileName=SNMP_profile3
```

See also *Discovering devices* on page 330, and for more information on the discovery process, see *Network Discovery and Basic Setup* guide.

The manage command

Description

Changes the state of the specified object to Managed. This command can only be applied to a Device object.

Syntax

`manage [object-path]`

Path

Path to the object to be managed or its ID.

Attributes

None

Return

None

Errors

The CommandExecutionError may be returned for the following reasons:

- The specified object is already managed.
- The object cannot be managed.

Example

To manage the device with an object ID of 3675:

```
manage [3675]
```

The unmanage command

Description

Changes the state of the specified object to Unmanaged. This command can only be applied to a Device object.

Syntax

```
unmanage [object-path]
```

Path

Path to the object to be unmanaged or its ID.

Attributes

None

Return

None

Errors

The CommandExecutionError may be returned for the following reasons:

- The specified object is already unmanaged.
- The object cannot be managed.

Example

To unmanage the current object:

```
unmanage
```

The resetCapability command

Description

Resets the capabilities discovered for a specific target. This command can only be applied to a Device or Interface object. Ensure that the device is in the unmanaged state before applying the resetCapability command.

After the command is applied, the capability of the target device and its child interfaces or the target Interface are linked to a default (NULL) capability.

Syntax

`resetCapability [object-path]`

Path

Either the path to or the ID of the Device or Interface object to be unmanaged.

Attributes

None

Return

None

Errors

The CommandExecutionError may be returned for the following reasons:

- The specified object is invalid for this command.

The Navigation Module

The Navigation Module provides commands that operate on objects derived from both the policy and topology trees. Navigation commands do not alter any objects. The commands are executed directly rather than being queued in a transaction.

The following commands are available:

Command	Summary
find	Locates objects in the EOM.
findParameters	Finds objects of a particular type that have parameters with certain values.
getAttributes	Lists the attributes of an object.
getChildren	Lists the children of an object.
getID	Returns the ID of an object.
getName	Returns the name of an object.

Command	Summary
getParameters	Returns a list of parameters that are applied to the target object.
getParents	Lists the parents of an object.
getPath	Returns the path to the current position in the EOM.
setPath	Sets the current path within the EOM.
getTargets	Returns the targets to which a ParameterSetInstance applies.
xmlExport	Retrieves a sub-tree of the EOM and outputs in XML format.

The find command

Description

Locates an object in the object model and returns its path. The match is made against the object name and/or object attributes.

The search string may contain the following wildcard tokens:

* Represents any number of characters

? Represents a single character

The search starts at the point specified by the object path and can search up or down the topology tree.

Syntax

`find [object-path] search-string [attributes]`

Path

Path to an object or its ID. This is the start point of the search. If no path is given, the current object is assumed.

Attributes

Parameters may be specified to narrow the search. Only objects matching the search string and any specified attributes are returned:

Attribute	Type	Description
FindDirection (This is the preferred 'find' parameter as the deprecated Direction parameter collides with similarly named object model attributes.)	Enum	Indicates the direction of the search. Possible values are: Parent – to search upwards through the hierarchy. Child – to search downwards through the hierarchy (this is the default).
Direction (deprecated)	*Enum*	*Indicates the direction of the search. Possible values are:* *Parent – to search upwards through the hierarchy.* *Child – to search downwards through the hierarchy (this is the default).*
Various	Various	Attribute=value pairs on which to search. Wildcards are permitted in string arguments. This attribute is optional if an object is specified.

Return

List of all the objects that meet the search criteria.

Errors

The CommandExecutionError may be returned for the following reasons:

- The MatchConditions are malformed.
- The find command could not return anything. For example, an Object Type used in the command does not exist.

Examples

The following command searches for interfaces with the name Serial0:

```
find . interface:"Serial0"
```

The following command searches up the tree from the current location for any customer objects:

```
find . Customer:"*" finddirection=parent
```

The following command searches the domain called Acme to find VPN objects starting with the string "Mari" and with an IP address starting with 212:

```
find domain:"Acme" "vpn:Mari*" direction=child ipAddress="212*"
vpnType=Mesh
```

returns:

```
[276]   vpn:"Marigold"
[923]   vpn:"Mario"
```

For more examples of using the find command, see *Finding objects* on page 329.

The findParameters command

Description

Locates an object in the object model that has parameters set to particular values and returns its path. The match is made against the object name and/or object parameters.

Parameters are applied to an object when a ParameterSetInstance object is linked to it or to a parent object from which it inherits.

The search string may contain the following wildcard tokens:

* Represents any number of characters

? Represents a single character

The search starts at the point specified by the object path and can search up or down the topology tree.

Note that if no parameters are specified then the `findParameters` command has the same effect as the `find` command.

Syntax

`findParameters [`*`object-path`*`] `*`search-string`*` [`*`parameters`*`]`

Path

Path to an object or its ID. This is the start point of the search. If no path is given, the current object is assumed.

Attributes

Parameters may be specified to narrow the search. Only objects matching the search string and any specified parameters are returned:

Attribute	Type	Description
Direction	Enum	Indicates the direction of the search. Possible values are: Parent – to search upwards through the hierarchy. Child – to search downwards through the hierarchy (this is the default).
Various	Various	Attribute=Value pairs on which to search. Wildcards are permitted in string arguments.

Return

List of all the objects that meet the search criteria.

Error

The CommandExecutionError may be returned for the following reasons:

- The MatchConditions are malformed
- The findParameters command could not return anything. For example, an object type used in the command does not exist

Example

To find all interfaces with the parameter OCH_MeasureCBQos=True

```
findParameters / Interface:"*" OCH_MeasureCBQos=True
```

The getAttributes command

Description

Lists the attributes for a specified object.

Syntax

`getAttributes [`*`object-path`*`]`

Path

Path to an object or its ID. If no path is given, the current object is assumed.

Attributes

None

Return

List of the attributes for the specified object, including their data type and access rights (r=read, c=create, w=write).

Errors

The CommandExecutionError may be returned for the following reasons:

- The object specified is not found
- You do not have permission to view the object

Example

To return the attributes for a selected EventSubscription object:

```
getAttributes /System:"System"/EventSubscription:"mySubscription"
```

returns:

```
    rcw String     Name               = "mySubscription"
    rcw String     Description        = ""
    rcw Boolean    Enable             = True
    rcw Boolean    SendPendingEvents  = False
    rcw Enum       DeliveryType       = Netcool
    rcw String     DeliveryDetails    = ""
    rcw U32        Id                 = 2
```

The getChildren command

Description

Lists all child objects that are linked to the specified object.

Syntax

`getchildren [`*`object-path`*`]`

Path

Path to an object or its ID. If no path is given, the current object is assumed.

Attributes

None

Return

List of children of the specified object, returned as a sequence of objects.

Errors

The CommandExecutionError may be returned for the following reasons:

- The object specified is not found

Example

To return the children of the current object:

```
getChildren
```

returns:

```
[697]   domain:"zeus"
[745]   domain:"diana"
[211]   domain:"Aphrodite"
```

The getID command

Description

Returns the ID of an object.

Syntax

`getid [`*`object-path`*`]`

Path

Path to the object. If no path is given, the current object is assumed.

Attributes

None

Return

The object ID number.

Errors

The CommandExecutionError may be returned for the following reasons:

- The object specified is not found

Example

To return the ID number of the VPN called Executive:

```
getID /domain:"europe"/customer:"myCustomer"/vpn:"executive"
```

returns:

```
[8359]
```

The getName command

Description

Returns the object name for the specified object.

Syntax

`getName [`*`object-path`*`]`

Path

The object path or ID. If no path is given, the current object is assumed.

Attributes

None

Return

The name of the current object.

Errors

The CommandExecutionError may be returned for the following reasons:

- The object specified is not found
- You do not have permission to view the object

Example

To return the name of the parent object:

```
getName ..
```

To return the name of the object with an ID of 8359:

```
getName [8359]
```

The name is returned in the following format:

```
vpn:"executive"
```

The getParameters command

Description

Retrieves a list of parameters that are applied to the target object.

Parameters are applied when ParameterSetInstance objects (see page 83) are linked to the object, or to an object higher in the hierarchy. The Levels attribute of the ParameterSetInstance object determines which type of object the parameters are applied to.

Syntax

```
getParameters [object-path]
```

Path

The object path or ID. If no path is given, the current object is assumed.

Attributes

None

Return

Returns the parameters that are applied to the specified target object.

Errors

The CommandExecutionError may be returned for the following reasons:

- The object specified is not found
- You do not have permission to view the object

Example

Firstly, a parameter set must be applied using the `use` command. Specific values of parameters may be set using the `use` command, or by modifying the ParameterSetInstance object created. See *The use command* on page 37 for further information.

To apply the parameter set object (ID=15) to a network object (ID=350):

```
use [350] [15] name="Instance" OCH_NetflowEnabled=True Levels=4
```

This creates a ParameterSetInstance object with the name `"Instance"` which is a child of both the Network object specified and the ParameterSet. Only the `OCH_NetflowEnabled` and `Levels` attributes are set; other attributes are set to the default value. `Levels=4` means that the parameters are applied at the interface level within the specified network.

The ParameterSetInstance object created must then be linked to a DeviceRole and an InterfaceRole. For example, assuming the newly created ParameterSetInstance object has an ID of 555:

```
link [555] [23]
```

```
link [555] [28]
```

Using `getParameters` on one of the interfaces with the correct roles shows the applied parameters:

```
getParameters [2456]
```

Could return the following:

```
r-- Boolean     OCH_MeasureCBQoS     = False
r-- Boolean     OCH_MeasureCarQoSMIB = False
r-- Boolean     OCH_MeasureMIB2Stats = False
r-- Boolean     OCH_MeasureNetflowEnabled = False
r-- U32     OCH_NetflowVersion = 1
r-- U32     OCH_NetflowAggregation = 0
r-- U32     OCH_NetflowCacheSize = 0
r-- U32     OCH_NetflowTimeoutActive = 0
r-- U32     OCH_NetflowTimeoutInactive = 0
```

The getParents command

Description

Lists all the parents of the specified object.

Syntax

`getparents [`*`object-path`*`]`

Path

Path to an object or its ID. If no path is given, the current object is assumed.

Attributes

None

Return

List of parent objects for the specified object, returned as a sequence of objects.

Errors

The CommandExecutionError may be returned for the following reasons:

- The object specified is not found
- You do not have permission to view the object

Example

To list the parents of the object with an ID of 8349:

```
getParents [8349]
```

The details are returned in the following format:

```
[653]   domain:"europe"
```

The getPath command

Description

Returns the path to a given object ID or the current position in the object hierarchy.

Syntax

`getPath [`*`object-path`*`]`

Path

Object ID, or none to get the current path.

Attributes

None

Return

The current path, or the path to the given object. If an object ID was given that is in the current path or is a child of the current object, then the path returned is constructed from the current path.

If any other ID is given and there is more than one possible path to the object then a path is selected arbitrarily from the set of possible paths.

Errors

The CommandExecutionError may be returned for the following reasons:

- The object specified is not found
- You do not have permission to view the object

Example

To return the path of the current object:

```
getPath
```

The details are returned in the following format:

```
/domain:"europe"/customer:"myCustomer"/vpn:"executive"
```

The setPath command

Description

Sets the current path within the object hierarchy.

Syntax

`setPath [`*`object-path`*`]`

Path

Path to an object or its ID; may include the following symbols:

Symbol	Description
.	Current path
..	Path to parent object
/	Object separator

Symbol	Description
Object name	Set the path to the location of the specified object
Object ID	Set the path to the location of the object with the specified ID

Attributes

None

Return

The current path is returned.

Errors

The CommandExecutionError may be returned for the following reasons:

- The object specified is not found
- You do not have permission to view the object

Example

To set the path to the "Executive" VPN:

```
setPath customer:"myCustomer"/vpn:"executive"
```

returns:

```
/domain:"europe"/customer:"myCustomer"/vpn:"executive"
```

The getTargets command

Description

Retrieves the targets that a ParameterSetInstance object applies to. A ParameterSetInstance is applied to a parent object, but the parameters may be applied to other objects.

For example, if you link a ParameterSetInstance to a Network object, it would not be applied to the Network object, but to devices or interfaces that inherit from the Network object. (The usual rules of inheritance apply.) The `getTargets` command returns a list of those objects the ParameterSetInstance applies to.

Syntax

`getTargets [`*`object-path`*`]`

Path

Path to a ParameterSetInstance object or its ID.

Attributes

None

Return

Returns the targets that the ParameterSetInstance applies to.

Errors

The CommandExecutionError may be returned for the following reasons:

- The object specified is not found
- You do not have permission to view the object

Example

```
getTargets [2380]
```

returns a list of the interfaces where the ParameterSetInstance object has been applied, for example:

```
[302] Interface: "Ethernet1"
[318] Interface: "Ethernet2"
```

See also *Applying parameter sets* on page 368.

The xmlExport command

Description

Outputs the external object model from the target object in XML format. The output is formatted as follows:

<*ObjectType attribute1*="*value1*" *attribute2*="*value2*"...>

    <*ObjectType attribute1*="*value1*" *attribute2*="*value2*"...>

    <*ObjectType attribute1*="*value1*" a*ttribute2*="*value2*"...>

</*ObjectType*>

If the object has children, they are shown further indented and the </*ObjectType*> closes the definition.

If the object has no children, the tag takes the form <*ObjectType attributes*.../>

XML Standard Encoding

In releases of Service Activator prior to 4.0, the same non-standard encoding scheme was used when writing attribute values on the CORBA channel. The output of the `xmlExport` command was not compatible with the XML standard, and could not be processed correctly by off-the-shelf libraries.

Starting with release 4.0 of Service Activator, the OIM now follows the XML standard way of encoding certain characters when using the ISO-8859-1 charset, as described in the section 2.4 of the XML standard document.

The following escape characters are used starting with release 4.0:

Sequence	Character Represented
`&`	&
`&quot;`	"
`<`	<
`>`	>

To override this new behavior and preserve the pre-4.0 release encoding, use the following parameter:

```
xmlEscape=[true / false]
```

xmlExport compression

The data stream created and sent to the client by the `xmlExport` command can be compressed using the `compress` flag. This compresses the stream to BASE64 using the standard ZLIB format, with no header. Compression typically reduces the amount of memory used and the size of the data to 10% of its original size resulting in a decrease of execution time for the operation. To enable this feature, set the `compress` parameter to `true`:

```
xmlExport compress=true
```

Syntax

`xmlExport [`*`object-path`*`] [`*`xmlEscape=<boolean-value>`*`] [`*`compress=<boolean-value>`*`]`

Path

Path to an object or its ID. If no path is given, the current object is assumed.

Attributes

None

Return

Details of all objects below the target object in the hierarchy in XML format.

Errors

The CommandExecutionError may be returned for the following reasons:

- The object specified is not found
- You do not have permission to view the object

Example

```
xmlExport [400] xmlEscape=true
```

Assuming the object with an ID of 400 is a domain, the following might be returned:

```
<Domain Id="400" name"Europe">
    <Customer Id="500" Name="Cust1" Remarks="A customer">
        <Site Id="600" Name="Site1"/>
    </Customer>
    </Customer Id="700" Name="Cust2"/>
</Domain>
```

The domain (ID=400) has two children which are both customers. One customer (ID=500) has a child (a site), but the other customer (ID=700) does not.

The Utility Module

The Utility Module provides commands that assist in the operation of OIM.

The following command is available:

Command	Summary
alias	Creates aliases for commands.

The alias command

Description

Defines an alias for a particular command. This allows you to use an alternative or a short form of a command if you prefer.

Aliases apply only to the current session; they are removed when the session terminates.

Syntax

`alias [`*`alias`*`=`*`command`*`]`

If no parameters are entered, the alias command returns all current aliases.

Path

None

Attributes

Attribute	Type	Description
Alias	String	Alias to be used.
Command	String	Existing command for which you want to use an alias.

Return

None

Errors

None

Example

To set up the alias "cd" to be used as an alternative to the `setpath` command:

```
alias cd=setpath
```

To list all current aliases:

```
alias
```

Chapter 3

The External Object Model

This chapter provides details of the External Object Model (EOM). It includes the following:

- General concepts of the EOM
- An explanation of the notation used to describe objects
- Full details of all the objects that can be accessed from the OIM API including their attributes and relationships between objects

General concepts

Purpose of the External Object Model

The EOM is a simplified version of Service Activator's internal object model. It defines all the objects that can be accessed or updated by external applications, including their attributes and the relationships between them. The EOM is a subset of the object model, allowing user programs to create and access data objects without requiring knowledge of the underlying complexity of the entire object model.

Overall structure

The EOM is divided into three major categories:

- **Policy** – The Policy Model contains objects for defining QoS and security in terms of rules and general QoS mechanisms. It is used in conjunction with the Topology Model to apply QoS and security policies to actual devices in a real network.
- **Topology** – The Topology Model contains objects that represent the network topology of the actual managed network, such as VPNs, devices, interfaces and VC objects.
- **System** – The System Model contains objects that represent the Service Activator system components and associated system management objects.

Object notation

Objects within the EOM are described under the following headings:

- **Description** – an explanation of the object.
- **Object diagram** – showing the relationships between objects.
- **Attributes** – the name, data type, default value and access type of each attribute of the object, plus any additional explanatory remarks. Note that attribute names must not include spaces, for example, PacketMarkingName. Case is not significant and is used in this document for clarity only.
- **Inheritance** – a definition of the attributes that are inherited from other objects (see *Object inheritance and abstract objects* on page 68).

Access types

The Access Type of each attribute is one of the following:

Type	Meaning
RO	Read Only.
RW	Read/Write.
RC	Read/Modify on create only.
WO	The attribute is masked but can be edited (for example, passwords).
CO	Can be set on object creation only. The attribute will not appear on the object.

Data types

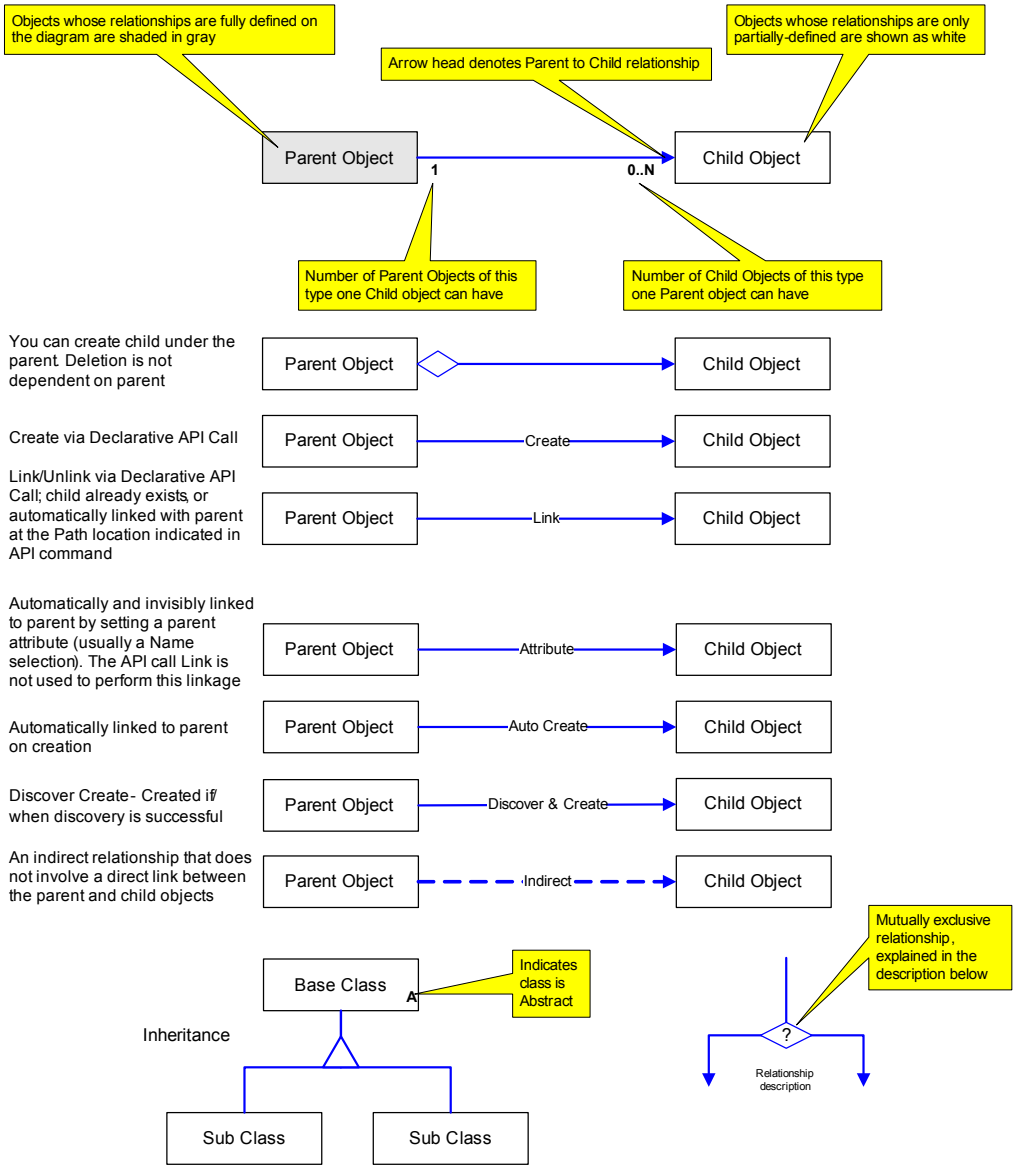
The Type of each attribute is one of the following:

Type	Meaning
U32	Unsigned 32-bit integer. In some cases the value may be restricted to a smaller range than 32 bits. In these cases the range is noted in the text.
String	A string, containing any alphanumeric characters, generally less than 127 characters. In some cases the maximum length is shorter or longer, and is detailed in the relevant parts of this document.
Enum	A string representing one member from a set of enumerated values.
Boolean	Boolean, can be True or False.

Type	Meaning
IPAddress	IPv4 address or subnet mask, in the format *nnn.nnn.nnn.nnn* where nnn <=255. IPv6 address is eight groups of four hexadecimal digits (for example, 2001:0db8:85a3:08d3:1319:8a2e:0370:7334)
DateTime	Date and time, in the format YYYY/MM/DD HH:MM:SS. All times in OIM are displayed and set as GMT.

Key to object diagrams

Relationships between objects are represented diagrammatically. The following diagram shows the standard way in which information is represented:



Linking by attribute

To simplify the manipulation of objects, the EOM has the ability in certain cases to link objects by attribute. This means that an attribute of an EOM object may actually be the name of another child object, and modifying that name will unlink the current child and link a new child with the specified name. This actual link is hidden from the user, and is performed automatically.

If such an attribute is modified then the attribute must be changed to the name of an existing object of the correct type. The system will locate the new object to be linked by searching, usually the domain or the policy object.

In most cases the attribute may be modified to an empty string. The child object will then be unlinked, and no replacement object will be linked.

The descriptions of the EOM object indicate which attributes (if any) automatically link to other objects.

Object inheritance and abstract objects

To simplify the representation of data, a concept of abstract objects is used. Abstract objects hold common attributes that are inherited by all child objects.

Abstract objects cannot themselves be accessed via the OIM command set.

The abstract objects are as follows:

Object	Purpose
Object	Inherited by all EOM objects; common attributes (Name and ID).
Traffic	Inherited by all traffic type objects (**TrafficGroup**, **TrafficCompound**, **TrafficMime**, **TrafficPort**, **TrafficURL**, **TrafficApplication**, **TrafficSubApplication**, **TrafficPacketMarking** or **TrafficDomainName**).
Rule	Inherited by all rule objects (**RuleAccess, RuleClass**, **RuleGeneric** or **RulePolice**); specifies common rule attributes except RuleGeneric, which specifies a configuration policy.

Object	Purpose
VCEndpoint	Inherited by all VC endpoint objects (**VCEndpointFr** or **VCEndpointAtm**).
Role	Inherited by **RoleDevice** and **RoleInterface** objects.
ClassificationBase	Inherited by **Classification** and **ClassificationGroup** objects.
EventFilter	Inherited by **EventFilterAttributeChange** and **EventFilterFaultMask** objects.

Object reference

Description

All objects within the EOM inherit the ID attribute and the Name attribute from the abstract object.

Attributes

Attribute Name	Type	Default	Access	Explanation
Id	U32	0	RO	Unique object reference, which remains unique for the lifetime of the object.
Name	String	""	RO/RW*	Name of object.

*Access varies according to object. See details of each object.

Object inheritance

Each object inherits attributes from its parent object. This is represented using the following notation:

object.parent.[*parent*]

For example:

`RuleAccess.Rule.Object`

indicates that the RuleAccess object inherits attributes from the Rule object, which in turn inherits attributes from the abstract object.

Summary of objects

Policy model objects

Object	Purpose
Account	Represents a user, host or subnet account or an account group: a source or destination point to which rules can be applied.
Classification	Represents a classification object, the association of a source and destination IP address and a traffic between those two hosts.
ClassificationFolder	Represents a classification folder, used to contain classification objects and other classification folders
ClassificationGroup	Represents a group of classification objects.
ClassificationOrder	Represents a sequence of classification objects.
ConcreteObject	Represents the actual implementation of a policy element or VPN; automatically created when an object is applied to a point in the network.
Cos	Represents a class of service.
COSFolder	Represents a class of service folder, used to contain class of service objects and other class of service folders
Customer	Represents a customer, to which VPNs and sites are linked.
CustomerFolder	Represents a customer folder, used to contain customers and other customer folders.
Domain	Represents a domain, the logical organization for which policies and services can be defined.
DriverScript	Represents a driver script, a set of commands defined in Python, that when applied to a device, results in IOS command script being generated.
DriverScriptFolder	Represents a driver script folder, used to contain driver scripts and other driver script folders.

Object	Purpose
InterfacePolicyRegistration	Represents the interface policy registration that informs the IPSA core about a generic policy and its abilities.
PacketMarking	Defines different classes of service.
ParameterSet	Represents a set of parameters that can be given different values when applied to different objects.
ParameterSetInstance	A template object defining a set of parameters that can be applied to different objects.
Period	Identifies the time, date and or days of the week to which a rule is to apply.
PHB	Represents the application of a specific queuing mechanism to a class of service
PHBAtm	Represents the queuing mechanism of ATM traffic shaping to an interface.
PHBFrts	Represents the queuing mechanism of a Frame Relay traffic shaping to an interface.
PHBGroup	Represents a PHB group: an implementation of a specific queuing/shaping mechanism available at an interface.
PHBGroupFolder	Represents a PHB Group folder, used to contain PHB Groups and other PHB Group folders.
PHBGroupInstance	Represents a particular application of a PHB group on an object: adds the notion of order between PHB groups.
PHBGroupMqc	Represents an MQC PHB group – the application of a queuing/shaping mechanism via Cisco's Modular QoS CLI.
PHBMqc	Represents the application of a specific MQC mechanism to a class of service.
PHBPolicingAction	Defines a policing action for use with an MQC PHB group that applies policing.
PHBWred	represents the application of a WRED mechanism to an interface.

Object	Purpose
Policy	Represents the root object of the policy tree.
PtToPtL2Martini	Represents the point-to-point Layer 2 Martini VPN service
RoleDevice	Represents a given role for a device which defines what policy can be applied to the device.
RoleFolder	Represents a role folder, used to contain interface and device role objects and other role folders
RoleInterface	Represents a given role for an interface which defines what policy can be applied to the device.
RtNumber	Represents a Route Target, as used in MPLS VPNs.
RuleAccess	Represents an access rule, used to deny or permit access to the network for specific identified traffic.
RuleClassification	Represents a classification rule, used to classify, mark, and manage network traffic.
RuleGeneric object	Represents a configuration policy.
Policy Type Object	Holds information regarding the creation of RuleGeneric objects.
Policy Type Folder Object	Categorizes policy types within the Object Model.
RulePolicing	Represents a policing rule, used to police traffic associated with a certain classification.
SAAOperation	Represents the parameters used to configure an SAA operation.
SAATemplate	Represents a parent object of SAAOperation. It groups a number of SAAOperation objects.
Site	Represents a site: a physical location defined by one or more devices and interfaces.
SiteFolder	Represents a folder used to contain site objects and site subfolders.
SiteHub	Represents the hub role of a site in a VPN.

Object	Purpose
SiteL2	Represents a Layer 2 Site.
StaticRoute	Represents a static route defined for a VPN site.
Tls	Represents a Transparent LAN Service (Layer 2 VPN). The Tls object is linked to a Customer, and SiteL2 objects are linked to it.
TrafficApplication	Represents an application-based traffic type.
TrafficCompound	Represents a compound traffic type: a traffic type that is a combination of two or more traffic types.
TrafficDomainName	Represents a domain-based traffic type.
TrafficGroup	Represents a traffic type group: a logical organization of traffic types into a folder-like structure.
TrafficMime	Represents a MIME-based traffic type.
TrafficPacketMarking	Represents a traffic type based on packet marking (DiffServ codepoint, IP Precedence or MPLS Experimental marking).
TrafficPort	Represents a port-based traffic type.
TrafficSubApplication	Represents a subapplication-based traffic type.
TrafficURL	Represents a URL-based traffic type.
Vpn	Represents a virtual private network, defined by a set of interfaces and/or CE routers.

Topology model objects

Object	Purpose
BgpAggregateAddress	Represents network statements that BGP will advertise for the site.
Device	Represents a device within the network: a network node that forwards IP packets, that is, a router or Layer 3 switch.
DeviceCapabilities	Represents the capabilities of a device.
DeviceType	Represents the model of a device, for example, Cisco 2500.
EigrpRedistribution	Redistribution attributes (delay, reliability, loading and mtu) from other protocols (connected, static, Bgp, Rip) into Eigrp.
EthernetVlan	Represents an Ethernet VLAN.
Interface	Represents an interface on a device.
InterfaceCapabilities object	Represents the capabilities and characteristics of an interface on the device.
SAP	Service Application Point. Provides a location to which to attach a VRF in an interface-less site.
Network	Represents a network, a logical object within a domain comprising a number of devices and, optionally, sub-networks.
OspfSummaryAddress	Represents the advertising of OSPF routes for redistribution as a summary address.
Segment	Represents the locally-connected network segment on an interface.
SubLayer	Represents one protocol sublayer of an interface.
SubInterface	Represents a sub-interface on an interface.
CreationMarkerSubInt	Represents a sub-interface with a PVC.
CreationMarkerVcFr	Represents an interface with a Frame-Relay VC endpoint.
Topology	Represents the root object of the Topology tree.

Object	Purpose
VcEndpointAtm	Represents an ATM PVC endpoint.
VcEndpointFr	Represents a Frame Relay PVC endpoint.
VlanInterface	Represents a VLAN interface.

System model objects

Object	Purpose
Component	Represents a Service Activator component: Component Manager, Policy Server, Proxy Agent, Device Driver, Event Handler, Integration Manager, System Logger, or Measurement Component.
EventCollector	Represents a monitoring place in the External Object Model; defines the objects on which faults are to be monitored.
EventFilterAttributeChange	Represents a specific filter associated with an event collector; defines a particular attribute to monitor.
EventFilterFaultMask	Represents a specific filter associated with an event collector; defines a specific fault or a type of fault to monitor.
EventSubscription	Represents an event subscription, defining the the way in which an external user subscribes to fault and event reporting.
ExternalSystem	Represents an external system or component.
Fault	Represents a fault that has been reported from a Service Activator component.
Options	Represents system-wide options.
Root	Represents the top of the tree of objects.
SnmpProfile	Represents a user-defined profile of SNMP attributes used to discover a group of devices or individual devices.
System	Represents the root object of the System tree.
SystemUser	The system user object is used to create new users and set security restrictions.
SystemUserGroup	A system user group defines the access level that its members have within Service Activator.
TransactionEntry	Represents a queued or scheduled transaction.

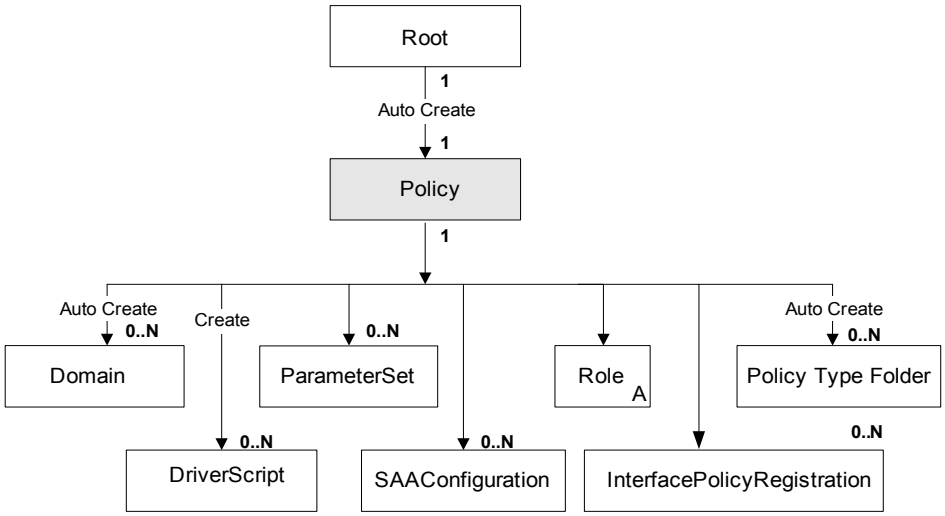
The Policy model

Policy object

Description

The Policy object represents the root of the entire policy tree.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RO	Always "Policy".

Object inheritance

Policy.Object

ParameterSet object

Description

ParameterSet objects represent a set of parameters that can be given different values when applied to different objects. ParameterSet objects define the names and data types of the parameters, but not the values.

You cannot create or modify ParameterSet objects through OIM.

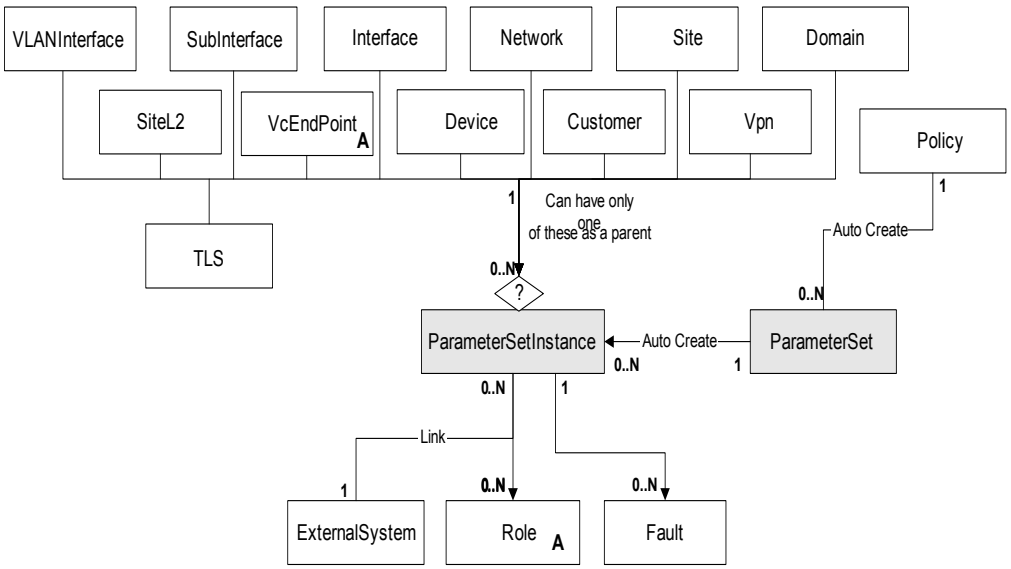
Two ParameterSet objects are created automatically as children of the Policy object. These are called CollectorParameterSet and MeasurementParameterSet.

An example showing how parameter sets are applied to configuration targets is shown on *Applying parameter sets* on page 368.

Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RO	Name of the ParameterSet.
Description	String	""	RO	Description of the ParameterSet.
Levels	U32	30	RO	The level at which the parameters apply. Bitwise value, 1 bit per target: Bit 0 = not used Bit 1 = Device Bit 2 = Interface Bit 3 = Subinterface Bit 4 = VCendpoint
Parameters	String	""	RO	The parameters that are defined by this ParameterSet. They will all have a value of "" as their values are only assigned when the ParameterSet is applied somewhere.
ExportedName	String	""	RO	Name of parameter set as sent to device driver.

Object diagram



Object inheritance

ParameterSet.Object

MeasurementParameterSet

The measurement parameters defined by this ParameterSet are as follows:

Attribute Name	Type	Default	Access	Explanation
OCH_MeasureCBQoS	String	False	RW	Specifies whether to measure class based QoS.
OCH_MeasureCarQoSMIB	String	False	RW	Specifies whether to measure CAR QoS MIB.
OCH_MeasureJuniperCoSMIB	String	False	RW	Monitor Juniper CoS MIBs
OCH_MeasureMIB2Stats	String	False	RW	Specifies whether to measure SNMP MIB2.

Attribute Name	Type	Default	Access	Explanation
OCH_NetflowEnabled	String	False	RW	Specifies whether to enable Netflow on the given object.
OCH_NetflowVersion	String	1	RW	Specifies which version of Netflow to use. 1=version 1, 2 =version 2, 3=Ag Only
OCH_NetflowAggregation	String	1	RW	Specifies which aggregation scheme to use for Netflow. 1=As, 2=Destination-prefix, 3=Prefix, 4=Protocol-port, 5=Source-prefix.
OCH_NetflowCacheSize	String	0	RW	Specifies the maximum number of entries in the cache. Range: 1024–524 288 0 = use default

Attribute Name	Type	Default	Access	Explanation
OCH_NetflowTimeoutActive	String	0	RW	Specifies the number of seconds before an inactive flow times out. Range 1–60. 0 = use default
OCH_NetflowTimeout Inactive	String	0	RW	Number of seconds flow is kept in cache. Range 10–600. 0 = use default

CollectorParameterSet

The collector parameters defined by this ParameterSet are as follows:

Attribute Name	Type	Default	Access	Explanation
OCH_CollectorName	String	""	RO	Name of collector.
OCH_CollectorType	String	""	RO	Type of collector system.
OCH_CollectorIpAddr1	String	""	RO	Primary IP address of collector system.
OCH_CollectorIpAddr2	String	""	RO	Secondary IP address of collector system.
OCH_CollectorPort1	String	""	RO	Primary port number for collector system.

Attribute Name	Type	Default	Access	Explanation
OCH_CollectorPort2	String	""	RO	Secondary port number for collector system.
OCH_CollectorURL	String	""	RO	URL for locating collector system.

ParameterSetInstance

Description

A ParameterSetInstance object represents an instance of a ParameterSet that has been applied. It is a child of the relevant ParameterSet and the object to which the parameters are applied using the `use` command (see *The use command* on page 37).

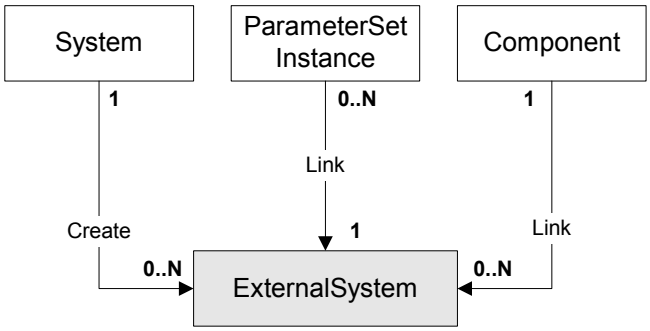
The ParameterSetInstance has the attributes defined by the ParameterSet object, but the parameters have actual values associated with them. Their types are correctly set, rather than just defined as strings, so for example, OCH_MeasureCBQoS is a Boolean value.

If the ParameterSetInstance is an instance of the MeasurementParameterSet, then the attributes of the ParameterSetInstance are modifiable by OIM.

If the ParameterSetInstance is an instance of the CollectorParameterSet, then the attributes of the ParameterSetInstance cannot be modified via OIM, but instead come directly from the ExternalSystem object that is a child of the ParameterSetInstance. The attributes of the ExternalSystem are modifiable, and modifying them also changes the parameter values of the ParameterSetInstance. For further information about ExternalSystem object, see *ExternalSystem object* on page 312.

An example showing how the parameter sets are applied to configuration targets is shown on *Applying parameter sets* on page 368.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Levels	U32	0	RW	Bitwise attribute. None = 0, VPN = 1, Device = 2, Interface = 4, SubInterface = 8, PVC = 16
Name	String	""	RO	Name of the ParameterSetInstance.
Order	U32	0	RO	The order of the ParameterSetInstance.
Parameters	Varies	Varies	RO/RW	The parameters from the ParameterSet, with values which apply to the configuration target.

Object inheritance

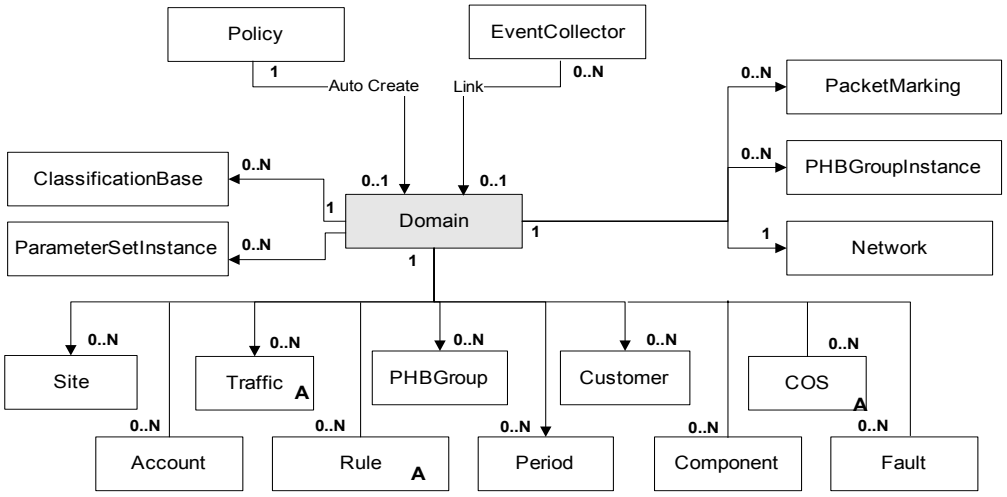
ParameterSet.Object

Domain object

Description

A policy domain is the logical organization for which policies and services can be defined, that is, all or part of a customer's network.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RO	Name of the domain.
Remarks	String	""	RW	Optional additional comments.
AsOverride	Boolean	True	RW	True = Set AS override for EBGP neighbors. False = Don't.
DomainAsOverride	Boolean	True	RW	True = Use the AsOverride value on all sites. False = Set AsOverride on each site.
AllowAsIn	U32	0	RW	Number of times the same AS can appear in the AS path list (0-10).

Attribute Name	Type	Default	Access	Explanation
VrfTableLimit	U32	0	RW	Maximum number of routes allowed in a VRF (0=No limit).
VrfTableLimit Warning	U32	75	RW	Threshold at which warning is issued. Range: 1–101, where 1–100 = percentage warning 101 = warning only
BgpMd5Key	Encrypted string	""	RW	BGP MD5 authentication key
PePeSendStandard Community	Boolean	True	RW	True = Use the PE-PE peering Standard send community tag. False = Don't.
PeCeSendStandard Community	Boolean	False	RW	True = Use the PE-CE peering Standard send community tag. False = Don't.
PeCeSendExtended Community	Boolean	False	RW	True = Use the PE-CE peering Extended send community tag. False = Don't.
LoadPolicy	String	unset	RW	Loads a policy file into the object_model.
ConfigurationPolicy ProxyAgent	String		RW	Allows you to designate a Proxy Agent that can be used to deliver specific services to a device that might be otherwise under the control of a different Proxy Agent. Example services include Configuration Policy, Sub-interface Configuration, and Virtual CE configuration. For example, you would select the Network Processor proxy to apply Configuration Policy configuration to targets under the Domain.

Attribute Name	Type	Default	Access	Explanation
UseServiceProxyForVpn	Boolean	False	RW	Assigns the Proxy Agent set by ConfigurationPolicyProxyAgent to all VPN Services - both new and existing.
UseLoopback	String	0	RW	The Loopback ID value is used to create a loopback interface name by appending it to the name 'loopback'. For example, if the Loopback ID is 0, the loopback interface name created is 'loopback0'. When a device in this domain is discovered, a check is made to see if a loopback interface matching this text string exists. If it does, the IP address of the loopback interface is stored with the device information.

Object inheritance

Domain.Object

InterfacePolicyRegistration

Description

The interface policy registration informs the IPSA core about a generic policy and its abilities, and is restricted to the interface management scope. Within this scope, the information about a policy is characterized by:

- Operation: What is the name of the policy? What does this policy do - Create or Decorate? How should this policy be represented in the GUI menus?
- Context: What kind of device does this policy apply to? At what level can this policy be made available - device, controller, interface or subinterface? If a controller, interface or subinterface is the level chosen, what kind of controller, interface or subinterface should it be?
- Creation Template: If the policy type is Create, what kind of interface or subinterface does it create? Is there any pattern that its name should follow? What kind of capabilities should that object have upon creation?

Creation capabilities apply only when the object being created is an interface. In that case, all the interface level capabilities have to be specified (linked) - interface

caps, subinterface caps and VC caps. Note that you must not modify a caps object linked to a registration. It might be in use by other interfaces - once a caps object is created under the Topology, it is immediately shareable. Instead, create a new caps object, unlink the old one and link in the new one. Or just repeat the interface linkage with another interface that has the desired caps.

A given policy may be registered many times depending upon how flexible it is. For instance, a policy may be able to create a subinterface or decorate an existing one. In such a case, you would register this policy twice - once each for Create and Decorate. Once an interface management generic policy has been registered appropriately, it is ready for use.

Once a given registration is in use, the management operations that can be performed on it are limited. This is to ensure we do not orphan the current users or create a mismatch between the current users and post-modification future users. In-use registrations cannot be deleted. They can be modified, but only certain attributes can change:

- They can be disabled. A disabled registration no longer appears on any GUI menus and cannot be used anew - the existing usages remain unimpacted.
- Their name can change.
- Their menu text can change.

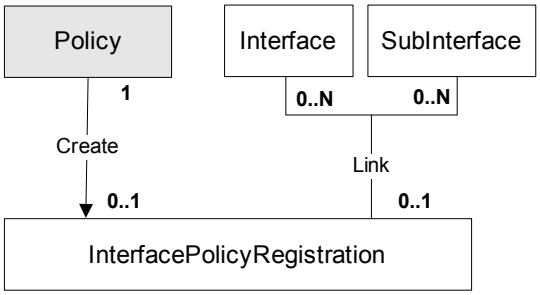
In order to create an interface (subinterface), the user must set their path to the Device (Interface). They have to then create a stub object: "create Interface:SomeName", or "create SubInterface:SomeName". Then they have to link the desired registration (child) to the stub object (parent). It is recommended they commit at this point so their EOM is updated with the new object. They have to then link the desired role (child) to the stub object. Under the stub object, they will find the generic policy instance. They have to modify its payload attributes with the appropriate XML string. Now, a commit will complete the creation operation.

To configure an object, you must simply link the desired registration (child) to the desired object (parent). It is recommended they commit at this point so their EOM is updated with the new objects. The desired object must be linked to an appropriate role. Under that desired object, they will find the generic policy instance. They have to modify its payload attributes with the appropriate XML string. Now, a commit will complete the configure operation.

To delete a created interface or subinterface, the user must simply delete the object.

To remove the configuration of a previously configured object, the user must simply unlink the applied registration from the object.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Context	Enum	Device	RW	The kind of object you will apply this registration on - either a device, controller, interface or sub-interface. 0 = "Device" 1 = "Interface" 2 = "SubInterface" 3 = "Controller"
ContextDefault	Boolean	False	RW	Context attribute is used only when ContextDefault is set to false.
ContextPattern	String		RW	Used only if the Context is Controller, Interface or SubInterface. The context matching is extended to additionally require that the context name begins with this pattern.

Attribute Name	Type	Default	Access	Explanation
ContextSnmpIfTypeList	String		RW	Comma seperated list of SNMP ifType of the interfaces or subinterfaces where this policy can be chosen. It is used only if Context is Interface or SubInterface and if ContextDefault is not set.
ContextSnmpIfType	U32	0	RW	The interface type. Valid for all Applied Contexts except for device. For example, if you are creating a sub-interface on an interface that has an ifType of "32", then the SNMP ifType is "32".
CreationPrefix	String		RW	This value pre-populates the interface properties page when you use this registered policy and allows Service Activator to validate the supplied interface or sub-interface name during creation.
CreationUseParentPrefix	Boolean	False	RW	Valid for sub-interface creation only. This value indicates whether the registered policy will get the prefix from the parent interface.
Enabled	Boolean	True	RW	Whether the registration is operational.
GenericPolicyTypeName	String		RW	Generic policy type

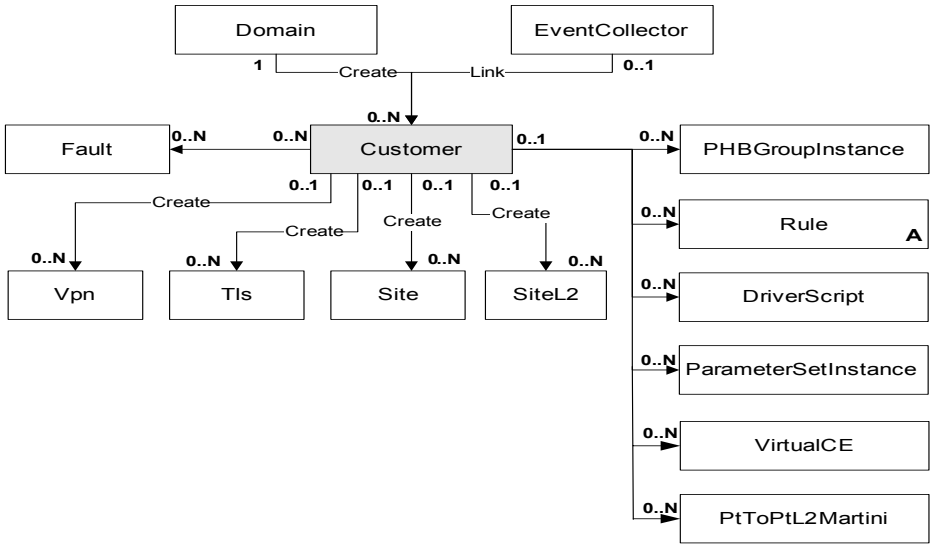
Attribute Name	Type	Default	Access	Explanation
MenuText	String		RW	GUI: The text that will display in the right-click drop-down menu at the point where you wish to apply interface management policies.
PolicyType	Enum	Create	RW	Identifies the interface policy type being registered.
SnmpIfSpeed	U32	0	RW	Default bandwidth for created interface or sub-interface in kbps. This value is only for validation purposes. The actual speed depends on the given configuration policy.
VendorEnterpriseId	U32	0	RW	A vendor-specific and series-specific ID. Refers to the kind of device you are applying the registration to. For example, the SNMP Enterprise ID for Cisco is "9" and for Juniper it is "2636".
Name	String		RO	Name for the interface policy registration

Customer object

Description

A customer object represents the concept of a customer, to which VPNs and sites are linked.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RW	Name of the customer.
Remarks	String	""	RW	Optional additional comments.
Reference	String	""	RW	Customer reference. No format is imposed.

Object inheritance

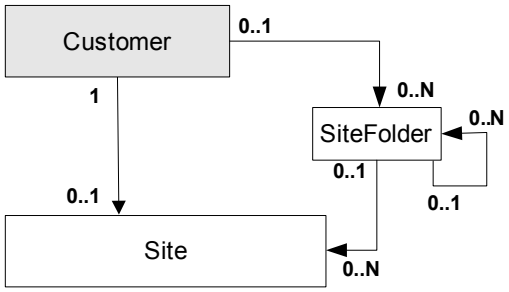
Customer.Object

SiteFolder

Description

The SiteFolder object defines a site folder used to contain site objects and site subfolders. A SiteFolder can be a child of Customer or a child of another SiteFolder object.

Object diagram



Attributes

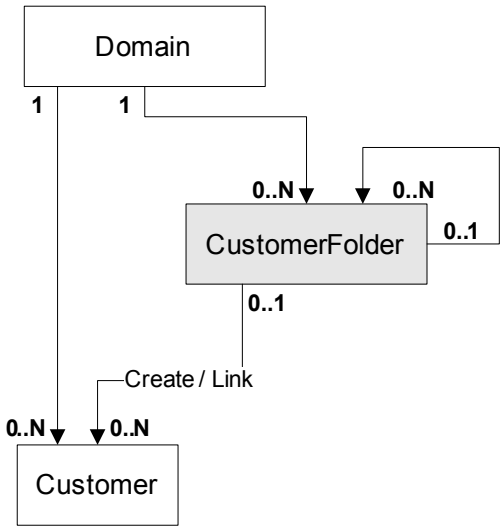
Attribute Name	Type	Default	Access	Explanation
AccountRef	String		RW	Text field to store customer reference information. Note: No inheritance is implied between sites in the site folder(s) and the Account Ref field value in the site folder.
Remarks	String		RW	Optional additional information about the site. This is a free-format text string. It is for information only and is not used by Service Activator.
Name	String		RW	

CustomerFolder object

Description

A CustomerFolder object represents a folder which contains customer objects, or other CustomerFolder objects, for purposes of organization within the GUI.

Object diagram



A Customer always has either 1 or 2 parents and is always linked to its parent Domain. It may be linked to zero or one parent CustomerFolders. In the Service Activator GUI, if a Customer has 2 parents, it will always be displayed under the CustomerFolder, not under the Customers folder. If a customer is created under a folder, it is automatically linked to its parent Domain.

A Customer folder is a child of either another Customer folder or the Domain, but not both.

Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RW	Name of the customer folder.
Remarks	String	""	RW	Optional additional comments.

Object inheritance

CustomerFolder.Object

Account object

Description

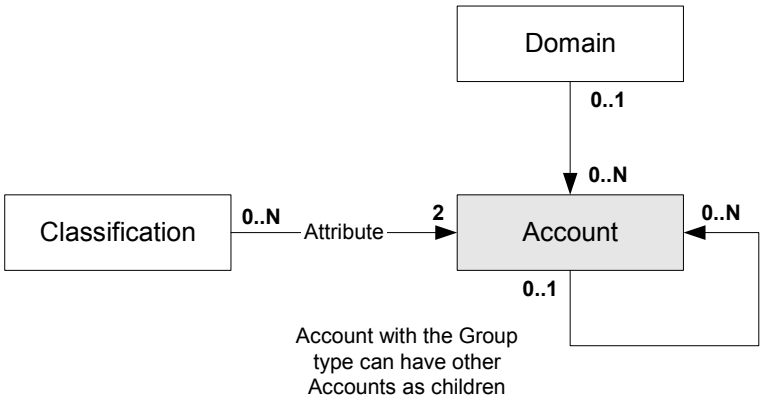
An account is a source or destination point in the network to which policy rules can be applied. An account in this context can be one of the following:

- A user, identified by a name and an IP address
- A host computer, identified by an IP address
- A subnet

Accounts can be organized into a hierarchical structure of groups for organizational purposes or to represent the structure of a company.

- Accounts must be set up via the Service Activator user interface, but existing accounts can be browsed in the EOM.
- In the EOM, Account objects are not strictly required for the correct operation of policy objects; use of Accounts is therefore limited.
- Existing accounts may be referred to by name when manipulating Rule objects.
- Rules do not require host/subnet accounts to be entered; IP addresses may be input directly.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RO	Name of the account.
Remarks	String	""	RO	Optional additional comments.
Type	String	"User"	RO	Type of account; one of: ■ **Group**: an administrative grouping ■ **User**: A named user and IP address ■ **Host**: A host by IP address ■ **Subnet**: A subnet

Object inheritance

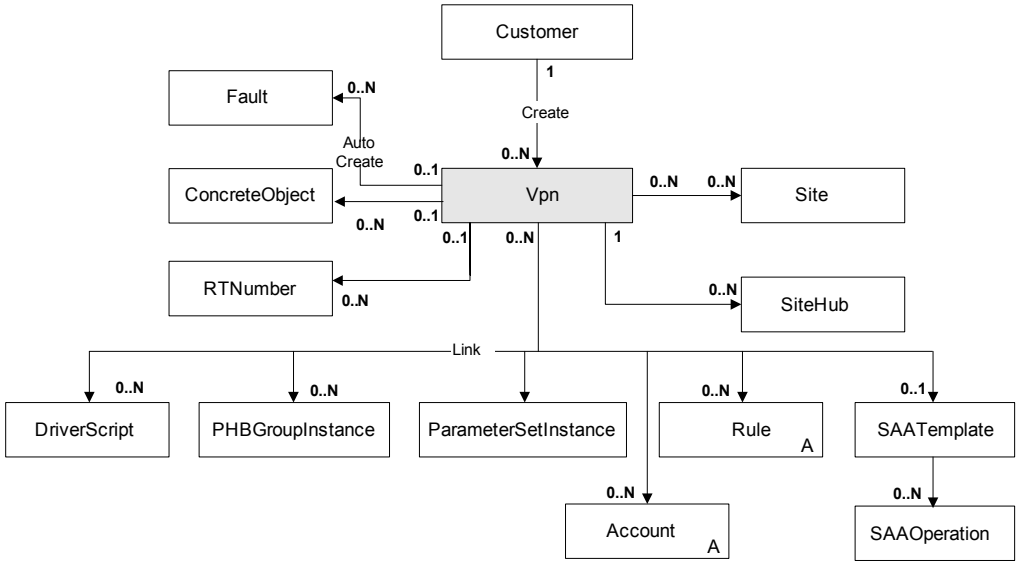
Account.Object

Vpn object

Description

A virtual private network, defined by a set of sites (PE interfaces or CE routers) at the edge of the core network cloud. A VPN is a general concept and is independent of the various technologies that may be used to provide the privacy and/or routing independence.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RW	Name of the VPN.
Description	String	""	RW	Free-format comments about the VPN.
Type	Enum	0		Type of VPN: 0 = None 1 = MplsVpn

Attribute Name	Type	Default	Access	Explanation
Level	U32	4	RW	Priority level of the VPN, in the range 0–7. This value is only used if a site is included in more than one VPN, which may result in conflict. The VPN with the lowest level number will take precedence.
MplsVpnType	Enum	0	RW	Topology of MPLS VPN: 0 = FullMesh 1 = HubAndSpoke 2 = Management
VpnId	U32	0	RO	Unique number of VPN relative to its customer.
Context	String	""	RW	Local context for driver scripts applied at network level (max 512 bytes).
UseOneRdPerVpn	Boolean	False	RW	True = Apply the same VRF table name and RD number to all sites that participate in this VPN False = Generate a site-specific VRF table name and RD number for each site that participates in the VPN
InstallDhcp	Boolean	False	RW	True = Install DHCP support on the VRFs. False = Do not install DHCP support on VRFs.
PrimaryDhcpIpAddr	IPAddress	0.0.0.0	RW	Primary DHCP Server

Attribute Name	Type	Default	Access	Explanation
SecondaryDhcpIpAddr	IPAddress	0.0.0.0	RW	Secondary DHCP Server
EbgpMd5Key	Encrypted string	""	RW	BGP MD5 authentication key ""=not used.
OspfMd5Key	Encrypted string	""	RW	OSPF MD5 authentication key ""=not used.
The following attributes only apply if **UseOneRdPerVpn** is set to True, i.e. VRF tables and RD numbers are defined once per VPN, rather than separately for each site (see also *Site objects* on page 117).				
ForceVrfInstall	Boolean	True	RW	True = VRF tables on corresponding interfaces must be installed and cannot be merged into other tables. False= VRF tables can be merged into other tables.
ShareableVrf	Boolean	False	RW	True = Other tables can be merged into this VRF table. False = Other tables cannot be merged into this VRF table.
OverrideVrf Table Limit	Boolean	False	RW	True = Use site-specific settings for VRF table limits False = Use domain defaults for VRF table limits
VrfTableLimit	U32	0	RW	Maximum number of routes allowed in a VRF (0=No limit).

Attribute Name	Type	Default	Access	Explanation
VrfTableLimit Warning	U32	0	RW	Percentage at which to warn of VRF table limits being exceeded. Range: 1–101 1–100 = percentage of VrfTableLimit reached warning. 101 = warning when VrfTableLimit reached.
VrfTableName	String	""	RW	The name of the VRF routing table.
RDHighOrder	U32	0	RW	The top 32 bits of the Route Descriptor value.
RDLowOrder	U32	0	RW	The bottom 32 bits of the Route Descriptor value.
VrfImport	U32	1	RW	The number of device redundant path configurations. Range is 2^{16} plus the default.
UseVrfImport	Boolean	True	RW	True = Use VRF import False = Do not use VRF import
EBgpMaxPaths	U32	1	RW	Allows the specification of the maximum number of parallel EBGP routes that can be installed on the device. This corresponds to the Cisco maximum-paths command. Range: 1–16.
IBgpMaxPaths	U32	4	RW	

Attribute Name	Type	Default	Access	Explanation
EIBgpMaxPaths	U32	1	RW	Allows the specification of the maximum number of parallel EBGP and IBGP routes that can be installed on the device. This corresponds to the Cisco maximum-paths eibgp command. Range: 1–16.
EigrpMaxPaths	U32	1	RW	Allows the specification of the maximum number of parallel EIGRP routes that can be installed on the device. This corresponds to the Cisco maximum-paths eigrp command. Range: 1–16 for IOS 12.3(2)T and later 12.3(T), and from 1–6 in earlier versions. The default value is 4.
EigrpMd5KeyChainRef	String		RW	Specify the key chain name to use with MD5 Authentication for EIGRP.
IBgpUnequalCost	Boolean	False	RW	Allows unequal cost load balancing by selecting iBGP paths that do not have an equal cost.
OspfMd5AreaLevelAuth	Boolean	False	RW	Enables MD5 key authentication for OSPF for the VPN.

Attribute Name	Type	Default	Access	Explanation
UseVrfLabel	Boolean	False	RW	Enables Juniper vrf-table-label support. When this field is set to true, the inner (VPN) label of a packet is removed as it arrives at a VRF so that it can be processed based on the contents of its IP header. When this field is set to false, incoming packets are mapped directly onto an outgoing (CE-facing) interface based on the inner VPN label.
VrfDesc	String		RW	Allows a VRF table route to be advertised to other PE routers only if its prefix matches one of those specified in the export map. The export map tags exported routes with the RT number of each site that needs to receive those routes. Export maps must be manually pre-configured on the PE router.
VrfExportFilter	String		RW	
VrfImportFilter	String		RW	A VRF Import Map allows the site to selectively import routes learned elsewhere.
CreateDefaultRT Numbers	Boolean	True	RO	Specifies that the RD number used for the selected interface is calculated by Service Activator.

Object inheritance

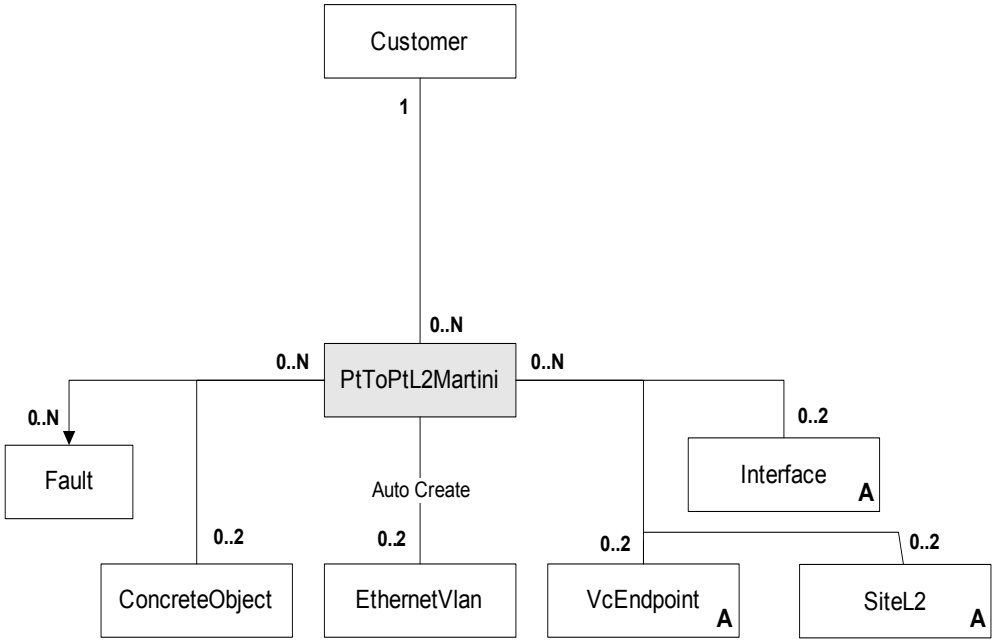
Vpn.Object

PtToPtL2Martini object

Description

A Layer 2 Martini virtual private network, defined by endpoints (PE interfaces or CE routers) at the edge of the core network cloud and encapsulating various types of data across the Martini VPN tunnel.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RW	Simple accessory to this object.
Remarks	String		RW	General comments.
ConnectionType	Enum	0	RW	The type of Martini connection to be made: ATM_AAL5 = 0 ATM_Cell = 1 Frame = 2 Ethernet = 3 Ethernet_VLAN = 4
MartiniVcId	U32	1	RW	Range: 1–0xFFFFFFFF
ActualMartiniVcId	U32	0	RO	Value actually in use, user-defined or generated
GenerateIdentifier	Boolean	False	RW	Allows the Martini site to be created if set to True. Note: When modifying bGenerateIdentifier to False you should also modify the MartiniVcId in the same transaction, as the default is '1'.

Object inheritance

L2PtToPtMartini.Object

TLS object

Description

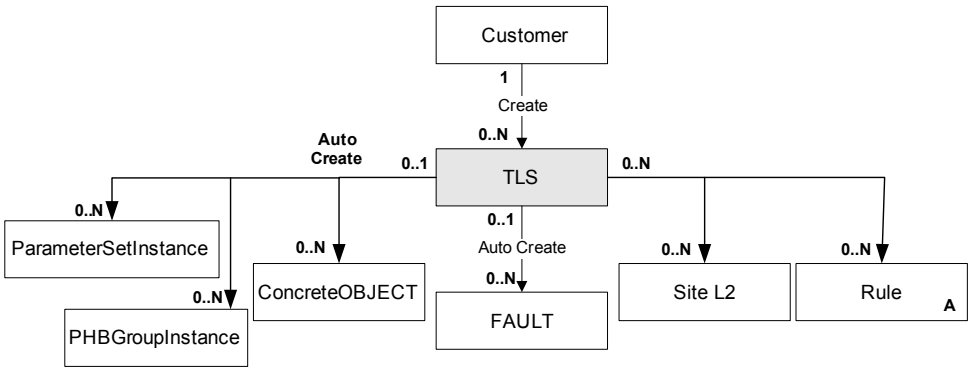
A Transparent LAN Service is a service used to connect together separate LAN segments via an MPLS cloud and make them appear as if they were forming a unique VLAN. TLS objects can be used to connect distinct physical ports or specific VLANs associated with physical ports. In the first case, the ports are known as "host ports", whereas in the second case they are called "trunk ports" (in both case, ports are a synonym for the Interface objects).

Note that policy (rules, PHB groups and driver groups) applied on the TLS or higher up in the inheritance hierarchy (for example on the customer) is not inherited through TLS objects down to the specific ports.

If a TLS has a ServiceType attribute of PortAndVlan, the range of VLAN IDs specified on the TLS will be used to set up the TLS customer profile on the PE port. The actual VLANs created on the PE port and on the MTU ports can be specified on the site itself (but must be a subset of the VLAN IDs specified on the TLS). If no VLAN IDs and no encapsulation are specified on the L2Site object, all the VLANs that are part of the TLS range will be created on the PE and MTU in the site.

The system validates that on a CE/MTU the same VLAN is not used in the context of different TLS services, to avoid cross communications between the TLSs. It is assumed that on the PE, VLANs created in the context of different PortAndVlan based TLSs will be prevented from communicating by the customer profile.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RW	Name of Layer 2 VPN.
Description	String	""	RW	Free-format description of the Layer 2 VPN.
Topology	Enum	0	RW	VPN topology. At present, only Full Mesh is supported. 0 = FullMesh 1 = HubAndSpoke 2 = Management
ServiceType	Enum	0	RW	Defines how access to the TLS is managed 0 = Port Based 1 = PortAndVlanBased
ProfileVlanIds	String	""	RW	If ServiceType = 1, specifies which VLANs should be made part of the customer profile. Can be a single VLAN ID, or several comma- separated IDs, and can include ranges (e.g. 4, 10-15)
PortsConfiguration	Enum	0	RW	If ServiceType = 0, all PE ports in all sites share the same configuration: 0 = OnlyUntaggedFrames Accept only untagged frames (access port) 1 = OnlyTaggedFrames Accept only pre-tagged frames (trunk port)
Conflict	Boolean	False	RO	True = TLS is failing. False = TLS is not in error.
GenerateVCId	Boolean	True	RW	autogeneration of the VC ID.

Attribute Name	Type	Default	Access	Explanation
Stacked	Enum	Unstacked	RW	To select a Stacked VLAN or Unstacked. Use Stacked when the packets coming into the site from the CE are already tagged with a customer-specific VLAN ID. An additional tag is added as the packets move out of the site to the PE. Valid entries are: Stacked, Unstacked.
StackedVlanTag	U32	0	RW	Provide the VLAN ID(s) that packets are to be tagged with.

Object inheritance

Tls.Object

SAATemplate object

Description

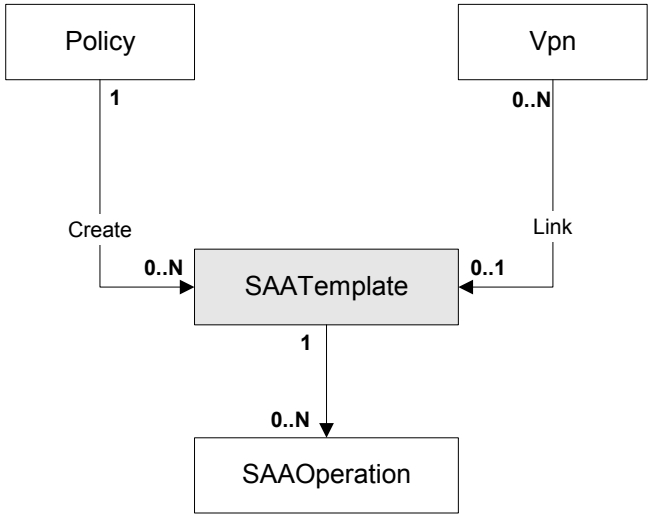
An SAATemplate object is a parent object which groups a number of SAAOperation objects.

Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RW	Name of the SAA template.
Remarks	String	""	RW	Optional additional comments.
DeviceBits	U32 [6-11]	10	RW	Defines how many devices can be configured per VPN ID (allowing up to 1024 devices per SAA VPN).
TosBits	U32 [1-3]	2	RW	Defines how many probes (SAA operations) can be configured (up to 64).
TypeBits	U32 [1-3]	2	RW	Defines how many measurement types can be configured per VPN (up to 8).

The combined value of the DeviceBits, TosBits and TypeBits attributes must be exactly 14.

Object diagram



Object inheritance

SAATemplate.Object

SAAOperation object

Description

An object that represents the parameters that will be used to configure an SAA operation and where the operation will be configured – that is, on one (half duplex) or both (full duplex) devices for each tested connection in a VPN.

Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RO	Name of the SAA operation object.
Type	Enum	0	RW	Type of operation: 0 = IcmpEcho 1 = TcpConnect 2 = UdpEcho 3 = Jitter
Duplex	Enum	0	RW	The test performed on a connection: 0 = HalfDuplex 1 = FullDuplex
Period	U32	0xFFFFFFFF	RW	Frequency with which a measurement is performed (seconds). Range: 0–604800 Default value indicates no specific period will be used.
Timeout	U32	0xFFFFFFFF	RW	Timeout in seconds. Must not be greater than frequency. Range 0–604 800 000 Default value indicates no specific timeout will be used.
Lifetime	U32	0xFFFFFFFF	RW	Lifetime of the operation defined in seconds. A value of –1 indicates the operation is active forever. Range: 0–2 147 483 647

Attribute Name	Type	Default	Access	Explanation
RisingThreshold	U32	0xFFFFFFFF	RW	Sets the rising threshold that generates a reaction event and stores history information for the operation. Defined in milliseconds. Frequency with which a measurement is performed (seconds). Default value indicates no specific period will be used. Range: 0–2 147 483 647
FallingThreshold	U32	0xFFFFFFFF	RW	Falling threshold in milliseconds. Default value indicates no specific period will be used. Range: 0–2 147 483 647
ThresholdType	Enum	1	RW	Algorithm used to calculate the passing of thresholds. 0 = Never 1 = Immediate 2 = Consecutive 3 = XofY 4 = Average
Consecutive Occurrences	U32	0	RW	The number of consecutive threshold violations that trigger the action defined by the ActionType attribute. 0 = Use default. Range: 0–16
Xofy_x	U32	0	RW	Value of X when ThresholdType is XofY. 0 = Use default. Range: 0–16

Attribute Name	Type	Default	Access	Explanation
Xofy_y	U32	0	RW	Value of Y when ThresholdType is XofY. 0 = Use default. Range: 0–16
Average	U32	0	RW	Parameter for 'Average' ThresholdType. Range: 0–16
ActionType	U32	0	RW	Bitwise value, indicating actions that occur when the threshold is passed. 0 = None 1 = Trap 2 = Nmvt 4 = Trigger
DsCodepoint	U32	0	RO	The IP ToS byte for request packets. Applies to ICMP Echo, UDP Echo and Jitter operations.
DestPort	U32	0	RW	Destination port number. 0 = use default. The following defaults apply: TcpConnect: 23 UdpEcho: 7 Jitter: 8000
SourcePort	U32	0	RW	Source port number. 0 = use default. The following defaults apply: TcpConnect: 23 UdpEcho: 7 Jitter: 8000

Attribute Name	Type	Default	Access	Explanation
RequestSize	U32	28-Icmp 1-Tcp 16-Udp 32-Jitter 28-Http 1-Dns	RW	Sets the protocol data size in the payload of the operation's request packet. 0xFFFFFFFF = Use default. Valid sizes are: IcmpEcho: >0–65535 TcpConnect: >=0–65535 UdpEcho: 4-8192 Jitter: 16-1500 Http: >=0 Dns: >=0
EnableControl	Boolean	False	RW	Enable/disable control message sent to destination port. Applies to TcpConnect, UdpEcho and Jitter.
PacketsInSequence	U32	10	RW	Number of packets in sequence. Range: 1–60000
PacketInterval	U32	20	RW	Inter-packet interval. Range: 1–60000 ms
EnableErrorChecking	Boolean	False	RW	True=Enables error verification checking.
EnableConnectChecking	Boolean	False	RW	True=Enables checks for connection loss in connection-oriented protocols.
EnableTimeoutChecking	Boolean	False	RW	True=Enables checks for RTR operation timeouts.
HistoryLives	U32	0	RW	Number of entries (lives) that are stored in the history table for a given operation. Range: 0–25

Attribute Name	Type	Default	Access	Explanation
HistoryFilter	Enum	0 – None	RW	Defines the type of history information that will be collected in the history table: 0 = None 1 = All 2 = OverThreshold 3 = Failures
HistoryBuckets	U32	15	RW	Specifies how many data points from which to record data for a given operation. Range: 1–60
SourcePortSpecified	Boolean	False	RW	Enables or disables the use of a the SourcePort attribute.

Object diagram

See SAATemplate.Object

Object inheritance

SAAOperation.SAATemplate.Object

RtNumber object

Description

Represents a Route Target. A user can create any number of RTs, and for each one select under what circumstances it is to be exported or imported.

RtNumber applicability to sites and VPNs

Two RtNumber objects are automatically created whenever a VPN is created. The attributes of the RT numbers are set according to whether the VPN's MplsVpnType attribute is HubAndSpoke, or FullMesh. The RT numbers represent the default VPN behavior, but may be altered by the user for different behavior. For example, in a hub and spoke VPN with several hubs, the hubs can be set to meshed or non-meshed by altering the RT number object's attributes.

Each RtNnumber object is given RtLowOrder and RtHighOrder values which are unique to the system. RtNumber objects also have three other attributes –

HubBehaviour, SpokeBehaviour, and MeshBehaviour which can take one of four values – Import, Export, ImportExport, or None.

By default, one of the RtNumber objects has HubBehaviour set to ImportExport, SpokeBehaviour set to Import, and MeshBehaviour set to ImportExport. The other RtNumber has HubBehaviour set to Import, SpokeBehaviour set to Export, and MeshBehaviour set to None.

A site in a VPN can see another site if it imports an RtNumber that the other site exports.

If the VPN is fully meshed, then the MeshBehaviour attribute is used from the RtNumber objects. In the default case, the MeshBehaviour is to import and export the first RtNumber, so all the sites import and export the same RtNumber ensuring that all the sites are meshed.

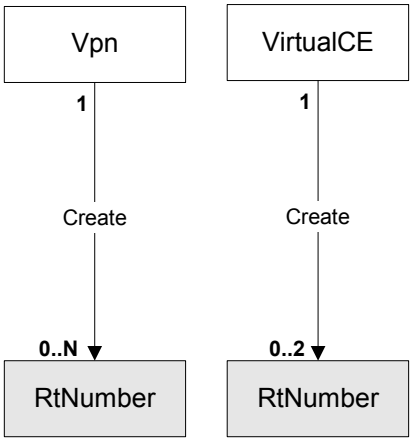
If the VPN is hub and spoke, the hub sites use the HubBehaviour, and the Spoke sites use the SpokeBehaviour. In the default case, the hub sites import and export Rt1. The spoke sites import Rt1, so the spokes can see the sites. The hub sites also import Rt2, and the spoke sites export Rt2, so the hubs can see the sites. The spokes export Rt2, but do not import Rt2, so the spokes cannot see other spokes. The hubs export Rt1, and import Rt1 so the hubs can see other hubs.

One common customization is to make the hubs non-meshed so that hubs cannot see other hubs. To achieve that, change the hub behavior of Rt1 from ImportExport to Export. This means that the hubs do not import the same number that they export, so the hubs will not be able to see each other.

RtNumber applicability to Virtual CEs

An RtNumber (RT) can be provisioned under a Virtual CE site. This functionality is enabled in the Virtual CE.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
RtHighOrder	U32	0	RW	Route Target number – high order value. If both RtHighOrder and RtLowOrder are set to 0, then the system default is used (the Domain ASN).
RtLowOrder	U32	0	RW	Route Target number – low order value. If both RtHighOrder and RtLowOrder are set to 0, then the system default is used (the ObjectId * 2).
HubBehaviour	Enum	0	RW	0 = None 1 = Import 2 = Export 3 = ImportExport

Attribute Name	Type	Default	Access	Explanation
SpokeBehaviour	Enum	0	RW	0 = None 1 = Import 2 = Export 3 = ImportExport
MeshBehaviour (See **Note** below)	Enum	0	RW	0 = None 1 = Import 2 = Export 3 = ImportExport
Name	String	""	RW	Name of the RtNumber object.
VrfTarget	Boolean	False	RCW	True = The Juniper cartridge generates VPN configuration using the VRF Target format. One each of Import, Export, and ImportExport can be created on Juniper devices. False = flag is disabled (default). The Juniper cartridge generates policy-based VPN configuration.

Note to MeshBehaviour

Values 0 (None) and 3 (ImportExport) are not supported for Virtual CEs. The complete set of route target restrictions for Virtual CEs follows:

- mesh
- zero or one import
- zero or one export

Object inheritance

RtNumber.Object

Site objects

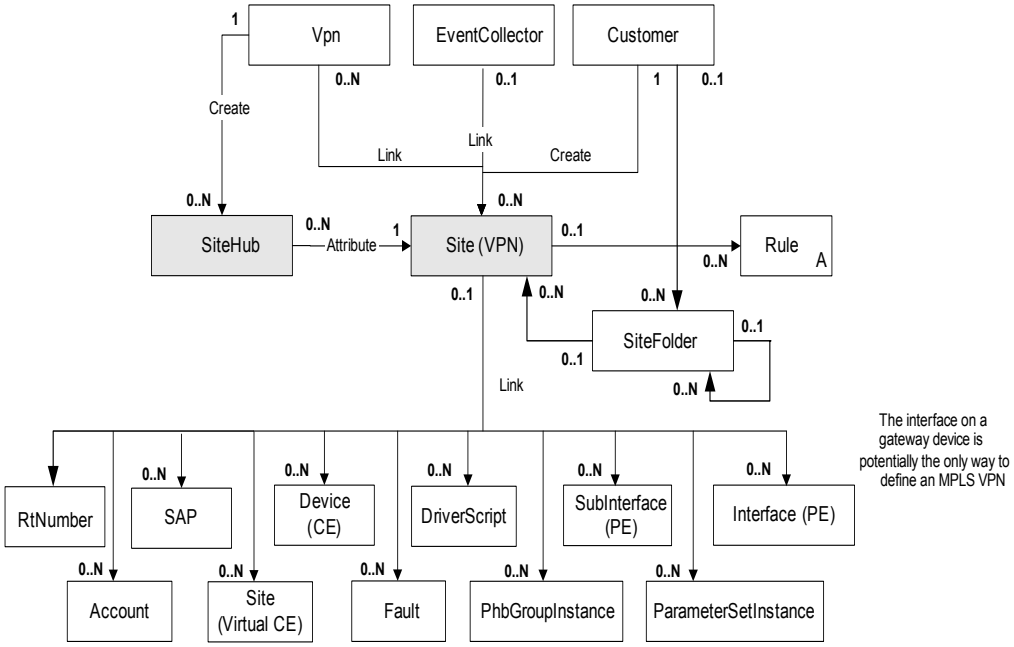
Site object

Description

A Site object has two types: VPN or Virtual CE.

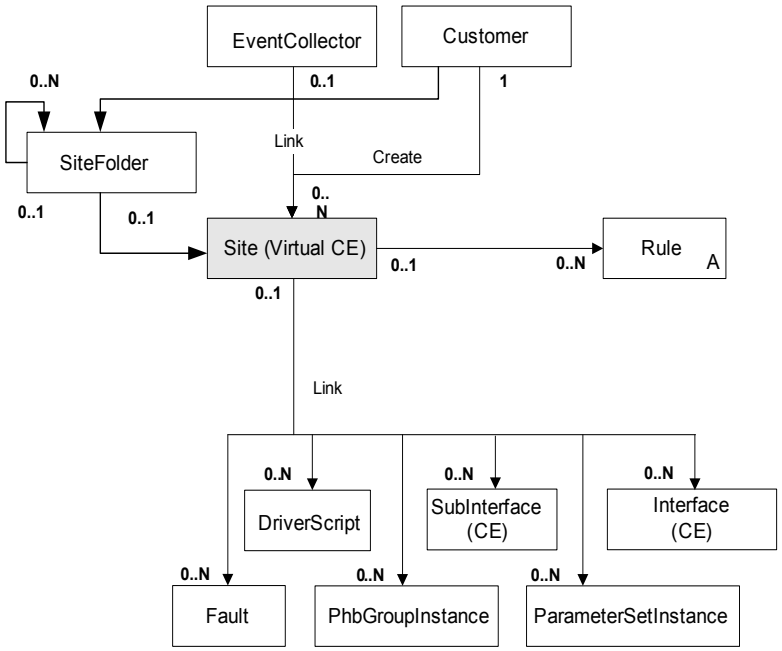
A Site object of type VPN represents a physical Site defined by one or more devices and interfaces. PHB groups and rules can be defined at a site level to apply to devices and interfaces within the site or closely related (such as an interface providing the VPN connection). Rules applied at site level are lower priority than rules defined at the device level.

Object diagram (Site object of type VPN)



A Site object of type Virtual CE represents a physical Site defined by one or more interfaces from a single CE device. PHB groups and rules can be defined at a site level to apply to interfaces within the site or closely related (such as an interface providing the VPN connection). Rules applied at site level are lower priority than rules defined at the device level.

Object diagram (Site object of type Virtual CE)



Attributes common to both Virtual CE and VPN site objects

Attribute Name	Type	Default	Access	Explanation
Type	Enum	VPN	RC	Type of site: 0 = VPN 1 = VirtualCE Type is defined when the object is created. You cannot convert one type to the other.
Name	String	""	RW	Name of the site.
Remarks	String	""	RW	Optional additional comments.

Attribute Name	Type	Default	Access	Explanation
Contact	String	""	RW	Name of primary contact at site.
Address	String	""	RW	Postal address of site.
Telephone	String	""	RW	Site phone number for contact.
Fax	String	""	RW	Site fax number for contact.
Email	String	""	RW	Site e-mail address for contact.
AccountRef	String	""	RW	Unique account number.
SiteOfOrigin	U32	4 294 967 295 (hex 0xffffffff)	RW	Service Activator generates the SOO value itself when required. Valid only if EBGP is the routing protocol. Range: 1–2 147 483 647 plus the default value.
Context	String		RW	
DomainVpnTag	U32	0	RW	To avoid routing loops when you are using OSPF as the routing protocol between a CE and PE device in an MPLS VPN. Routing loops can occur if OSPF routes are passed between PEs in the same network and VPN. The VPN Route tag is one mechanism that can be used to prevent routing loops in multi-homed VPNs.
EigrpAutonomousSystem	U32	0	RW	Specifies the EIGRP ASN for the site.

Attribute Name	Type	Default	Access	Explanation
EigrpSooAsn	U32	0	RW	The Site of Origin will have the form <ASN>:<Origin ID>. EigrpSooAsn is the first part of that value which is <ASN>.
EigrpSooOid	U32	0	RW	The Site of Origin will have the form <ASN>:<Origin ID>. EigrpSooOid is the second part of that value which is <Origin ID>.
EigrpSooRouteMapName	String		RW	Specify a manually configured Route Map. Make sure that the name contains no spaces; this includes leading or trailing spaces and spaces between characters. If there is any space present in the name, it will not be configured on the device.
UseEigrpSoo	Boolean	False	RW	To enable Site of Origin.
GeneratedSoo	Boolean	False	RW	To have Service Activator generate the Site of Origin.
InheritAsn	Boolean	False	RW	To have the site inherit the default EIGRP ASN specified for the VPN.
PublicIpAddress	IPAddress	0.0.0.0	RW	The public IP address of the interface connected to the site.
RedistributeStatic	Boolean	False	RW	Static routes are redistributed into the dynamic routing protocols configured for the site.

Attributes specific to VPN site objects

Attribute Name	Type	Default	Access	Explanation
BgpAsn	U32	0	RW	BGP Autonomous System Number. Unique number for routing.
RoutingProtocol	Enum	3	RW	Type of routing being used between the PE and CE, relevant to MPLS VPNs only: 0 = EBGP 1 = RIP 3 = None 4 = OSPF
InstallStatic	Boolean	True	RW	True = Static routing is used in conjunction with relevant routing protocol. False = Static routing is not used.
InstallLocalStatic	Boolean	True	RW	True = Static routes defined in the site are not redistributed. False = Static routes defined in the site are redistributed.

Attributes specific to Virtual CE site objects

Attribute Name	Type	Default	Access	Explanation
VirtualCEModellingOnly	Boolean	True	RW	True = no Virtual CE provisioning is done. False = Virtual CE provisioning is attempted.
InheritProxyAssignment	Boolean	True	RW	True = The site uses the domain-level dual-dispatch proxy. False = The site uses another proxy agent.
ProxyId	String	""	RW	Name of the proxy agent to use for dual-dispatch of the Virtual CE configuration if "InheritProxyAssignment" is False.
RDHighOrder	U32	0	RW	The top 32 bits of the Route Descriptor value.
RDLowOrder	U32	0	RW	The bottom 32 bits of the route Descriptor value.
VrfTableName	String	""	RW	The name of the VRF routing table.
OverrideVrfTable Limit	Boolean	False	RW	True = Use site-specific settings for VRF table limits False = Use domain defaults for VRF table limits.
VrfTableLimit	U32	0	RW	Maximum number of routes allowed in a VRF (0=No limit).

Attribute Name	Type	Default	Access	Explanation
VrfTableLimit Warning	U32	0	RW	Percentage at which to warn of VRF table limits being exceeded. Range: 1–101, where 1–100 = percentage of VrfTableLimit reached warning. 101 = warning when VrfTableLimit reached.
VrfDesc	String	""	RW	Optional string describes the VRF applied to router, if the router supports VRF description string.
EBgpMaxPaths	U32	1	RW	Maximum number of multipaths on EBGP. Range: 1-6. Valid only when protocol is EBGP on Virtual CE.
VrfRedundantPaths	U32	1 (turned off)	RW	The number of device redundant path configurations. Range is 2^{16} plus default value.
RedistributeDefault Route	Boolean	False	RW	True = Redistribute the Default route. False = Don't.
RedistributeConnected	Boolean	False	RW	True = Redistribute the connected routes. False = Don't.
RedistMetricConnected2 Bgp	U32	0	RW	Values for redistribution metric:distribution into BGP. Range is 0–2^{32}-1
RedistMetricStatic2Bgp	U32	0	RW	

Attribute Name	Type	Default	Access	Explanation
RedistPolicyConnected2Bgp	String	""	RW	Values for redistribution routemap name (policy): distribution into BGP.
RedistPolicyStatic2Bgp	String	""	RW	

Object inheritance

Site.Object

SiteHub object

Description

A SiteHub object indicates that a site is a hub of a hub and spoke or management VPN. To make an existing site a hub site, create a SiteHub object as a child of the VPN with the same name as the site to be a hub. To make the site a spoke again, delete the SiteHub object.

Object diagram

See Site *Object diagram (Site object of type VPN)* on page 118.

Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RW	Name of the site that is a hub in the parent VPN.

Object inheritance

SiteHub.Object

SiteL2 object

Description

SiteL2 objects represent Layer 2 sites – used when configuring Layer 2 connectivity services.

For port-based TLS services (i.e. the ServiceType attribute is PortBased), each Layer 2 site can contain only PE interfaces, and if multiple interfaces are included in the TLS, they must be on the same PE devices. In the case of a port-based TLS, no VLAN will be created on any device and therefore none of the VLAN-related attributes on the SiteL2 objects will be used.

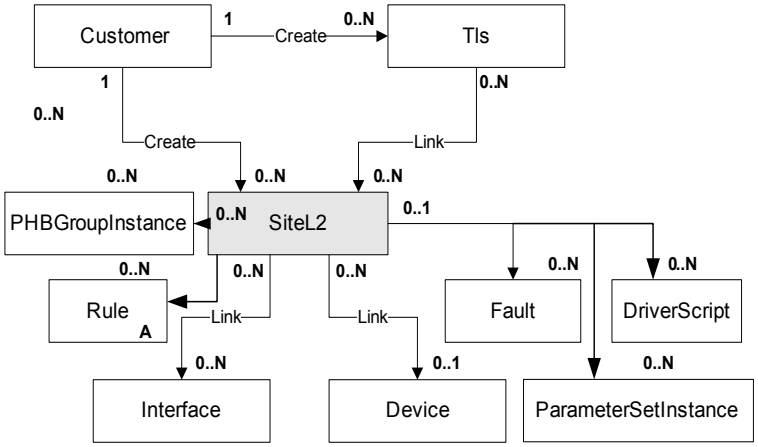
For port and VLAN-based TLS services (i.e. the ServiceType attribute is PortAndVlanBased), Layer 2 sites can contain also a CE/MTU device and both the CE port connecting the CE to the PE and the CE customer facing ports. The CE customer-facing ports must be assigned a role of Local, while the CE-to-PE port must be assigned a role of Access. Both the CE devices and its customer-facing interfaces must be linked to the site if the system is required to configure the CE/MTU. It is recommended that the CE port connecting the CE to the PE is also part of the site. Alternatively, the system will try and find an Access port on the CE device and consider that port as the port connecting the CE to PE.

It is important to note that the same physical interface (both on the PE and on the CE) can be part of multiple L2Site objects. In fact in the context of Layer 2 VPNs, trunk ports cannot be restricted to one customer only. Instead, physical ports can be part of multiple sites as long as the VLAN ID ranges installed on the same CE devices via different TLSs/Sites do not overlap. The EOM will validate that this is the case. It is important to note that this restriction will apply to CE/MTU devices only. In fact it is assumed that PE ports that are part of different PortAndVlan based TLSs.

The EOM also validates that all PE interfaces that are part of a SiteL2 object belong to the same PE device. In fact in the context of a TLS, this is the only meaningful configuration. No peering relationship will be established between PE devices whose interfaces are part of the same site.

The maximum speed specified for inbound and/or outbound rate limiting will be applied to the customer-facing ports in the L2Site. (Customer-facing ports are the PE Access Ports in those sites where there are no MTUs or the CE Local Ports in those sites that contain an MTU.) If this attribute is zero, no rate limiting is applied to the ports. If a port is part of multiple Layer 2 sites with conflicting rate limiting configuration, a warning will be raised.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RCW	Name of the Layer 2 site.
Remarks	String	""	RCW	Optional additional comments about this site.
Contact	String	""	RCW	Name of primary contact at site.
Address	String	""	RCW	Postal address of site.
Telephone	String	""	RCW	Site phone number for contact.
Fax	String	""	RCW	Site fax number for contact.
Email	String	""	RCW	Site e-mail address for contact.
AccountRef	String	""	RCW	Unique account number.
ServiceType	Enum	PortBased	RCW	0 = PortsBased 1 = PortsAndVlanBased

Attribute Name	Type	Default	Access	Explanation
The following attributes are used if ServiceType = 1				
SitePortsConfig	Enum	NoEncapsOnlyTaggedFrames	RCW	Configuration of the ports to and from the site. If the site contains only the PE, this configuration will affect the PE access port. If the site includes a CE/MTU device, this will affect the CE customer facing ports. 0 = NoEncapsOnlyTagged Frames. Do not encapsulate and accept tagged frames with VlanIds (trunk port) 1 = EncapsNoTagged Frames. Encapsulate all frames with EncapsVlanId and do not accept previously tagged frames (access port with encapsulation)
VlanIds	String	""	RCW	Lists all the VLAN IDs used in SitePortsConfiguration. List of VLAN IDs to configure on the customer facing interfaces. It is meaningful only if SitePortsConfig = 0.
EncapsVlanId	U32	0	RCW	Specifies the VLAN ID used to encapsulate frames if SitePortsConfig = 1.
VlanType	Enum	VLAN	RCW	

Object inheritance

Tls.Object

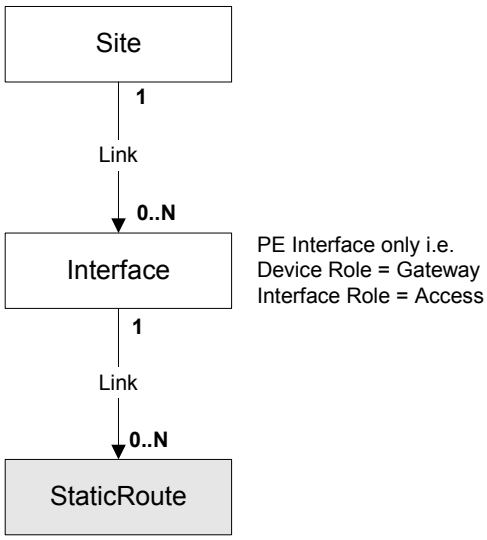
StaticRoute object

Description

A StaticRoute object represents a static route defined for a VPN site (relevant if static routing is used between the PE and CE router). A StaticRoute object can be linked to an interface object that is a child of a site. If the interface is then unlinked from the site, the StaticRoute object will be deleted.

A StaticRoute object can only be created as a child of an interface that is linked to a site, but not linked to an interface.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RW	Name of the static route.
IpAddr	IPAddress	0.0.0.0	RW	IP Address to match on.
SubnetMask	IPAddress	0.0.0.0	RW	Subnet mask for route.
NextHop	IPAddress	0.0.0.0	RW	Next hop IP address.
NextHopType	Enum	IpAddress IfName	RW	How the next hop IP address is to be specified in the configuration command. 0=IP Address & I/F 1=IP Address only 2=I/F only 3=Null0 I/F
Metric	U32	1	RW	Metric for the route (that is, a weight).
Global	Boolean	False	RW	True = the given next hop address is in the non-VRF routing table False = the next hop address is not a global route.
Permanent	Boolean	False	RW	True = this route will not be removed, even if the interface shuts down. False = route is not permanent.

Attribute Name	Type	Default	Access	Explanation
UseTagNumber	Boolean	False	RW	True = TagNumber will be configured.
TagNumber	U32	0	RW	Label (tag) value that can be used for controlling redistribution of routes through route maps.

Object inheritance

StaticRoute.Object

Traffic type objects

A traffic type is a categorization of specific IP traffic on the network. Traffic types are used as components of rules to define the traffic to which a QoS or security policy is applied.

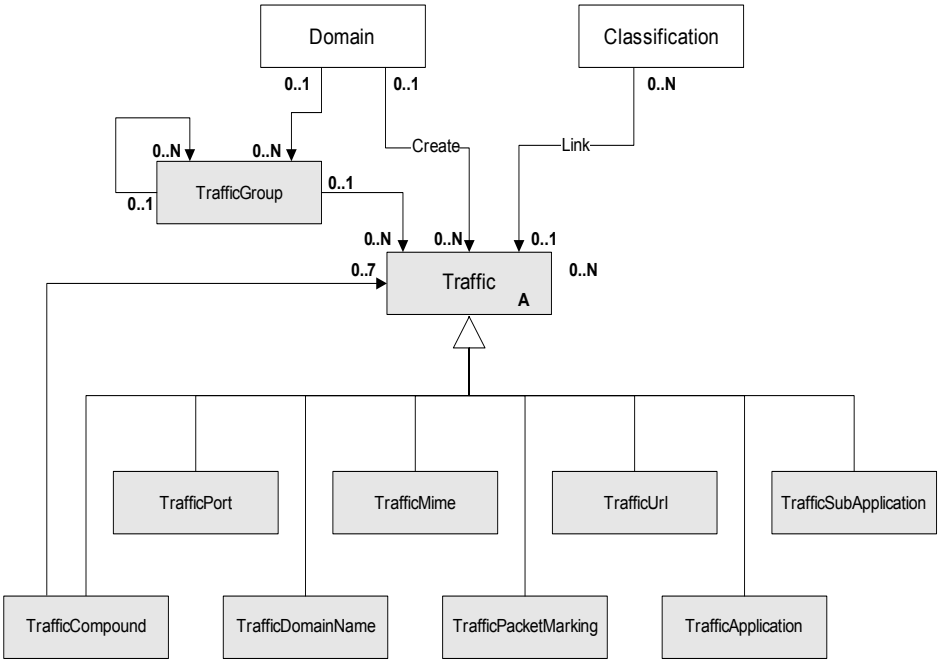
There are a number of different objects representing different traffic type categorization:

Object	Purpose
Traffic	Abstract traffic object, for inheritance purposes.
TrafficGroup	Represents a group of traffic types, allowing traffic types to be organized into a hierarchical folder-like structure. Traffic type groups are for administrative purposes only; you cannot apply rules to them.
TrafficCompound	Represents a compound traffic type; a combination of two or more traffic types.
TrafficMime	Represents a MIME-based traffic type, which classifies traffic by MIME type and packet length.
TrafficUrl	Represents a URL-based traffic type, which classifies traffic by URL.

Object	Purpose
TrafficPort	Represents a port-based traffic type, which classifies traffic by IP port number and IP protocol. Both source and destination ports can be defined either as a single port or a range.
TrafficApplication	Represents an application protocol-based traffic type, which classifies traffic by application protocol name.
TrafficSubApplication	Represents a subapplication-based traffic type, which classifies traffic by subapplication name.
TrafficPacketMarking	Represents a Packet-Marking-based traffic type, which classifies traffic by DiffServ Codepoint/IP Precedence/ MPLS Experimental bits.
TrafficDomainName	Represents a domain name-based traffic type, which classifies traffic by DNS Domain Name.

The most common way of classifying traffic is by port number and/or IP protocol, since almost all devices support this. Where supported by devices, it is also possible to identify traffic by packet marking, DNS domain name, application, sub-application, and MIME type or URL for HTTP traffic.

Object diagram



Traffic object (abstract)

Description

A Traffic object is an abstract object that defines attributes common to all traffic types.

Object diagram

See *Object diagram* on page 133.

Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RO/RC*	Name of traffic object.
Remarks	String	""	RO/RC*	Optional additional comments.

* Access is Read Only unless creating a TrafficUrl object, in which case access is Read/Create.

Object inheritance

Traffic.Object

TrafficGroup object

Description

A TrafficGroup object represents a group [folder] of traffic types. This folder is an administrative entity only, allowing traffic types to be grouped together. These groupings appear in the tree hierarchy but cannot be selected as part of a rule.

Object diagram

See *Object diagram* on page 133.

Attributes

No additional attributes over the Traffic object (abstract).

Object inheritance

TrafficGroup.Traffic.Object

TrafficCompound object

Description

A TrafficCompound object allows two or more traffic types to be combined, for example to identify traffic by port/IP protocol and by URL.

Currently one of each basic type only is allowed. Certain combinations may not in fact make sense, but no restrictions are enforced. The traffic types linked to a TrafficCompound object are not visible via the EOM, but their values are shown as attributes of the TrafficCompound.

Object diagram

See *Object diagram* on page 133.

Attributes

The TrafficCompound contains all the attributes of the other Traffic types.

Attribute Name	Type	Default	Access	Explanation
URL	String		RW	Text string representing a URL.
MIMEType	String	""	RW	Text string representing a MIME type, e.g. text/plain; charset=us-ascii
MinLength	U32	0	RW	Mime: Minimum length of packets in bytes. Range: 0–65535 Must be less than the maximum length.
MaxLength	U32	0	RW	Mime: Maximum length of packets in bytes Range: 0–65535 Must be greater than the minimum length.
DestPortMax	U32	0	RO	Port: Upper bound of destination port range. Same as SourcePortMax to specify a single port. Range: 0–65535
DestPortMin	U32	0	RO	Port: Lower bound of destination port range. Range: 0–65535

Attribute Name	Type	Default	Access	Explanation
IpProtocol	U32	0	RO	Port: Number representing IP protocol (such as 6 for TCP, 17 for UDP, etc). Range: 0–255
SourcePort Max	U32	0	RO	Port: Upper bound of source port range. Same as SourcePortMin to specify a single port. Range: 0–65535
SourcePort Min	U32	0	RO	Port: Lower bound of source port range. Range: 0–65535
ApplicationN ame	String		RW	Valid name of the application protocol on which the traffic type is based, such as "http".
SubApplicati onName	String		RW	Name of subapplication.
DomainNam e	String		RW	DNS Domain Name (partial match is allowed).

Attribute Name	Type	Default	Access	Explanation
MarkingValue	U32	0	RO	Actual marking value. Ex: For IPPrecedence MarkingType. Range: 0–7
MarkingType	Enum		RO	0 = DSCodepointValue 1 = MplsHeader 2 = FrDe 3 = AtmClp 4 = AlcatelIntQ 5 = AlcatelIntQ3CoS 6 = AlcatelIntQ1CoS 7 = IPPrecedence 8 = DiscardClass 9 = Trust

Object inheritance

TrafficCompound.Traffic.Object

TrafficMime object

Description

A TrafficMime object represents a MIME-based traffic type, which classifies traffic by MIME type and packet length.

Object diagram

See *Object diagram* on page 133.

Attributes

Attribute Name	Type	Default	Access	Explanation
MIMEType	String	""	RCW	MIME type.
MinLength	U32	0	RCW	Minimum length in bytes.
MaxLength	U32	0	RCW	Maximum length in bytes.
Name	String	""	RCW	Name of the TrafficMime
Remarks	String	""	RCW	
Id	U32	6331	RO	

Object inheritance

TrafficMime.Traffic.Object

TrafficUrl object

Description

A TrafficUrl object represents a URL-based traffic type, which classifies traffic by URL.

Object diagram

See *Object diagram* on page 133.

Attributes

Attribute Name	Type	Default	Access	Explanation
URL	String	""	RW	Text string representing a URL. A wildcard (*) at the end of the character string can be used to match multiple URLs.

Object inheritance

TrafficUrl.Traffic.Object

TrafficPort object

Description

A TrafficPort object represents a port-based traffic type, which classifies traffic by IP port number and IP protocol.

Both source and destination ports can be defined either as a single port or a range. To define a single port, set the Min and Max to the same value. If the Min and Max values are both set to 0, no port applies.

Object diagram

See *Object diagram* on page 133.

Attributes

Attribute Name	Type	Default	Access	Explanation
SourcePortMin	U32	0	RW	Lower bound of source port range. Range: 0–65535
SourcePortMax	U32	0	RW	Upper bound of source port range. Same as SourcePortMin to specify a single port. Range: 0–65535
DestPortMin	U32	0	RW	Lower bound of destination port range. Range: 0–65535
DestPortMax	U32	0	RW	Upper bound of destination port range. Same as SourcePortMax to specify a single port. Range: 0–65535
IpProtocol	U32	0	RW	Number representing IP protocol (such as 6 for TCP, 17 for UDP, etc). Range: 0–255

Attribute Name	Type	Default	Access	Explanation
TcpOptions	U32	0	RW	Bitwise attribute containing various TCP control bits. Do not choose both ACK and RST bits - these two bits comprise the Established status. Urg = 1, Ack = 2, Psh = 4, Rst = 8, Syn = 16, Fin = 32, Established = 128
IcmpOptions	U32	0	RW	Bitwise attribute. Echo-Request = 1, Echo-Reply = 2, TTL-Exceeded = 4, Unreachable = 8, Redirect = 16, Time-Exceeded = 32, Packet-Too-Big = 128, Source-Quench = 256, AdministrativelyProhibited = 512

Object inheritance

TrafficPort.Traffic.Object

TrafficApplication object

Description

A TrafficApplication object represents an application protocol-based traffic type, which classifies traffic by application protocol name.

Object diagram

See *Object diagram* on page 133.

Attributes

Attribute Name	Type	Default	Access	Explanation
ApplicationName	String		RW	Valid name of the application protocol on which the traffic type is based, such as "http".
Type	Enum	Application	RW	Application = 0, RTP = 1
UDPPortRange	U32	0	RW	Range of the UDP port for RTP. Options are 0-16383
UDPStartPort	U32	2000	RW	UDP start port number for RTP. Range is 2000-65535.

Object inheritance

TrafficApplication.Traffic.Object

TrafficSubApplication object

Description

A TrafficSubApplication object represents a subapplication-based traffic type, which classifies traffic by subapplication name.

Object diagram

See *Object diagram* on page 133.

Attributes

Attribute Name	Type	Default	Access	Explanation
ApplicationName	String	""	RO	Name of subapplication.

Object inheritance

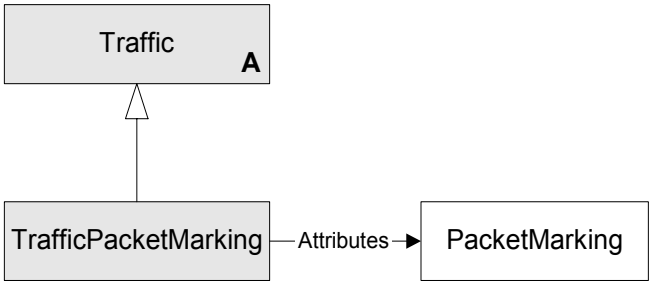
TrafficSubApplication.Traffic.Object

TrafficPacketMarking object

Description

A TrafficPacketMarking object represents a packet marking-based traffic type, which classifies traffic by DiffServ Codepoint, IP Precedence or MPLS Experimental bits.

Object diagram



Attributes

None.

Object inheritance

TrafficPacketMarking.Traffic.Object

TrafficDomainName object

Description

A TrafficDomainName object represents a domain name-based traffic type, which classifies traffic by DNS Domain Name.

Object diagram

See *Object diagram* on page 133.

Attributes

Attribute Name	Type	Default	Access	Explanation
DomainName	String	""	RO	DNS Domain Name (partial match is allowed).

Object inheritance

TrafficDomainName.Traffic.Object

Rule objects

A QoS or access control policy is implemented by creating and applying a set of rules. Generally, each rule consists of a set of conditions that, when true, result in a set of actions.

There rule objects in the EOM are:

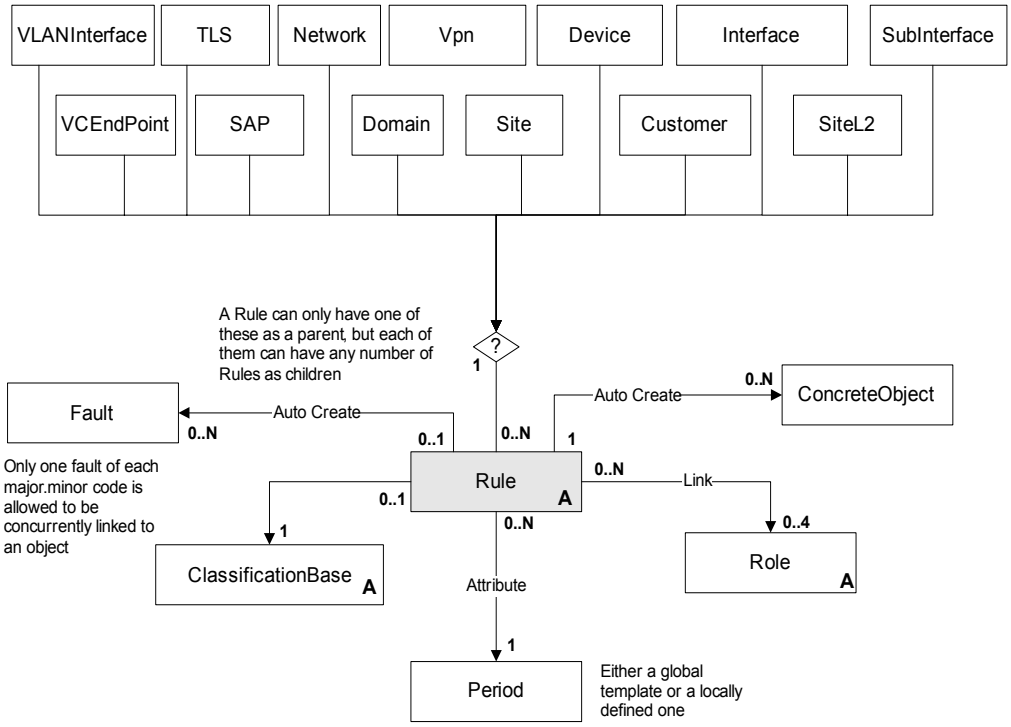
- **RuleAccess** – Represents an access rule, used to deny or permit access to the network for specific identified traffic.
- **RuleClassification** – Represents a classification rule, used to classify, mark, and manage network traffic.
- **RuleGeneric** – Represents a configuration policy, which allows the entry of either a set of raw XML commands, or an HTML-based entry form to collect information which is then converted to XML. The XML commands are passed to the Network Processor.
- **RulePolicing** – Represents a policing rule, used to classify traffic, set bandwidth and burst requirements and define actions for conforming and exceeding traffic.

Rule object (Abstract)

Description

The abstract Rule object defines attributes common to all rule types.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RW	Name of the rule.
Order	U32	0	RW	Rule Order. Indicates pre-requisites ordering for a set of rules. Lowest number (0) indicates highest order. If changed for one rule, all lower priority rules are renumbered.
Disabled	Boolean	False	RO	True = rule is to be disabled at next propagate. False = rule is not to be disabled.
InError	Boolean	False	RO	True = rule is currently failing. False= rule does not contain errors.
Outbound	Boolean	Value depends on rule type	RW	True = rule is active on traffic outbound from an interface. False = rule is not active outbound.
Inbound	Boolean	Value depends on rule type	RW	True = rule is active on traffic inbound to an interface False = rule is not active inbound.
StartTime	DateTime	Now	RW	Start date/time - YYYY/ MM/DD HH:MM:SS.
EndTime	DateTime	2020/01/01 00:00:00	RW	End date/time - YYYY/ MM/DD HH:MM:SS.

Attribute Name	Type	Default	Access	Explanation
DaysOfWeek	U32	127 (i.e. all days selected)	RW	Bitwise value, 1 bit per day of the week: Bit 0 = Monday Bit 1 = Tuesday Bit 2 = Wednesday Bit 3 = Thursday Bit 4 = Friday Bit 5 = Saturday Bit 6 = Sunday
The following attribute can be used to link directly to other objects in the EOM:				
PeriodName	String	Null	RW	Template name if global, otherwise empty string. Entering a global name links the rule to the Period and sets StartTime, EndTime and DaysOfWeek to Read Only. The global period objects must be linked to the domain to be found.

Object inheritance

Rule.Object

RuleAccess object

Description

A RuleAccess object represents an access rule. Access rules are used to provide security by denying or permitting access to the network for specific identified traffic. A RuleAccess object is defined in terms of the following conditions and actions:

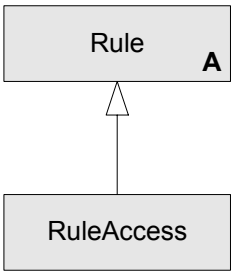
Conditions:

- Classifications: One or more Classification objects defining the traffic the rule applies to in terms of source address/mask, destination address/mask and traffic type.
- Date and time: Identified by Period
- Direction: Identified by Outbound and Inbound attributes

Actions:

Permit or deny access to traffic matching the conditions.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Permit	Boolean	True	RCW	Permit or Deny traffic; list is searched until match occurs. True=Explicit permit, stop searching ordered list. False=Deny.
IsNamedAcl	Boolean	False	RCW	Specify if the access rule will be implemented as a named ACL. If set to false, the access rule will be implemented as a numbered ACL.
AclName	String	""	RCW	Name of the ACL implemented for this access rule. If left empty, the system generates the name.
AclNumber	U32	168	RCW	Number of the ACL implemented for this access rule. 0 means <Generate> a number.

Attribute Name	Type	Default	Access	Explanation
Management Override	Boolean	False	RCW	To avoid automatic creation of ACL rules. ACL rules are automatically generated to assure SNMP and Telnet access from IPSA to the CE.
Log	Boolean	False	RCW	Enables / Disables Cisco ACL logging. Headers of packets affected by the access rule are stored in the routing engine and can be displayed using a show log command. Note: Enabling ACL logging may negatively affect router performance: ■ processor power may be used to perform the logging ■ log files may impact available disk space
Disabled	Boolean	False	RCW	When selected, it switches off the rule.
Fragments	Boolean	False	RCW	When checked, non-initial packet fragments are included for matching. (This relates to the fragments parameter in Cisco ACL statements).
Inbound	Boolean	True	RCW	Applies to inbound traffic.
InError	Boolean	False	RO	True = rule is currently failing. False= rule does not contain errors.

Attribute Name	Type	Default	Access	Explanation
Name	String	"LinkingRuletoNetwork"	RCW	Identifying name of the access rule.
Outbound	Boolean	False	RCW	Applies to outbound traffic.
Order	U32	0	RCW	Rule Order. Indicates pre-requisites ordering for a set of rules. Lowest number (0) indicates highest order. If changed for one rule, all lower priority rules are renumbered.
DaysOfWeek	U32	127	RCW	Bitwise value, 1 bit per day of the week: Bit 0 = Monday Bit 1 = Tuesday Bit 2 = Wednesday Bit 3 = Thursday Bit 4 = Friday Bit 5 = Saturday Bit 6 = Sunday
EndTime	Date	2020/01/01 00:00:00	RCW	End date/time - YYYY/MM/DD HH:MM:SS.
PeriodName	String	""	RCW	Template name if global, otherwise empty string. Entering a global name links the rule to the Period and sets StartTime, EndTime and DaysOfWeek to Read Only. The global period objects must be linked to the domain to be found.

Attribute Name	Type	Default	Access	Explanation
StartTime	Date	2008/10/20 05:34:49	RCW	Start date/time - YYYY/MM/DD HH:MM:SS.
ID	U32	6355	RO	Internal ID number of this object; allocated automatically by Service Activator.

Object inheritance

RuleAccess.Rule.Object

RuleClassification object

Description

A RuleClassification object represents a classification rule. Classification rules are used to classify traffic and mark and/or shape traffic.

A RuleClassification object is defined in terms of the following conditions and actions:

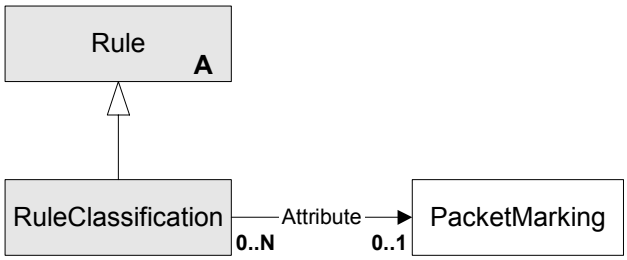
Conditions:

- Classifications: One or more Classification objects defining the traffic the rule should apply to in terms of source address/mask, destination address/mask and traffic type.
- Date and time: Identified by Period
- Direction: Identified by Outbound and Inbound attributes

Actions:

- Mark IP packets with DiffServ codepoint, IP Precedence value or MPLS experimental value.
- Apply a maximum bandwidth, if possible (not used at present)
- Apply a guaranteed minimum bandwidth (not used at present).

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
BandwidthGuaranteed	U32	0	RCW	Bandwidth level guaranteed for the traffic (in Kbits/s). Must be >= BandwidthLimit, if set. 0 indicates no guarantee.
BandwidthLimit	U32	4294967295	RCW	Maximum bandwidth that can be allocated to traffic (in Kbits/s). Must be <= BandwidthGuaranteed, if set. –1 indicates no limit.
MarkingType	Enum	DSCodepointValue	RO	Marking to be applied 0 = DsCodepoint 1 = MplsExperimental
MarkingValue	U32	0	RO	This value depends on the Marking Type, and defines how to mark matching traffic. Value restricted to 0–255.
Disabled	Boolean	False	RCW	When selected, it switches off the rule.
Inbound	Boolean	False	RCW	Applies to inbound traffic.
InError	Boolean	False	RO	True = rule is currently failing. False= rule does not contain errors.

Attribute Name	Type	Default	Access	Explanation
Name	String	"RuleClassify"	RCW	Name of DiffServ codepoint or MPLS experimental bit with which packets are to be marked. The combo box lists all packet markings that have been set up in Service Activator. Where supported, packets can be marked using: ■ IP DiffServ codepoint values ■ MPLS experimental bit values ■ IP Precedence ■ Frame Relay DE bit ■ ATM CLP bit ■ None - no packets are marked
Outbound	Boolean	True	RCW	Applies to outbound traffic.
Order	U32	0	RCW	Rule Order. Indicates pre-requisites ordering for a set of rules. Lowest number (0) indicates highest order. If changed for one rule, all lower priority rules are renumbered.

Attribute Name	Type	Default	Access	Explanation
DaysOfWeek	U32	127	RCW	Bitwise value, 1 bit per day of the week: Bit 0 = Monday Bit 1 = Tuesday Bit 2 = Wednesday Bit 3 = Thursday Bit 4 = Friday Bit 5 = Saturday Bit 6 = Sunday
EndTime	Date	2020/01/01 00:00:00	RCW	End date/time - YYYY/MM/DD HH:MM:SS.
PeriodName	String	""	RCW	Template name if global, otherwise empty string. Entering a global name links the rule to the Period and sets StartTime, EndTime and DaysOfWeek to Read Only. The global period objects must be linked to the domain to be found.
StartTime	Date	2008/10/20 05:34:49	RCW	Start date/time - YYYY/MM/DD HH:MM:SS.
Id	U32	6366	RO	Internal ID number of this object; allocated automatically by Service Activator.

Attribute Name	Type	Default	Access	Explanation
The following attribute can be used to link directly to other objects in the EOM				
PacketMarkingName	String	""	RW	Name of the PacketMarking object to use to mark packets.

Object inheritance

RuleClassification.Rule.Object

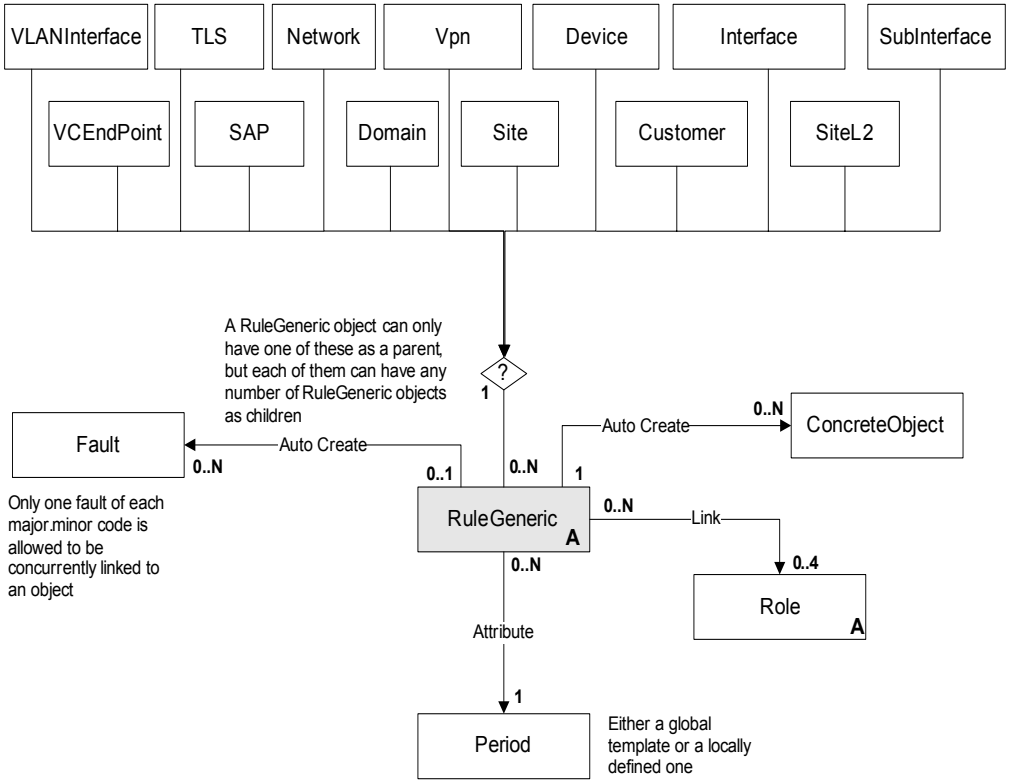
RuleGeneric object

Description

A RuleGeneric object represents a configuration policy, which allows the entry of either a set of raw XML commands, or an HTML-based entry form to collect information which is then converted to XML. The XML commands are passed to the Network Processor.

You can also specify the ContentValue of a RuleGeneric object as a filename. The filename must be an absolute and complete path ending with ".xml" extension. The file itself can be xml or multi-line. The file contents and namespace are not validated.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
ContentType	String	""	RCW	Configuration policy type.
ContentValue	String	""	RCW	XML content of the configuration policy.
Disabled	Boolean	False	RCW	If True, no concrete will be created.
InheritProxyAssignment	Boolean	True	RCW	If True, the same Proxy Agent as the Domain under which the Configuration Policy resides is used. If False, the Proxy Agent specified in ProxyId is used.
Name	String	""	RCW	Name of the configuration policy.
ProxyId	String	""	RCW	The Proxy Agent to use if InheritProxyAssignment is False.
DontCascadeTargets	Boolean	True	RC	Should this policy create concretes at the point of application and on all child Configuration Targets (if set to false) or only on the specific Configuration Target that it is linked to (if set to true)?

Attribute Name	Type	Default	Access	Explanation
Inbound	Boolean	False	RCW	Configuration policy applies to incoming traffic. True = Configuration policy is active on traffic inbound to an interface False = Configuration policy is not active inbound
InError	Boolean	False	RO	True = Configuration policy is currently failing False = Configuration policy does not contain errors
Outbound	Boolean	True	RCW	Configuration policy applies to outbound traffic. True = Configuration policy is active on traffic outbound from an interface False = Configuration policy is not active outbound
Order	U32	0	RCW	Configuration policy Order.
DaysOfWeek	U32	127	RCW	Bitwise value, 1 bit per day of the week: Bit 0 = Monday Bit 1 = Tuesday Bit 2 = Wednesday Bit 3 = Thursday Bit 4 = Friday Bit 5 = Saturday Bit 6 = Sunday

Attribute Name	Type	Default	Access	Explanation
EndTime	Date	2020/01/01 00:00:00	RCW	End date/time - YYYY/MM/DD HH:MM:SS.
PeriodName	String	""	RCW	Template name if global, otherwise empty string. Entering a global name links the rule to the Period and sets StartTime, EndTime and DaysOfWeek to Read Only. The global period objects must be linked to the domain to be found.
StartTime	Date	Now	RCW	Start date/time - YYYY/MM/DD HH:MM:SS.
Id	U32		RO	Internal ID number of this object; allocated automatically by Service Activator.

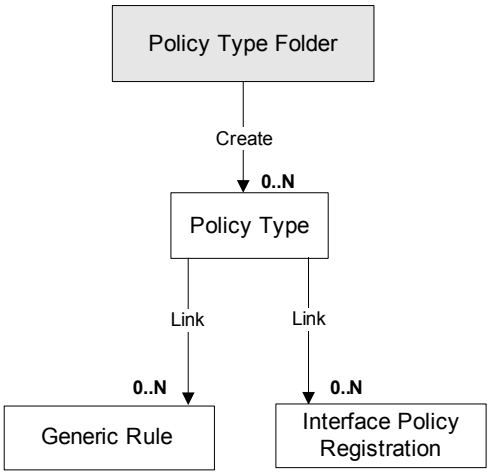
Object inheritance

RuleGeneric.Rule.Object

Policy Type Object

The Policy Type object holds information regarding the creation of RuleGeneric objects. Policy Type objects act as templates for Rule Generic Object. They contain the HTML text that is used to populate the forms for a Generic Rule.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RW	Name of the policy type.
Category	Enum	Interface, Service, and Unclassified	RW	Configures the category of the policy type.
Type	String	""	RW	Configures the type of policy
Remarks	String	""	RW	Description of the type of policy.
HtmlPage	String	""	RW	Imported HTML to be used for data entry.
Schema	String	""	RW	The schema for which the XML is validated.

Attribute Name	Type	Default	Access	Explanation
Disabled	Boolean	False	RW	Specify if the policy type is disabled.
Visible	Boolean	True	RW	Specify if the policy type is visible in the IPSA GUI.

Policy Type Folder Object

The Policy Type Folder object categorizes policy types within the Object Model. A Policy Type must be located in a Policy Type Folder. These folders can also create a hierarchy of Policy Types and Policy Type Folders. Each Policy Type folder can have either a Policy Type Folder or a Policy Type as parent. Only those Policy Types, that have a matching Category Type, can reside under the folder of a particular category.

Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String		RCW	Name of the configuration policy folder.
Remark	String		RCW	Remarks for this configuration policy folder.
Category	Enum	Interface, Service, and Unclassified	RCW	Category of the configuration policy folder.

RulePolicing object

Description

A policing rule is used to police identified traffic between a defined source and destination point. A rule defines the bandwidth and burst requirements for a given

traffic type, between a source and destination, and the action to be taken if traffic conforms to or exceeds the requirements.

A Policing Rule is defined in terms of the following conditions and actions:

Conditions

- Classifications: One or more Classification objects defining the traffic the rule should apply to in terms of source address/mask, destination address/mask and traffic type.
- Date and time: Identified by Period
- Direction: Identified by Outbound and Inbound attributes

Actions

- Set CAR parameters: Committed Rate, Normal Burst Size and Excess Burst Size.
- Traffic that conforms to or exceeds a specified bandwidth can be dropped, transmitted, or allowed to continue. In addition, you can choose to re-mark both conforming and exceeding traffic with a different packet marking.

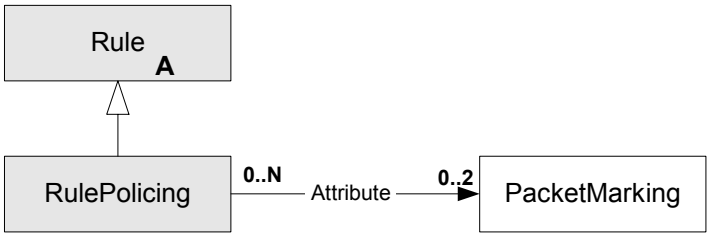
Example

The following command specifies that conforming traffic is to be transmitted and exceeding traffic is to be re-marked to Bronze:

```
Modify ConformAction="Transmit" ExceedAction="TransmitAndRemark"
ExceedPacketMarking="Bronze"
```

If the actions are Drop or Transmit, then the corresponding packet marking has no effect and may be set to anything (including "").

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
CommittedRate	U32	8000	RCW	CAR committed rate (bits per second).
NormalBurstSize	U32	1000	RCW	CAR normal burst size (bytes).
ExcessBurstSize	U32	2000	RCW	CAR excess burst size (bytes).
ExceedAction	Enum	Transmit	RW	Action taken when traffic exceeds specified bandwidth. 1 = Drop 2 = Transmit 10 = TransmitAndRemark 12 = RemarkAndContinue
ConformAction	Enum	Transmit	RCW	Action taken when traffic conforms to specified bandwidth. 1 = Drop 2 = Transmit 10 = TransmitAndRemark 12 = RemarkAndContinue
AclNumber	U32	0	RCW	ACL number to be used for CAR policing. 0=Driver chooses. Valid values if user-specified are 100–199 and 2000–2699.
ConformPacketMarking	String	""	RCW	Name of PacketMarking object used to remark conforming packet will be remarked as if ConformAction is TransmitAndRemark or RemarkAndContinue.

Attribute Name	Type	Default	Access	Explanation
ExceedPacketMarking	String	""	RCW	Name of PacketMarking object used to remark exceeding packets if ExceedAction is TransmitAndRemark or RemarkAndContinue.
Disabled	Boolean	False	RCW	The rule is switched off when Disabled is selected.
ExceedAction	Enum	Transmit	RCW	Specifies the action to be taken when traffic exceeds the agreed bandwidth level.
Inbound	Boolean	False	RCW	Checkbox indicating the traffic direction to which the rule applies. It is 'in' in this case.
InError	Boolean	False	RO	
Name	String	"SimpleTest"	RCW	Identifying name of the policing rule.
Outbound	Boolean	True	RCW	Checkbox indicating the traffic direction to which the rule applies. It is 'out' in this case.
Order	U32	0	RCW	
DaysOfWeek	U32	127	RCW	
EndTime	Date	2020/01/01 00:00:00	RCW	
PeriodName	String	""	RCW	

Attribute Name	Type	Default	Access	Explanation
StartTime	Date	2008/10/21 05:59:16	RCW	
ID	U32	11503	RO	Internal ID number of this object. It is allocated automatically by the system.

ExceedPacketMarking and ConformPacketMarking are Link by Attribute attributes which take names of other objects.

Object inheritance

RulePolicing.Rule.Object

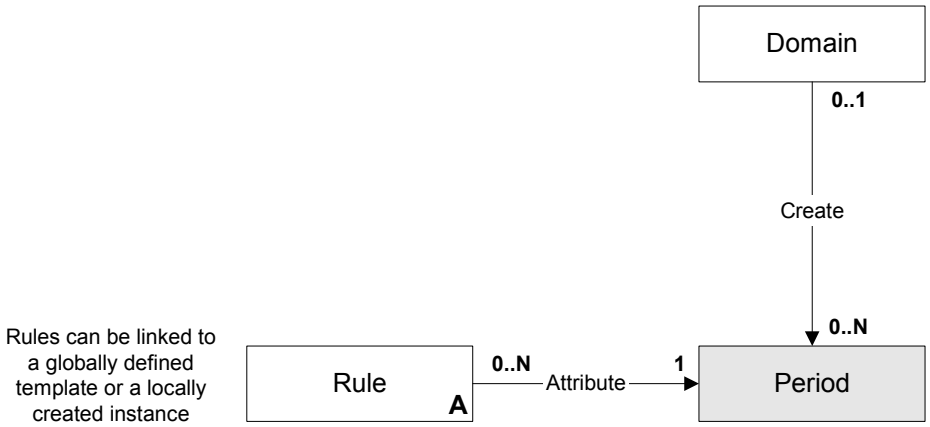
Period object

Description

A Period object identifies the particular dates, times and/or days of the week when a rule is to apply. It can be a one-off period or a repeating time period. The period object can either be a globally-named template or a specific local definition of a date/time period.

The Period specifies (possibly contiguous) recurring spans of time of not more than 24 hours. Each span is the same length. The first span starts at the specified **StartTime**. The last span finishes at the specified **EndTime**. After the first span, subsequent spans start exactly 24 hours after the preceding one started. The length of the span is calculated such that the last span will end at the specified **EndTime**. Spans whose start day of the week in UTC corresponds to a zero bit in **DaysOfWeek** are dropped.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RO	Template name if global, otherwise NULL.
StartTime	DateTime	Now	RO	Start date/time - YYYY/MM/DD HH:MM:SS.
EndTime	DateTime	2020/01/01 00:00:00	RO	End date/time - YYYY/MM/DD HH:MM:SS.
DaysOfWeek	U32	127 (i.e. all days selected)	RO	Bitwise value, 1 bit per day: Bit 0 = Monday Bit 1 = Tuesday Bit 2 = Wednesday Bit 3 = Thursday Bit 4 = Friday Bit 5 = Saturday Bit 6 = Sunday

Object inheritance

Period.Object

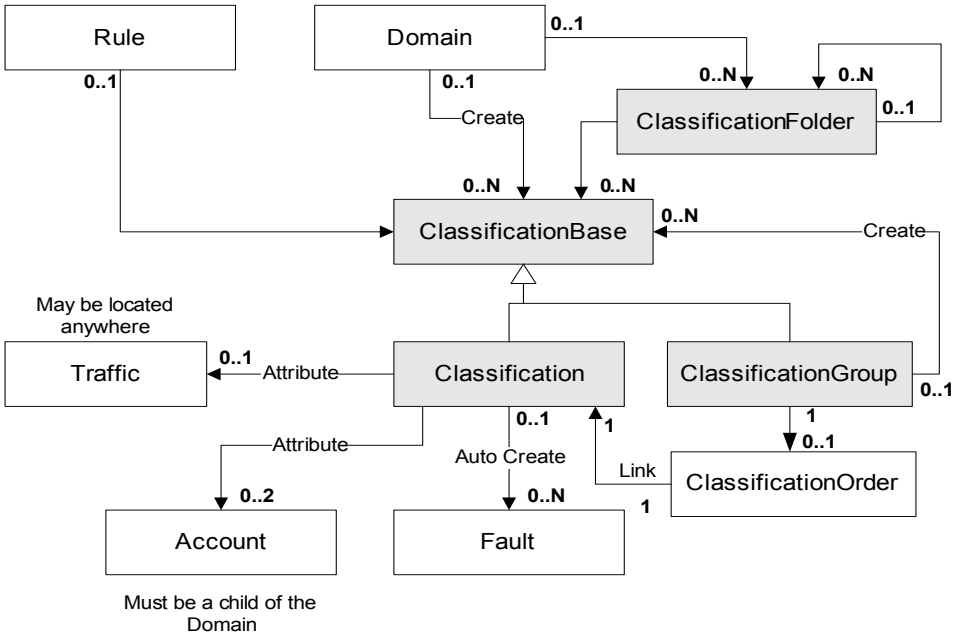
Classification objects

ClassificationBase object (abstract)

Description

The Classification object represents one or many combinations of source addresses and masks, destination addresses and masks and traffic types.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	0	RW	Name of the classification object.
Remarks	String	""	RW	Optional additional comments.

Object inheritance

ClassificationBase.Object

ClassificationGroup object

Description

The ClassificationGroup object is a composite to allow the aggregation of other Classification objects that are commonly used together. A ClassificationGroup object is a child of either a Domain or a ClassificationFolder object, but not both.

Object diagram

See *Object diagram* on page 169.

Attributes

Attribute Name	Type	Default	Access	Explanation
Aggregate	Boolean	True	RW	True if match statements are to be aggregated, otherwise False. Only relevant where the classification is associated with an MQC PHB group via a CoS.
MatchType	Enum	0	RW	Specifies how traffic is matched: 0 = MatchAny 1 = MatchAll Only relevant where the classification is associated with an MQC PHB group via a CoS.
AclIdType	Enum	2	RW	Defines how ACL identifiers are generated: 0 = AutoGenerate 1 = Numbered 2 = Named Only relevant where the classification is associated with an MQC PHB group via a CoS.

Attribute Name	Type	Default	Access	Explanation
AclName	String	Name	RW	Name of the ACL to use. Only relevant where the classification is associated with an MQC PHB group via a CoS.
AclNumber	U32	0	RW	The ACL Number to use. Only relevant where the classification is associated with an MQC PHB group via a CoS.

Object inheritance

ClassificationGroup.ClassificationBase.Object

ClassificationOrder

Description

Re-orders the classifications in the group.

The sequence of Classification objects displayed within a Classification Group reflects the order of evaluation. The first Classification in a group matching a packet directs QoS operations on that packet. After the first match, all following Classifications are ignored.

Attributes

Attribute Name	Type	Default	Access	Explanation
orderNumber	U32	0	RW	Relative ordering of a Classification with respect to a parent ClassificationGroup. 0 - 0xFFFFFFFF, default: 0.
Name	String	Classification Order	RO	Name assigned to the Classification Order.
Id	U32		RO	Unique ID assigned to the Classification Order.

Classification object

Description

The Classification object represents a unique combination of Source Address and Mask, Destination Address and Mask and Traffic Type. A Classification object is a child of either a Domain or a ClassificationFolder object, but not both.

Object diagram

See *Object diagram* on page 169.

Attributes

Attribute Name	Type	Default	Access	Explanation
SourceIpAddr	IPAddress	0.0.0.0	RW	IPv4 host or network address to identify traffic source. 0.0.0.0/Mask 0 = Any Relevant when AddressType=IPv4
SourceMask	IPAddress	0.0.0.0	RW	Network portion of IPv4 address to match. Relevant when AddressType=IPv4.
DestinationIpAddr	IPAddress	0.0.0.0	RW	IPv4 network or host address to identify traffic destination. 0.0.0.0/Mask 0 = Any Relevant when AddressType=IPv4.
DestinationMask	IPAddress	0.0.0.0	RW	Network portion of IPv4 address to match. Relevant when AddressType=IPv4

Attribute Name	Type	Default	Access	Explanation
Aggregate	Boolean	True	RW	True if match statements are to be aggregated, otherwise False. Only relevant where the classification is associated with an MQC PHB group via a CoS.
MatchType	Enum	0	RW	Specifies how traffic is matched against classifications: 0 = Any 1 = All Only relevant where the classification is associated with an MQC PHB group via a CoS.
AclIdType	Enum	2	RW	Defines how ACL identifiers are generated: 0 = AutoGenerate 1 = Numbered 2 = Named Only relevant where the classification is associated with an MQC PHB group via a CoS.

Attribute Name	Type	Default	Access	Explanation
AclName	String	Name	RW	Name of the ACL to use. Only relevant where the classification is associated with an MQC PHB group via a CoS.
AclNumber	U32	0	RW	The ACL Number to use. Only relevant where the classification is associated with an MQC PHB group via a CoS.
AddressType	Enum	IPv4	RW	Selection of address type. Values are: 0 = IPv4 1 = MAC 2 = IPv6
SourceIpv6Addr	String	::/0	RW	IPv6 network or host address to identify traffic source. ::/0 = Any Relevant when AddressType=IPv6
DestinationIpv6Addr	String	::/0	RW	IPv6 network or host address to identify traffic destination. ::/0 = Any Relevant when AddressType=IPv6.

Attribute Name	Type	Default	Access	Explanation
ClassificationMatch	Boolean	TRUE	RW	Whether traffic that matches the specified criteria in the classifier is explicitly included in the processing applied to the classifier or if it is excluded. (corresponds to permit or deny in Cisco ACLs)
SourceMacAddr	String		RW	Identifies the source MAC address by which traffic will be classified.
SourceMacMask	String		RW	Identifies the source MAC mask by which traffic will be classified.
DestinationMacAddr	String		RW	Identifies the destination MAC address by which traffic will be classified.
DestinationMacMask	String		RW	Identifies the destination MAC mask by which traffic will be classified.
Fragments	Boolean	TRUE	RW	When checked, non-initial packet fragments are included for matching. (This relates to the fragments parameter in Cisco ACL statements.)

Attribute Name	Type	Default	Access	Explanation
Log	Boolean	TRUE	RW	Enables / Disables Cisco ACL logging. Headers of packets affected by the access rule are stored in the routing engine and can be displayed using a show log command. Note: Enabling ACL logging may negatively affect router performance: · processor power may be used to perform the logging · log files may impact available disk space
The following attributes can be used to link directly to other objects in the EOM:				
SrcAccountName	String	""	RW	Name of an account to use as source instead of the SourceIpAddress and SourceMask.
DstAccountName	String	""	RW	Name of an account to use as destination instead of the DestinationIpAddr and DestinationMask.
TrafficName	String	""	RW	The name of the traffic object. The traffic object specified may be anywhere in the hierarchy. **Note:** Cannot be a TrafficGroup object.

Object inheritance

Classification.ClassificationBase.Object

ClassificationFolder object

Description

The ClassificationFolder object defines a classification folder, used to contain classification objects, classification group objects, and classification subfolders. A classification folder can be a child of the Domain, or a child of another ClassificationFolder object.

Object diagram

See *Object diagram* on page 169.

Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RW	Name of the Classification folder.
Remarks	String	""	RW	Optional additional comments.

Object inheritance

ClassificationFolder.ClassificationBase.Object

PHBGroup objects

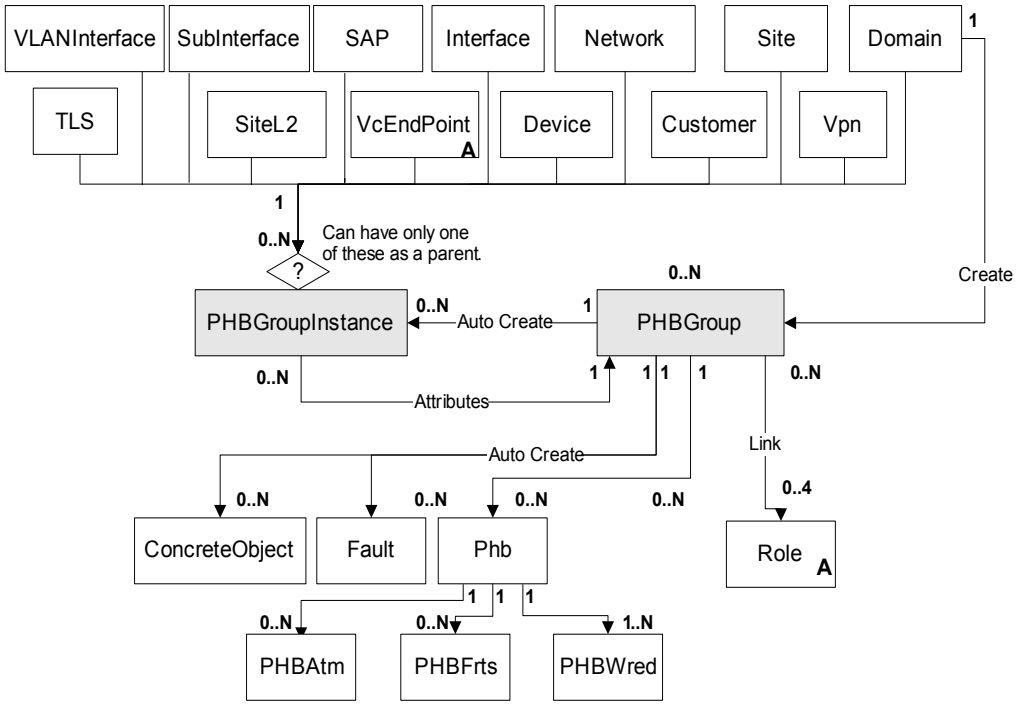
PHBGroup object

Description

The PHBGroup object represents a PHB group – an implementation of a specific queuing/shaping mechanism available at an interface.

> Oracle Communications cannot guarantee compatibility of PHBGroup objects between different releases of Service Activator, unlike most other object types. We therefore strongly recommend that any client software you create encapsulates the data manipulated for PHBGroup objects. This ensures a maintainable code base and minimizes the effect of future changes.
>
> The PHBGroup object is a resource-intensive object and we strongly recommend that you create only a limited number. Each PHBGroup object may be used by multiple PHBGroupInstance objects, effectively acting as a template.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	0	RW	Name of the object.
Description	String		RW	Text entered here is configured as a policy-map description on the device. This behavior is turned off by default. To use this feature, it must be turned on in the capabilities files.
ConfiguredName	String	Value in the **Name** field	RW	Name of the configured PHB Group on the device
Inbound	Boolean	False	RW	True = PHB Group applies to inbound traffic on an interface.
Outbound	Boolean	False	RW	True = PHB group applies to outbound traffic on an interface.
Conflict	Boolean	False	RO	Indicates that this PHB Group is in conflict.
Weight	U32	9	RW	Weight Factor.
ECN	Boolean	False	RW	Modifies WRED performance by marking packets with ECT and ECN bits in the IP header, when congestion occurs in the network.

Attribute Name	Type	Default	Access	Explanation
Action	U32	384 (i.e default WFQ and default WRED)	RW	Bitwise value indicating the queuing mechanism or traffic shaping mechanism to apply: 1 = PQ 2 = TS 4 = WFQ 8 = WRED 16 = WRR 32 = FRTS 64 = ATMQoS 128 = Default WFQ settings 256 = Default WRED settings
WfqAsPercent	Boolean	False	RW	Use WFQ value as a percentage

Object inheritance

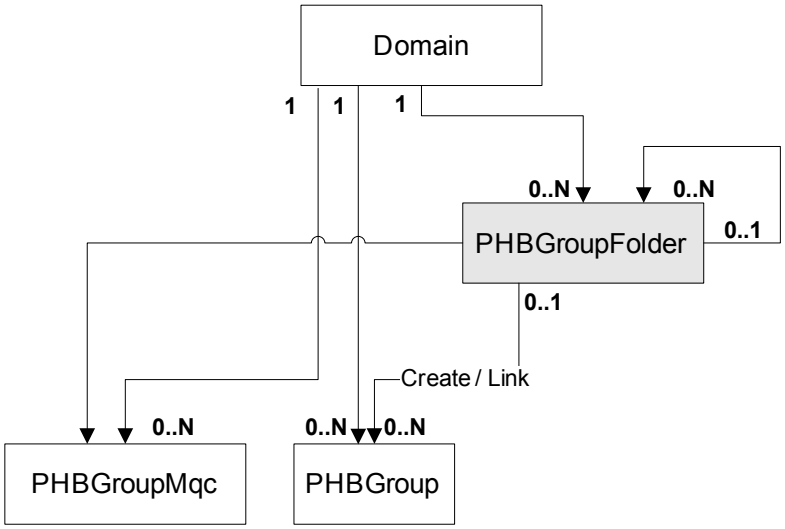
PHBGroup.Object

PHBGroupFolder object

Description

A PHBGroupFolder object represents a folder which contains PHBGroup objects, or other PHBGroupFolder objects, for purposes of organization within the GUI.

Object diagram



A PHBGroup always has either 1 or 2 parents and is always linked to its parent Domain. It may be linked to zero or one parent PHBGroupFolders. In the Service Activator GUI, if a PHBGroup has 2 parents, it will always be displayed under the PHBGroupFolder, not under the PHB Groups folder. If a PHBGroup is created under a folder, it is automatically linked to its parent Domain.

A PHBGroupFolder is a child of either another PHBGroupFolder or the Domain, but not both.

Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RW	Name of the PHBGroup folder.
Remarks	String	""	RW	Optional additional comments.

Object inheritance

PHBGroupFolder.Object

PHBGroupInstance object

Description

The PHBGroupInstance object represents a particular instance of a PHB group. It includes all the attributes of the domain level PHBGroup template and adds the ability to order them, allowing you to define the precedence in case of multiple PHB groups applied at one level.

A PHBGroupInstance is created by the `use` command. If a PHBGroup is used on one of the PHBGroupInstance possible parents, then a PHBGroupInstance will automatically be created as a child of both the PHBGroup and the other object. For more information, see *The use command* on page 37.

Object diagram

See *Object diagram* on page 179.

Attributes

All the attributes of the corresponding PHBGroup are included in this object. In addition, it defines the following new attributes:

Attribute Name	Type	Default	Access	Explanation
Order	U32	0	RW	Order of this PHB group at the local level.
Name	String		RO	The name of the PHB group object that this is an instance of.

Object inheritance

PHBGroupInstance.PHBGroup.Object

PHB object

Description

The PHB object represents the application of a specific queuing mechanism to a class of service.

> The attributes associated with the WFQ queuing mechanism are Read Only in the PHB object. If you wish to configure WFQ, you must use an MQC PHB group. For more information, see *PHBGroupMqc object* on page 190.

Object diagram

See *Object diagram* on page 179.

Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	0	RO	Name of the object. The OIM automatically assigns the value assigned to the object's ClassName attribute.
ID	U32		RO	Unique ID used to reference this object.
Order	U32	0	RCW	Order of this object, compared to other PHB objects.
The following attribute can be used to link directly to a Cos object in the EOM				
ClassName	String	""	RCW	Name of the CoS object to match.
WFQ				
Weight	U32	100	RCW	WFQ weighting to be allocated to CoS.

Attribute Name	Type	Default	Access	Explanation
QueuePriority	Enum	0	RCW	Priority of this queue. Values: 0 = Low (None) 1 = High (Requested)
DropStrategy	Enum	0	RCW	Values: 0 = None 1 = DefaultWred 2 = WRED 3 = TailDrop
Limit	U32	20	RCW	Packet Limit - set if DropStrategy = TailDrop
SetDe	Boolean	False	RCW	True = set the Frame Relay DE bit.
WeightFactor	U32	9	RCW	Specifies an exponent weight factor used in calculating the average queue length. Range = 1-16
Priority Queuing				
Priority	Enum	2	RCW	0 = High 1 = Medium 2 = Normal 3 = Low
Rate Limiting (Traffic Shaping)				
AverageRate	U32	100	RCW	Transmission rate in Kbits/s for the selected class of service.
BurstRate	U32	150	RCW	Transmission burst rate in Kbits/s for the selected class of service.

Attribute Name	Type	Default	Access	Explanation
BurstInterval	U32	10	RCW	Interval, in seconds, over which traffic in the selected class of service is allowed to maintain its maximum burst.
WRR				
Bandwidth	U32	2500	RCW	Bandwidth weighting for this class of service.
PacketLimit	U32	20	RCW	Packet Limit.

PHBAtm object

Description

The PHBAtm object represents the application of an ATM traffic shaping mechanism to a VC endpoint.

Object diagram

See *Object diagram* on page 179.

Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	"PhbAtm"	RO	Identifier used to access the object.
Type	Enum	0	RW	Traffic shaping mechanism to apply: 0 = UBR 1 = CBR 2 = RtVBR 3 = NrtVBR 4 = ABR
PCR	U32	22500	RW	Peak Cell Rate. Used in all service classes. Defined in bits/s. Range 1-45 000 000

Attribute Name	Type	Default	Access	Explanation
SCR	U32	22500	RW	Sustainable Cell Rate. Used for VBR. Defined in bits/s. Range: 1–45 000 000
MBS	U32	32700	RW	Maximum Burst Size. Used for VBR. Specifies the largest burst of data above the ensured rate that will be allowed temporarily on the PVC. Range: 0–65535
MCR	U32	22500	RW	Minimum Cell Rate. Used for ABR. Specifies the minimum value for the ACR. Defined in bits/s. Range: 1–45 000 000.
HoldQueueDepth	U32	0	RW	Maximum number of packets that can be stored in the traffic-shaping queue for an ATM PVC. Range: 5–1024. 0=device default.
TransmitRingLimit	U32	0	RW	Allows the hardware queue depth to be configured. Range: 3–6000. 0=device default.
VcClassName	String		RW	VC class name. The maximum length of the name is 126 characters.

PHBFrts object

Description

The PHBFrts object represents the application of a Frame Relay traffic shaping mechanism to an interface.

Object diagram

See *Object diagram* on page 179.

Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	"PhbFrts"	RO	Identifier used to access the object.
Shaping	Boolean	True	RCW	Indicates whether shaping is to be performed.
CIR	String	56000	RCW	Committed Information Rate (bits per second) Range: 1–45 000 000
CIR_In	String	56000	RCW	Committed Information Rate (bits per second) applied as an inbound override for CIR Range: 1–45 000 000
CommittedBurst	String	56000	RCW	Committed burst size (bits). Range: 1000– 160 000 000
CommittedBurst_in	String	56000	RCW	Inbound override for Committed burst size (bits). Range: 1000– 160 000 000
ExcessBurst	String	0	RCW	Excess burst size (bits) Range: 0–16 000 000

Attribute Name	Type	Default	Access	Explanation
ExcessBurst_In	String	0	RCW	Inbound override for excess burst size (bits) Range: 0–16 000 000
MinCIR	String	28000	RCW	Minimum CIR (bits per second). Range: 1000–45 000 000
MinCIR_In	String	28000	RCW	Minimum CIR (bits per second) applied as an inbound override for MinCIR. Range: 1000–45 000 000
BECNAdapt	Boolean	True	RCW	True = adapt to Backward Explicit Congestion Notification flag.
FECNAdapt	Boolean	True	RCW	True = adapt to Forward Explicit Congestion Notification flag.
FRF12Fragment	Boolean	True	RCW	True = Use FRF.12 fragmentation flag.
FRF12Fragment Size	U32	53	RCW	FRF.12 fragment size (bytes). Range: 16–1600
HoldQueueDepth	U32	0	RCW	Maximum number of packets that can be stored in the traffic-shaping queue for a Frame Relay PVC. Range: 1–2048. 0 = device default.
ID	U32	22795	RO	

PHBWred object

Description

The PHBWred object represents the application of a WRED mechanism to an interface. It is created as a child of a PHB object automatically when the Class of Service that the PHB object represents has Packet Markings attached to it. One PHBWred object is created for each Packet Marking.

A default PHBWred object is created when the 'Default' drop strategy is selected.

Object diagram

See *Object diagram* on page 179.

Attributes

Attribute Name	Type	Default	Access	Explanation
MaxDrop	U32	10	RCW	Minimum threshold value for packet discard for selected CoS.
MinDrop	U32	1	RCW	Maximum threshold value for packet discard for selected CoS.
Denominator	U32	10	RCW	Value for Drop probability.

PHBGroupMqc objects

PHBGroupMqc object

Description

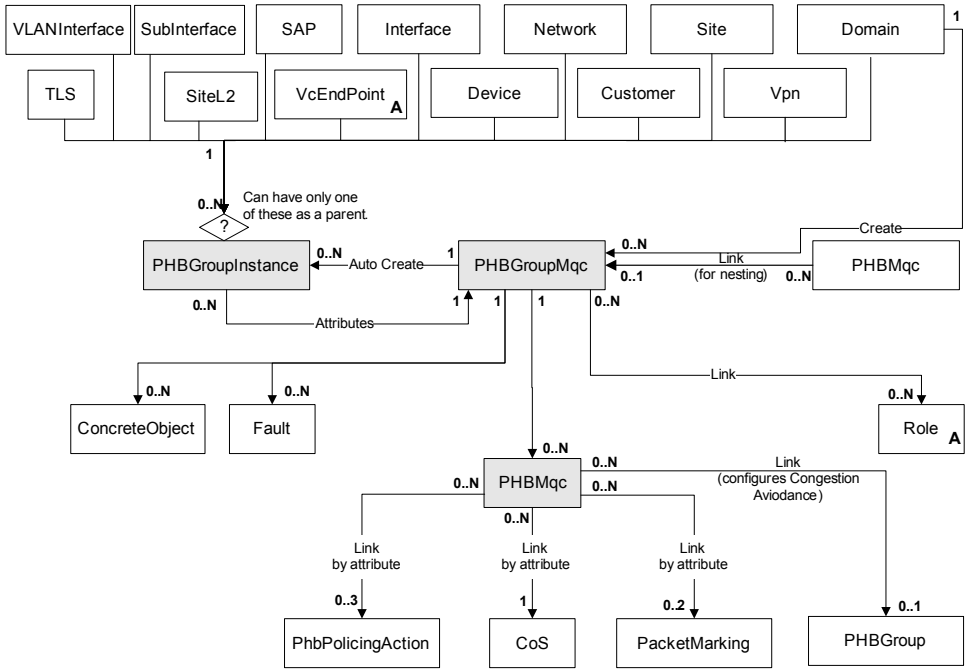
The PHBGroupMqc object represents an MQC PHB group – the application of a queuing/shaping mechanism via Cisco's Modular QoS CLI.

The PHBGroupMqc object defines generic parameters. The PHBMqc object defines the specific MQC parameters to be applied to a particular CoS handled by an interface.

A PHBGroupMqc object may be linked to multiple PHBMqc objects.

When defining a PHBMqc object to apply congestion avoidance, the drop mechanism may be defined by a standard WRED PHB group.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	0	RW	Name of the object.
ConfiguredName	String	Value in the **Name** field	RW	Name of the configured MQC PHB group on the device.
Description	String		RW	Text entered here is configured as a policy-map description on the device. This behavior is turned off by default. To use this feature, it must be turned on in the capabilities files.

Attribute Name	Type	Default	Access	Explanation
MaxReservedBandwidth	U32	0	RW	Cisco devices have a default maximum reserve bandwidth value of 75 percent that is designed to leave sufficient bandwidth for overhead traffic. Specify a new value if required.
Inbound	Boolean	False	RW	True = Mechanism applies to incoming traffic.
Outbound	Boolean	True	RW	True = Mechanism applies to outgoing traffic.
Conflict	Boolean	False	RO	True = MQC PHB group is failing. False = MQC PHB group is not in error.
LlqBandwidthType	Enum	0	RW	Indicates how LLQ Bandwidth value is interpreted: 0 = Absolute 1 = Percentage 2 = PercentageRemaining
WfqBandwidthType	Enum	0	RW	Indicates how LLQ Bandwidth value is interpreted: 0 = Absolute 1 = Percentage 2 = PercentageRemaining

PHBMqc object

Description

The PHBMqc object represents the application of a specific MQC mechanism to a class of service.

Object diagram

See *Object diagram* on page 191.

Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	0	RO	Name of the object. The OIM automatically assigns the value assigned to the object's CosName attribute.
Order	U32	0	RW	Order of this object, compared to other PHBMqc objects.
AggregatePolicer	Boolean	FALSE	RW	Set AggregatePolicer to true to activate aggregate policing.
PolicerName	String		RW	Specify the aggregate policer name when AggregatePolicer is true.
PoliceRateType	Enum	Absolute	RW	The police rate can be specified in absolute terms (bits/second) or as a percentage of the total available bandwidth. Valid choices are: absolute, percent
VipFairQueue	Boolean	FALSE	RW	When set to true, normal weighted fair queuing is selected for the CoS.

Attribute Name	Type	Default	Access	Explanation
VipFlowQueueLimit	U32	0	RW	Specifies queue limit in number of packets for fair-queuing.
TrustMarking	String	0	RW	This configures the trust state, which selects the value that QoS uses as the source of the internal DSCP value.
The following attribute can be used to link directly to a Cos object in the EOM				
CosName	String	0	RW	Name of the Class of Service to be used.
Action	U21	0	RW	A bitfield describing the MQC action(s) specified: 0 = None 1 = QueueLLQ 2 = QueueCBWFQ 4 = PoliceSingleRate 8 = PoliceTwoRate 16 = Shape 32 = Mark 64 = Congestion 128 = Nest 512=Default WFQ
Queue				
LLQWeight	U32	100	RW	Treated as a percentage or an absolute value depending on the value of the LlqBandwidthType attribute on the **PhbGroupMqc** object. If expressed as a percentage, the range is 1–100. If expressed as an absolute value, the range is 32–155 000 (Kbits/s)

Attribute Name	Type	Default	Access	Explanation
LLQBurst	U32	0	RW	Configures the LLQ to accommodate temporary bursts of traffic. Range: 32–2 000 000 bytes. 0 = device default.
WFQWeight	U32	100	RW	Treated as a percentage or an absolute value depending on the value of the WfqBandwidthType attribute on the **PhbGroupMqc** object. If expressed as a percentage, the range is 1–100. If expressed as an absolute value, the range is 8–2 000 000 (Kbits/s)
Single Rate Policing				
SR_CIR	U32	8000	RW	Committed information rate in bits/s. Range: 8000–4 000 000 000
To load the device default value for **SR_CBS**, send the special value 4 294 967 295.				
SR_CBS	U32	1000	RW	Committed burst size in bytes. Range: 1000–512 000 000
To load the device default value for **EBS**, send the special value 4 294 967 295.				
EBS	U32	1000	RW	Excess burst size in bytes. Range: 1000–512 000 000 Used only if ViolateAction is specified within Policing Action.

Attribute Name	Type	Default	Access	Explanation
Two Rate Policing				
TR_CIR	U32	8000	RW	Committed information rate in bits/s. Range is 8000-4 000 000 000.
TR_CBS	U32	1000	RW	Committed burst size in bytes. Range is 1000–512 000 000.
PIR	U32	8000	RW	Peak information rate in bits/s. Range is 8000–4 000 000 000.
PBS	U32	8000	RW	Peak burst size in bytes. Range is 1000–512 000 000.
Policing Action				
ConformAction	String	""	RW	Name of a **PolicingAction** object used to define the action to perform on conforming traffic.
ExceedAction	String	""	RW	Name of a **PolicingAction** object used to define the action to perform on exceeding traffic.
ViolateAction	String	""	RW	Name of a **PolicingAction** object used to define the action to perform on violating traffic.
Shaping				
BasicShaping	Boolean	False	RW	True = When selected, applies basic traffic shaping based on the CIR value only. Applies to Cisco 10000 devices only.

Attribute Name	Type	Default	Access	Explanation
Shape_CIR	U32	56000	RW	CIR in bits/s. Range is 8000–154 400 000, must be a multiple of 8000.
DefaultBc	Boolean	True	RW	True = Use the device default for the Bc value. False = Specify a Bc value.
Bc	U32	28000	RW	The normal burst size in bits. Range: 32–154 400 000.
DefaultBe	Boolean	True	RW	True = Use the device default for the Be value. False = Specify a Be value.
Be	U32	0	RW	The peak burst size in bits. Range: 0–154 400 000
ShapeAverage	Boolean	True	RW	True = Use Average rate traffic shaping False = Use Peak rate traffic shaping.
ShapingBuffers	U32	0	RW	Number of shaping buffers available to the outgoing queues used for shaping. Range: 1–4096. 0 = device default.
FrExtension	Boolean	False	RW	True = Enable the Frame Relay shaping extensions (MinCIR, BECNAdapt, FECNAdapt).

Attribute Name	Type	Default	Access	Explanation
MinCIR	U32	56000	RW	Minimum rate to which traffic will be throttled in response to BECN and fecn messages. Range: 8000–154 400 000 The value must be a multiple of 8000. Only valid if FrExtension is True.
BECNAdapt	Boolean	False	RW	True = Shaping adapts to BECN. Only valid if FrExtension is true.
FECNAdapt	Boolean	False	RW	True = Shaping adapts to FECNs. Only valid if FrExtension is true.
Marking				
SetDe	Boolean	False	RW	True = set FR DE False = don’t set FR DE.
SetAtmClp	Boolean	False	RW	True = set ATM CLP, False = don’t set ATM CLP.
DscpMarking	String	False	RW	Name of a PacketMarking object representing the DiffServ Codepoint to use.
IPPrecedenceMarking	String	False	RW	Name of a PacketMarking object representing the IP Precedence to use.
MplsMarking	String	False	RW	Name of a PacketMarking object representing the MPLS Experimental marking to use.
discardclassmarking	String	False	RW	Name of a PacketMarking object representing the Discard Class to use.

Attribute Name	Type	Default	Access	Explanation
Congestion Avoidance				
QueueLimit	U32	0	RW	Limit to apply in packets. The permitted range varies, depending on the device. Check the capabilities. A value of 0 applies the device's default limit.
WredStrategy	Enum	0	RW	How packets are discarded when congestion occurs 0 = None 1 = Default 2 = WRED. In this case, the **PhbMqc** object must be linked to a **PhbGroup** object.
Classification				
ClassMapMatching	Enum	0	R	0 = match-any 1 = match-all
Nesting				
NestedPhbGroup	String			Name of a **PhbGroupMqc** object to be applied to this Class of Service.

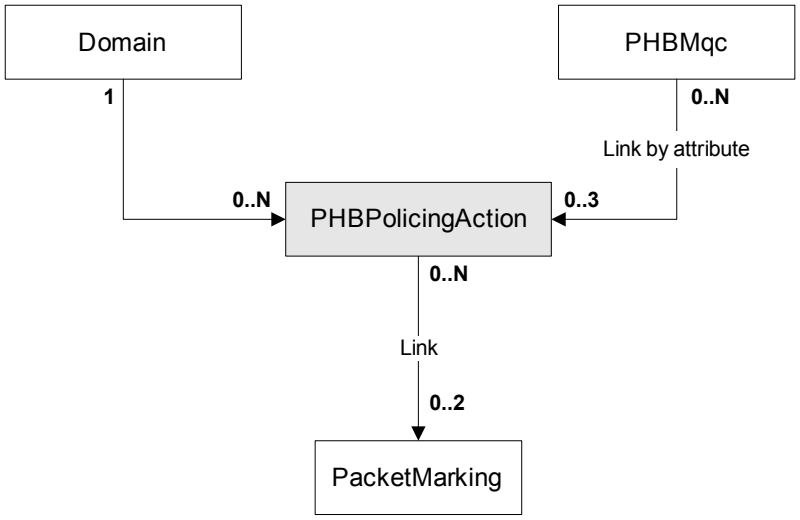
PHBPolicingAction object

Description

The PHBPolicingAction object defines a policing action for use with an MQC PHB group that applies policing. If the object defines a set action using DiffServ

codepoints, IP Precedence or MPLS experimental bits, one or two PacketMarking objects must be linked to the PHBPolicingAction object.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RW	Name of the object.

Attribute Name	Type	Default	Access	Explanation
Type	Enum	0	RW	Enumeration indicating whether to apply a default or user-defined policing action: 0 = User 1 = DefaultConform 2 = DefaultExceed 3 = DefaultViolate
Actions	U32	0	RW	Bitwise value, indicating action(s) to take. 0 = Drop 1 = Transmit 2 = SetClpTransmit 4 = SetFrDeTransmit 8 = SetDscpTransmit 32 = SetIPPrecedenceTransmit 16 = SetMplsExpTransmit

Object inheritance

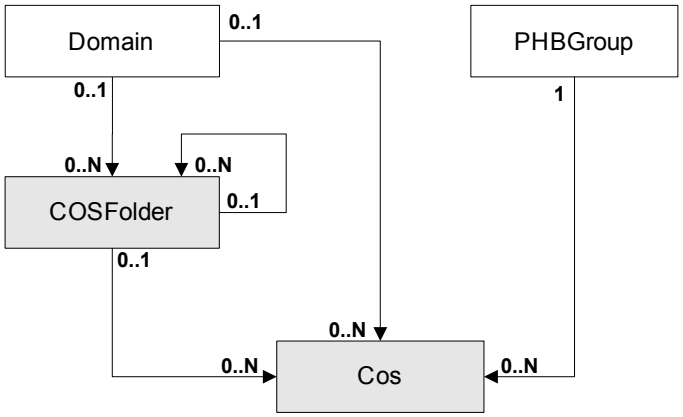
PHBPolicingAction.Object

Cos object

Description

The Cos object defines a class of service. The Cos object is a child of the Domain or a COSFolder, but not both.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	0	RW	The name of the class of service.
ConfiguredName	String	Value in the **Name** field	RW	Name of the configured Class of Service on the device.
Remarks	String	""	RW	Free-format text field, to add a description of the class of service; max length 200 characters
Type	Enum	0	RW	0 = User 1 = Default (IP Class of Service) 2 = NonIP

Object inheritance

Cos.Object

COSFolder object

Description

The COSFolder object defines a class of service folder, used to contain class of service (CoS) objects and class of service subfolders. A COSFolder is a child of the Domain or another COSFolder object, but not both.

Object diagram

See *Object diagram* on page 202.

Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RW	Name of the class of service folder
Remarks	String	""	RW	Optional additional comments

Object inheritance

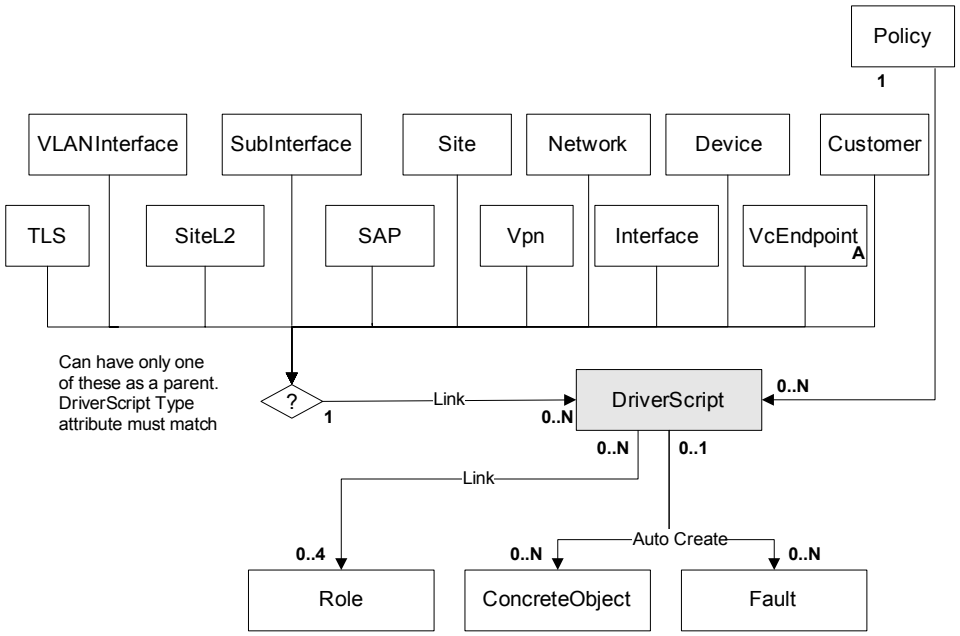
COSFolder.Cos.Object

DriverScript object

Description

Represents a script, written in Python, that when applied to a device, results in the generation of a command script used to configure a device directly.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	Unique	RW	Name of the driver script.
Script	String	""	RW	The text of the Python script. Note that when creating DriverScript objects via the OIM CLI, it is possible to pass the content of a Python script using the following syntax: Script="*pathname*.py" where *pathname* is the full path of the script. Note that no spaces are permitted in *pathname* and the script must have a .py extension. For example: Script="c:\nvram.py"
Type	Enum	0	RW	Type of object this script applies to: 0 = Device 1 = Interface 2 = SubInterface 3 = AtmPvc 4 = FrPvc
DeviceDriverType	String	cisco	RW	Indicates which device driver will run the script.
When	U32	1	RW	When script is applied: 0 = Before standard configuration changes. 1 = After standard configuration changes.

Attribute Name	Type	Default	Access	Explanation
Repeat	Boolean	False	RW	False = Apply once only. True = Repeat on each propagate.
ReApply	Boolean	False	RW	False = Do not re-apply. True = Repeat on next propagate only. The attribute is automatically set to False after the next propagate.
OnRestart	Boolean	False	RW	False = Don't apply on a device restart. True = Apply on each device restart.
Order	U32	0	RW	Script Order. Lowest number (0) indicates highest priority. Note that if this is changed for one script, all scripts with lower priority numbers are renumbered.
Disabled	Boolean	False	RW	True = script is to be disabled at next propagate. False = script not to be disabled.
InError	Boolean	False	RO	True = script is currently failing. False = script is not in error.

Attribute Name	Type	Default	Access	Explanation
CreateConcretesOnce	Boolean	False	RW	This Boolean is only used for 'Run Once' type scripts. When set to true, it stops the creation of new concretes after the script's first execution.
HasRun	Boolean	False	RO	Service Activator sets this value when running the script.

Object inheritance

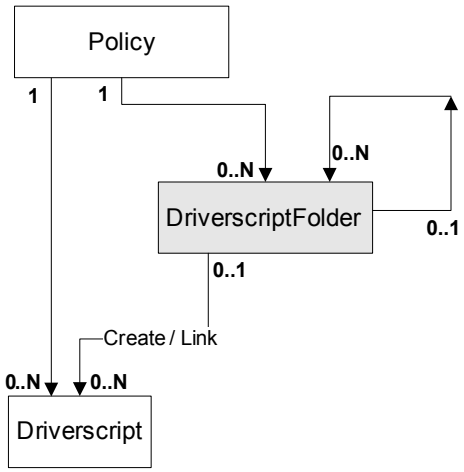
DriverScript.Object

DriverscriptFolder object

Description

A DriverscriptFolder object represents a folder which contains Driverscript objects, or other DriverscriptFolder objects, for purposes of organization within the GUI.

Object diagram



A Driverscript always has either 1 or 2 parents and is always linked to its parent Policy. It may be linked to zero or one parent DriverscriptFolders. In the Service

Activator GUI, if a Driverscript has 2 parents, it will always be displayed under the DriverscriptFolder, not under the Driver Scripts folder. If a Driverscript is created under a folder, it is automatically linked to its parent Policy.

Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RW	Name of the Driverscript folder.
Remarks	String	""	RW	Optional additional comments.

Object inheritance

DriverscriptFolder.Object

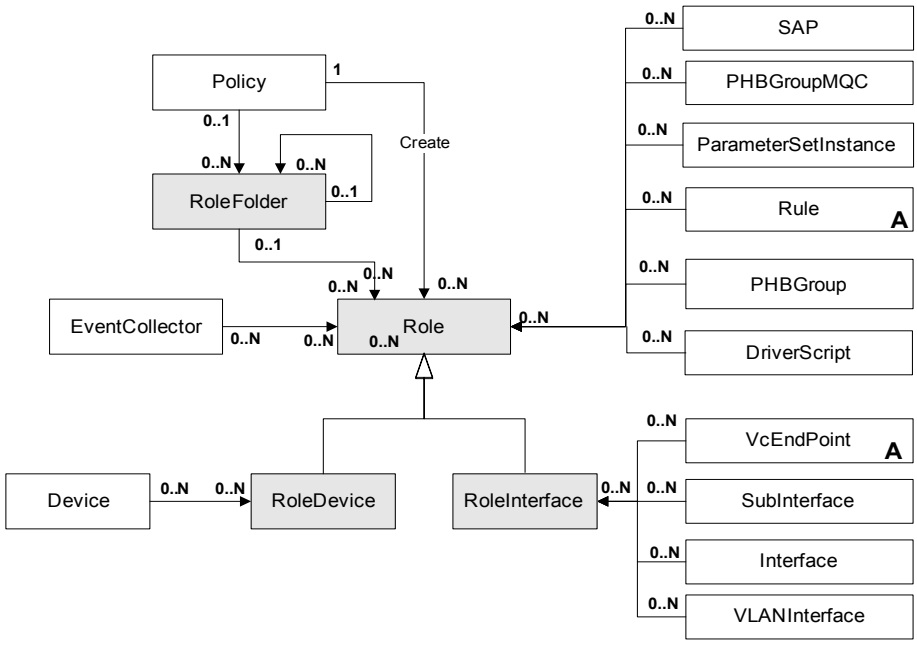
Role objects

Role object (Abstract)

Description

The Role object is used to define the role assigned to a configured object. It is used to determine the targets that policy elements (rules, PHB groups and driver scripts) apply to. Note that rules and PHB groups can have up to four roles (system-defined device role, user-defined device role, system-defined interface role and user-defined interface role).

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RC	Name of the role.
Remarks	String	""	RC	Optional additional comments.

Object inheritance

Role.Object

RoleFolder object

Description

The RoleFolder object defines a role folder, used to contain interface and device role objects and role subfolders. A role folder is the child of the Policy object or another RoleFolder object, but not both.

Object diagram

See *Object diagram* on page 209.

Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RW	Name of the role folder.
Remarks	String	""	RW	Optional additional comments.

Object inheritance

RoleFolder.Role.Object

RoleDevice object

Description

The RoleDevice object is used to define the role assigned to devices. RoleDevice objects can only be linked to Device objects. The RoleDevice object is a child of either the Policy object or a RoleFolder, but not both.

Object diagram

See *Object diagram* on page 209.

Attributes

Attribute Name	Type	Default	Access	Explanation
Type	Enum	0	RO	Defines the type of device role: 0 = UserDefined 1 = Access 2 = Gateway 3 = Core 4 = Any 5 = Shadow

Object inheritance

RoleDevice.Role.Object

RoleInterface object

Description

The RoleInterface object is used to define the role assigned to a configured object. RoleInterface objects can be linked to Interface, SubInterface and VcEndPoint objects. The RoleInterface object is a child of either the Policy object or a RoleFolder, but not both.

Object diagram

See *Object diagram* on page 209.

Attributes

Attribute Name	Type	Default	Access	Explanation
Type	Enum	0	RO	Defines the type of interface role: 0 = UserDefined 1 = Core 2 = Local 3 = Access 4 = Disabled 5 = Any

Object inheritance

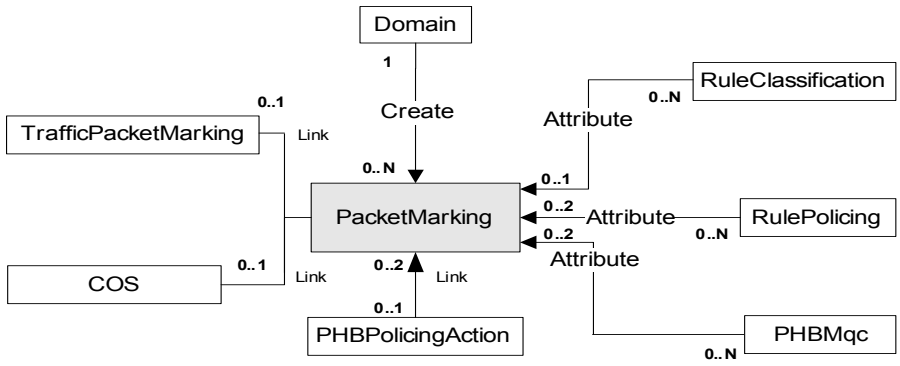
RoleInterface.Role.Object

PacketMarking object

Description

The PacketMarking object defines a type of packet marking, which may be a DiffServ codepoint, an IP Precedence value, an MPLS Experimental value, a Frame Relay DE bit setting or an ATM CLP bit setting.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RW	Name of the object.
MarkingType	Enum	0	RW	0 = DSCodepointValue 1 = MplsHeader 2 = FrDe 3 = AtmClp 4 = AlcatelIntQ 5 = AlcatelIntQ3CoS 6 = AlcatelIntQ1CoS 7 = IPPrecedence 8 = DiscardClass 9 = Trust
MarkingValue	String	0	RW	This value depends on the MarkingType, and defines how to mark traffic. The range is 0–7 for MPLS headers IP Precedence and DiscardClass; 0–63 for DiffServ codepoint.
Id	U32	0	RO	Unique ID used to reference this object

Object inheritance

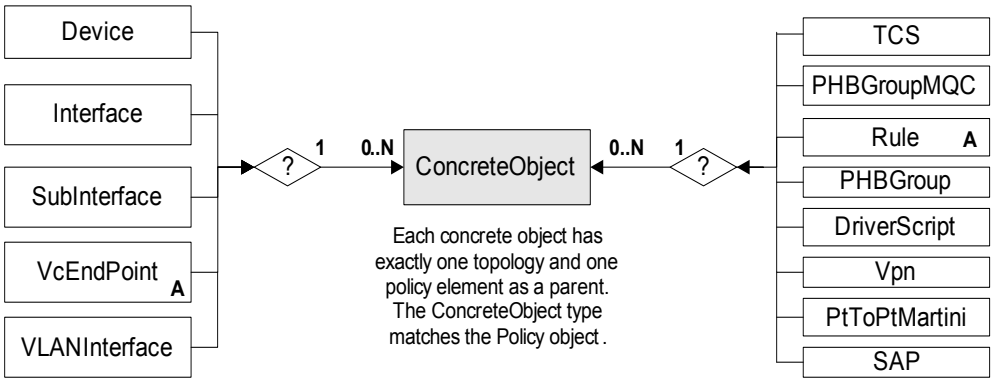
PacketMarking.Object

ConcreteObject object

Description

Concrete objects are automatically created and represent the actual application of a Rule, PHB group, VPN or Driver Script to a specific point in the network. Each rule, PHB group, VPN or Driver Script may result in a number of ConcreteObjects.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RO	Name of the concrete object.
Type	String	"Vpn"	RO	Type of object that created this concrete object. Can be: "RulePolicing" "RuleClassification" "RuleAccess" "DriverScript" "PHBGroup" "Vpn"
Enabled	Boolean	True	RCW	True = policy element is active. False = policy element is disabled.
Conflict	Boolean	False	RO	True = policy element is failing. False = policy element is not in error.
Direction	Enum	Outboun d	RO	0 = Inbound 1 = Outbound

Attribute Name	Type	Default	Access	Explanation
State	Enum	0	RO	0 = Inactive 1 = Active 2 = Installed 3 = Failed 4 = Finished 5 = RunFailed 6 = RunOnceFailed 7 = Uninstalled 8 = UninstallFailed Uninstalled indicates successful removal of a policy from the device, after the policy concrete was disabled. UninstallFailed indicates failure to remove a policy from the device, after the policy concrete was disabled. The policy remains on the device, and its concrete remains disabled in the GUI.
VpnOrder	U32	0	RO	When the object is a concrete VPN, this is a unique ID used to reference this object.
AuditMismatchIgnored	Boolean	False	RCW	Set this Boolean so you can ignore an audit mismatch that has been fixed, or was never a real problem. (Set to ignore the mismatch when you do not plan to rerun the device audit again soon.) Unset this Boolean when you no longer wish to ignore an audit mismatch.
AuditState	Enum	NotAvailable	RO	Read-only field displays Passed, Failed, or Not Available
NotificationCount	U32		RO	Concrete state notification counter
ExternalId	U32		RO	External ID of the concrete.

Object inheritance

ConcreteObject.Object

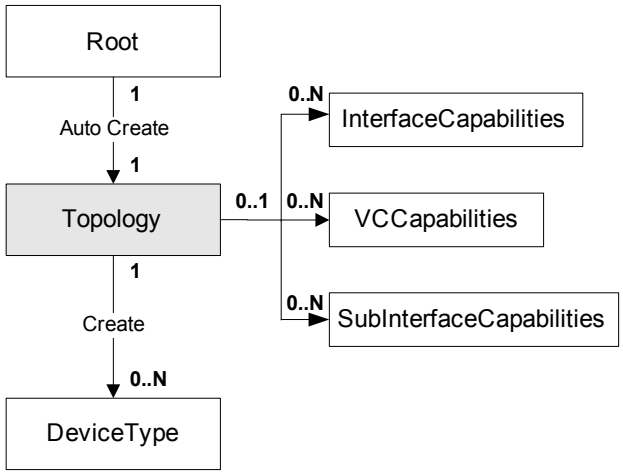
The Topology model

Topology object

Description

The Topology object represents the root of the entire Topology tree.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	"Topology"	RO	Always “Topology”.
ReadCommunity	String	public	RW	SNMP community string for Read Access
DiscoveryInProgress	Boolean	False	RO	True when a discovery request is made, set back to false, when request satisfied

Object inheritance

Topology.Object

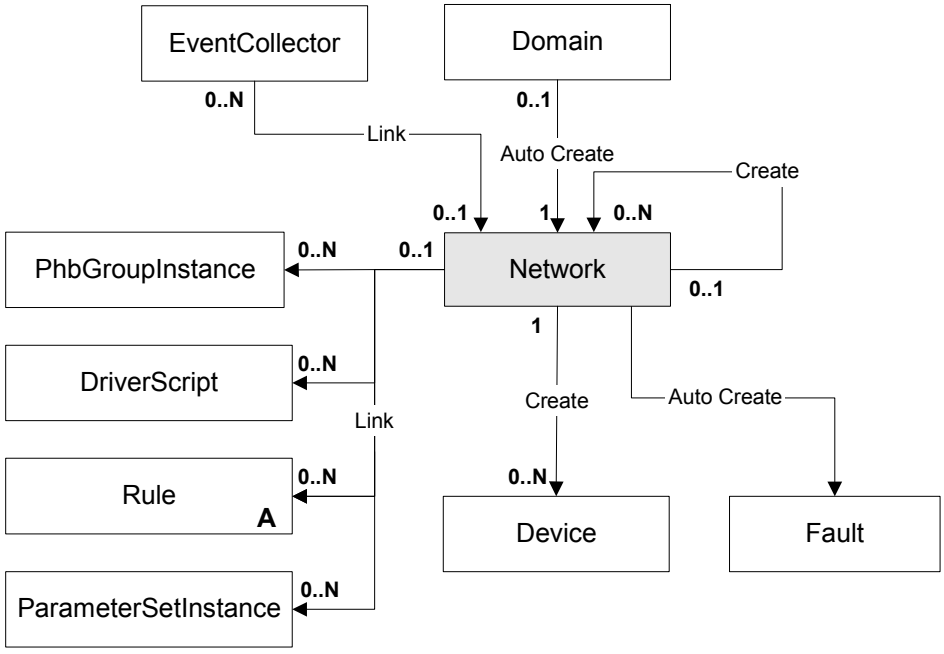
Network object

Description

Each domain has one root-level network object associated with it, which is automatically created with the domain and deleted when the domain is deleted.

A domain's network can be further partitioned by creating further levels of network objects under the root network object. A network needs to be linked either to a domain or to another network object. Each device within the domain is assigned to one network object.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RO	Name of network. Root network takes the name of the domain by default.
Description	String	""	RW	Free-format comments about the network.
BgpAsn	U32	0	RW	Border Gateway Protocol, Autonomous System Number.
InheritsBgpAsn	Boolean	False	RW	True=Inherits ASN from parent object. False=Does not inherit ASN.
WriteCommunity	String*	""	RO	The SNMP write community to use when fetching capabilities.
Username	String*	""	RO	The username for login when fetching capabilities.
Password	String*	""	RO	The password for login when fetching capabilities.
EnablePassword	String*	""	RO	The enable password for login when fetching capabilities.
RsaPrivateKey	String*	""	RO	Name of private key file when fetching capabilities.
InheritsSecurity	Boolean	False	RO	True = Discovery will use the security attributes of the network object rather than the values entered as parameters to the discover command. False = Discovery does not inherit security settings.

Attribute Name	Type	Default	Access	Explanation
Context	String	""	RW	Local context for driver scripts applied at network level (max 512 bytes).
MaxTransactionSize	U32	0	RW	Maximum number of matches of Configuration Threshold regex pattern allowed in a device configuration session.
IgnoresTransactionSize	Boolean	False	RW	True = Configuration Thresholding is turned on. False = Configuration Thresholding is turned off.
InheritsTransactionSize	Boolean	False	RW	True = all Configuration Thresholding parameters (not just the maximum transaction size) are inherited from parent network. False = parameters are not inherited.
MatchesPatternTransactionSize	String	""	RW	Configuration Threshold regex pattern (limit 127 characters)

* Encrypted string

Object inheritance

Network.Object

Modifying the regular expression for MatchesPatternTransactionSize

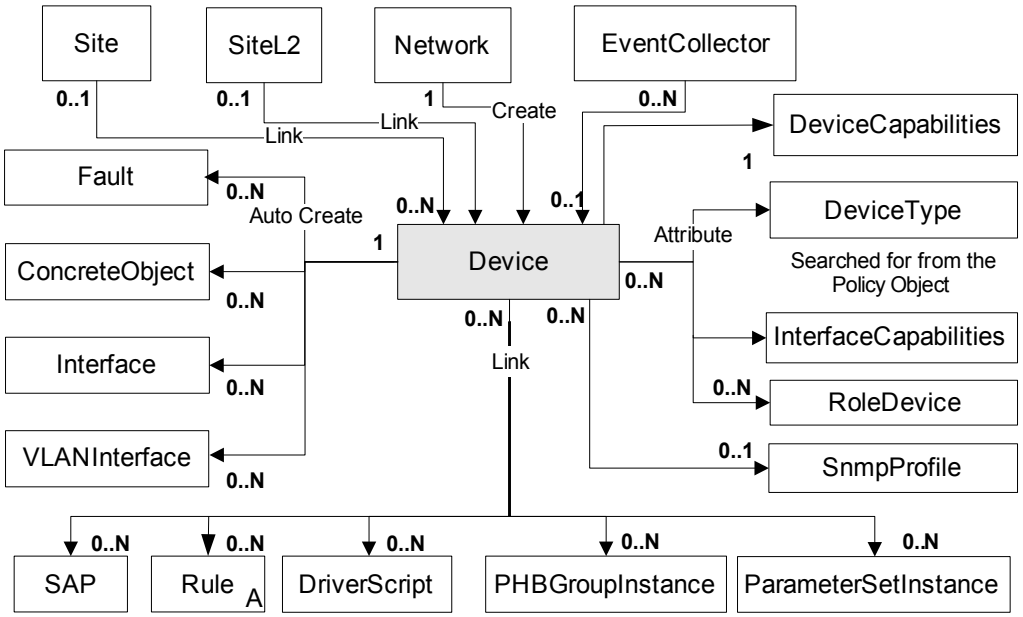
Refer to the topic *Configuration Thresholding feature - modifying the regular expression* on page 378.

Device object

Description

The Device object is used to represent a network node that forwards IP packets, that is, a router or Layer 3 switch, rather than an end system host or server.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RW	Name of device. This is taken from the node name, which is set from the MIB value sysname.0.
Description	String	""	RC	SNMP sysDescription.
Location	String	""	RC	SNMP sysLocation.

Attribute Name	Type	Default	Access	Explanation
IpAddr	IPAddress	0.0.0.0	RW	Address used to communicate with the device.
LoopbackAddr	IPAddress	0.0.0.0	RW	IP address of the loopback interface.
WriteCommunity	String*	""	RC	SNMP Write community.
ReadCommunity	String*	""	RC	SNMP Read community.
IsVirtual	Boolean	True	CO	True = Creates a virtual device. Can be set on create and does not appear on the object. False = not a virtual device.
Username	String*	""	RW	Username.
LoginPassword	String*	""	WO	Login Password. Required if AccessStyle set to NamedUser, Anonymous, SSH or passwordOnly.
EnablePassword	String*	""	WO	Enable Password. Required if AccessStyle set to NamedUser, Anonymous, or SSH.
InheritsSecurity	Boolean	True	RW	True = device security options are inherited from the network object. False = security settings are specific for the device.
State	Enum	7	RO	Last known device state: 0 = Down 1 = Unmanaged 2 = New 3 = NotFound 4 = PreProvisioned 5 = Managed 6 = InterventionRequired 7 = Virtual

Attribute Name	Type	Default	Access	Explanation
UnmanagedAction	Enum	2	RW	The action to take when unmanaging the device and any of its interfaces: 0 = RemoveConfiguration 1 = LeaveConfiguration 2 = UseGlobalSettings
SnmpVn	Enum	3	RW	Version of SNMP to use to interrogate the device: 0 = None 1 = SnmpV1 2 = SnmpV2c 3 = SnmpV1V2c
SnmpRetries	U32	2	RW	The number of retry attempts when sending a PDU (Protocol Data Unit). Range: 0–20
SnmpTimeout	U32	3	RW	Timeout of PDU response Range: 1–30 seconds
SysObjectId	String	""	RC	SNMP obtained unique object identifier.
UpTime	U32	Now	RC	Date and Time since the device is available (seconds).
CommandDelivery Mode	Enum	0	RW	0 = Online: commands delivered to device 1 = OfflineMaintenance: commands not delivered, concretes and other states updated 2 = OfflineTest: commands not delivered

Attribute Name	Type	Default	Access	Explanation
AccessType	Enum	5	RW	0 = NamedUser 1 = Anonymous 2 = TACACS 3 = SNMPv1 4 = SNMPv2c 5 = None 6 = SSH 8 = RSA_SSH 9 = passwordOnly
RsaPrivateKey	String*	""	RO	Name of SSH RSA private key file (max length 3072 characters).
Context	String	""	RW	Local context for driver scripts applied to devices (max 512 bytes).
ManualConfig Mode	Enum	3	RW	Action on this device for detecting manual configuration: 0 = Disable 1 = Warn 2 = Fail 3 = Inherit
UseSaaIpAddr	Boolean	False	RW	Indicates if SaaIpAddr is to be used as destination IP address for SAA. If set to false, IpAddr will be used instead.
SaaIpAddr	Boolean	0	RW	Destination IP address for SAA.
SaaSourceIpAddr	Boolean	0	RW	Source IP Address for SAA
SaaSourceIpAddr Mode	Enum	NotConfigured	RW	Valid modes are: 0=NotConfigured, 1=SameAsDestination, 2=DeviceManagementAddress, 3=SpecifiedAddress. When choosing 3, SaaSourceIpAddr will be used as source IP address for SAA.
AuditId	U32		RO	

Attribute Name	Type	Default	Access	Explanation
AuditState	Enum	NotAvailable	RO	NotAvailable, Passed, Failed, Error
CliPort	U32	23	RW	CLI/Telnet Port to talk to device (for example, Telnet Port 23).
bgpLocalAs	U32	Set by discovery process	RC	Discovered device ASN. This value overrides the domain ASN for BGP configuration.
EigrpAsn	U32	0	RW	ASN for EIGRP. When specified, this ASN is used to create an EIGRP routing process which will contain the individual vrf-address-family configuration required for each VPN running EIGRP.
LoopBackId	U32	0	RW	The Loopback ID value is used to create a loopback interface name by appending it to the name 'loopback'. For example, if the Loopback ID is 0, the loopback interface name created is 'loopback0'. When a device in this domain is discovered, a check is made to see if a loopback interface matching this text string exists. If it does, the IP address of the loopback interface is stored with the device information. Range: 0–4 294 967 295 Note: Any changes made to the default loopback ID for the domain, or for any devices, have no effect until the affected devices are re-discovered. Note that on Juniper M-series devices, the loopback ID must always be specified as 0 through the user interface.

Attribute Name	Type	Default	Access	Explanation
OverrideLoopbackId	Boolean	False	RW	Set this value to override the default loopback ID specified for the domain for this device.
DeviceType	String	""	RW	Name of DeviceType object containing details of this type of device. The DeviceType object must be a child of the Topology object. This attribute can be used to link directly to other objects in the EOM.
MaxTransactionSize	U32	0	RW	Maximum number of matches of Configuration Threshold regex pattern allowed in a device configuration session.
IgnoresTransactionSize	Boolean	False	RW	True = Configuration Thresholding is turned on. False = Configuration Thresholding is turned off.
InheritsTransactionSize	Boolean	False	RW	True = all Configuration Thresholding parameters (not just the maximum transaction size) are inherited from parent network. False = parameters are not inherited.
MatchesPatternTransactionSize	String	""	RW	Configuration Threshold regex pattern (limit 127 characters).
StrictClassAggregation	Boolean	False	RW	Enables strict aggregation processing for classifications applied to the current device, which ensures classifications from a contained group are promoted to the parent group, when aggregation is enabled on the contained classification group.

* Encrypted string

Object inheritance

Device.Object

Modifying the regular expression for MatchesPatternTransactionSize

Refer to the topic *Configuration Thresholding feature - modifying the regular expression* on page 378.

DeviceCapabilities

Description

The DeviceCapabilities object represents the capabilities and characteristics of a device. One object exists for each device.

Attributes

Attribute Name	Type	Default	Access	Explanation
AhMacSecurityAlgorithmsCaps	U32	0	RW	Indicates support for AhMacSecurityAlgorithmsCaps.
CapabilitiesSet	Boolean	False	RW	Indicates support for capabilities sets.
CASupport	Boolean	False	RW	Indicates support for CA.
CompressionAlgorithmCaps	U32	0	RW	Indicates support for CompressionAlgorithmCaps.
DhGroupCaps	U32	0	RW	Indicates support for DhGroupCaps.
EspSecurityAlgorithmsCaps	U32	0	RW	Indicates support for EspSecurityAlgorithmsCaps.
GreSupport	Boolean	False	RW	Indicates support for Gre.

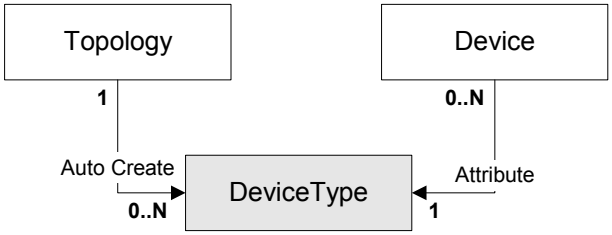
Attribute Name	Type	Default	Access	Explanation
IkeSecurityAlgorithmsCaps	U32	0	RW	Indicates support for IkeSecurityAlgorithmsCaps.
IPsecModesCaps	U32	0	RW	Indicates support for IPsecModesCaps.
SAANetflowVersionSupport	U32	0	RW	Indicates support for SAANetflowVersion.
SAASupport	Boolean	False	RW	Indicates support for SAA.
SAATypesSupported	U32	0	RW	Indicates support for SAATypes.
SharedKeySupport	Boolean	False	RW	Indicates support for SharedKey.
TlsSupport	Boolean	False	RW	Indicates support for Tls.
TrustedRootCAsSupport	Boolean	False	RW	Indicates support for TrustedRootCAs.
VlansSupport	Boolean	False	RW	Indicates support for Vlans.
Name	String		RO	Name of the object.

DeviceType object

Description

A DeviceType object represents a device type recognized by Service Activator. They are automatically created on startup, and on discovery of a device of a new type.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RW	Name of device type.
DeviceDriver	String	""	RW	Name of the device driver used to manage this device type.
Vendor	String	""	RW	Company that makes the device (such as Cisco).
Product	String	""	RW	Name of the device (such as 2500).
SoftwareVersion	String	""	RW	Version of software in device (such as 12.1).
SoftwareOs	String	""	RW	Name of operating system (such as IOS).
SysObjectId	String	""	RW	SNMP-obtained unique object identifier.
StrictClassAggregation	Boolean	False	RW	Enables strict classification aggregation on this device

Object inheritance

DeviceType.Object

InterfaceCapabilities object

Description

An InterfaceCapabilities object represents the capabilities and characteristics of an interface on the device. One object exists for each interface. These objects are created by reading the interface table of each device. Capabilities objects are re-used for objects which have the same set of capabilities parameters.

Warning: Do not attempt to modify values in the Capabilities object through the OIM. Do not attempt to create Capabilities objects through the OIM. These actions will cause system instability.

Attributes

Attribute Name	Type	Default	Access	Explanation
AccessRulesSupported	U32	0	RCW	Indicates that you can implement access rules on this interface.
ATMQoSSupport	U32	0	RCW	Indicates that you can implement PHB groups using ATM Traffic Shaping on this PVC.
ATMQueueDepthSupport	Boolean	False	RW	Indicates that you can implement PHB groups using ATM queue depth.
ATMTxRingLimitSupport	Boolean	False	RW	Indicates that you can implement PHB groups using ATMTxRingLimit.
BECNAdaptSupport	Boolean	False	RCW	Indicates that you can implement PHB groups using BECNAdapt.

Attribute Name	Type	Default	Access	Explanation
CapabilitiesSet	Boolean	False	RCW	Indicates support for capabilities sets.
CbqClassify	U32	0	RCW	Indicates that you can implement classification rules on this interface using Cbq.
CbqClassifyEnh	U32	0	RW	Indicates that you can implement enhanced classification rules on this interface using Cbq.
CbqSupport	U32	0	RCW	Indicates that you can implement PHB groups using Cbq.
ClassificationAccessRules	U32	0	RCW	Indicates that you can implement classification rules on this interface.
ClassificationEnhAccessRules	U32	0	RW	Indicates that you can implement enhanced classification rules on this interface.
ClassificationEnhPolicingRules	U32	0	RW	Indicates that you can implement classification of enhanced policing rules on this interface.

Attribute Name	Type	Default	Access	Explanation
ClassificationEnhServiceRules	U32	0	RW	Indicates that you can implement classification of enhanced service rules on this interface.
ClassificationPolicingRules	U32	0	RCW	Indicates that you can implement classification of policing rules on this interface.
ClassificationServiceRules	U32	0	RCW	Indicates that you can implement classification of service rules on this interface.
DeMarking	Boolean	False	RCW	Indicates support for DeMarking.
FECNAdaptSupport	Boolean	False	RCW	Indicates that you can implement PHB groups using FECNAdapt.
FRF12Support	Boolean	False	RCW	Indicates that you can implement PHB groups using FRF12.
FRQueueDepthSupport	Boolean	False	RW	Indicates that you can implement PHB groups using FRQueueDepth.
FrtsInboundSupport	Boolean	False	RW	Indicates the direction (inbound) in which you can implement PHB groups using FRTS.

Attribute Name	Type	Default	Access	Explanation
FrtsSupport	Boolean	False	RCW	Indicates that you can implement PHB groups using FRTS.
GuaranteesSupported	U32	0	RCW	Indicates that you can implement PHB groups using Guarantees.
IpUnnumberedConfigSupport	Boolean	False	RW	Indicates that you can implement PHB groups using IpUnnumberedConfig.
LabelSwitchingSupport	Boolean	False	RCW	Indicates that you can implement PHB groups using LabelSwitching.
LimitsSupported	U32	0	RCW	Indicates that you can implement PHB groups using Limits.
MarkingPolicingRules	U32	0	RCW	Indicates that you can implement marking policing rules on this interface.
MarkingServiceRules	U32	0	RCW	Indicates that you can implement marking service rules on this interface.
MarkingSupported	U32	0	RCW	Indicates that you can implement PHB groups using Marking.
MqcAggregatePolicer	Boolean	False	RW	Indicates support for MqcAggregatePolicer.

Attribute Name	Type	Default	Access	Explanation
MqcClassify	U32	0	RCW	Indicates that you can implement classification rules on this interface using Mqc.
MqcClassifyEnh	U32	0	RW	Indicates that you can implement ehanced classification rules on this interface using Mqc.
MqcLlqBandwidthTypeSupport	U32	0	RCW	Indicates that you can implement PHB groups using MqcLlqBandwidthType.
MqcLlqBurstRateSupport	Boolean	False	RW	Indicates that you can implement PHB groups using MqcLlqBurstRate.
MqcLlqSupport	Boolean	False	RCW	Indicates that this interface supports MqcLlq.
MqcMarking	U32	0	RCW	Indicates support for MqcMarking.
MqcMaxReservedBandwidth	Boolean	False	RW	Indicates support for MqcMax reserved bandwidth.
MqcNestingSupport	Boolean	False	RCW	Indicates that this interface supports MqcNesting.
MqcPoliceRateTypeSupport	U32	1	RCW	Indicates that this interface supports MqcPoliceRateType.

Attribute Name	Type	Default	Access	Explanation
MqcQueueLimitSupport	Boolean	False	RCW	Indicates that this interface supports MqcQueueLimit.
MqcShapeFrtsSupport	Boolean	False	RCW	Indicates that this interface supports MqcShapeFrts.
MqcShapeSupport	Boolean	False	RCW	Indicates that this interface supports MqcShape.
MqcShapingBuffersSupport	Boolean	False	RW	Indicates that this interface supports MqcShapingBuffers.
MqcSingleRatePoliceAction	U32	0	RCW	Indicates support for MqcSingleRatePoliceAction capability.
MqcSingleRatePoliceSupport	Boolean	False	RCW	Indicates that this interface supports MqcSingleRatePolice.
MqcSupport	U32	0	RCW	Indicates that this interface supports Mqc.
MqcTwoRatePoliceAction	U32	0	RCW	Indicates support for MqcTwoRatePoliceAction capability.
MqcTwoRatePoliceSupport	Boolean	False	RCW	Indicates that this interface supports MqcTwoRatePolice.
MqcWfqBandwidthTypeSupport	U32	0	RCW	Indicates that this interface supports MqcWfqBandwidthType.

Attribute Name	Type	Default	Access	Explanation
MqcWfqSupport	Boolean	False	RCW	Indicates that this interface supports MqcWfq.
MqcWredClassify	U32	0	RCW	Indicates that you can implement classification rules on this interface using MqcWred.
MqcWredClassifyEnh	U32	0	RW	Indicates that you can implement enhanced classification rules on this interface using Mqc.
MqcWredEcn	Boolean	False	RW	Indicates support for MqcWredEcn.
MqcWredSupport	U32	0	RCW	Indicates that this interface supports MqcWred.
Outbound	Boolean	TRUE	RCW	Indicates the direction of the service.
PolicingSupported	U32	0	RCW	Indicates that you can implement policing rules.
PqClassify	U32	0	RCW	Indicates that you can implement classification rules on this interface using Pq.

Attribute Name	Type	Default	Access	Explanation
PqClassifyEnh	U32	0	RW	Indicates that you can implement enhanced classification rules on this interface using Pq.
PqSupport	U32	0	RCW	Indicates that this interface supports Pq.
PtToPtEncapsulationCaps	U32	0	RCW	Indicates support for Point to Point EncapsulationCaps capability.
RlimBurst	Boolean	False	RCW	Indicates support for RlimBurst capability.
RlimClassify	U32	0	RCW	Indicates that you can implement classification rules on this interface using Rlim.
RlimClassifyEnh	U32	0	RW	Indicates that you can implement enhanced classification rules on this interface using Rlim.

Attribute Name	Type	Default	Access	Explanation
RoutingCaps	bitmap	0	RCW	0 - None 1 - RIP for VPN 2 - Static for VPN 4 - EBGP for VPN 8 - OSPF for VPN 16 - RIP for Virtual CE (not yet supported) 32 - Static for Virtual CE 64 - EBGP for Virtual CE 128 - OSPF for Virtual CE (not yet supported)
VCConfigSupport	Boolean	False	RCW	Indicates that VCConfig is supported.
VpnSupport	bitmap	0	RCW	Indicates that this interface supports VPNs. 0 - None 1 - MPLS 2 - IPsec 4 - Virtual CE
WfqClassify	U32	0	RCW	Indicates that you can implement PHB groups using the WFQ.
WfqClassifyEnh	U32	0	RW	Indicates that you can implement PHB groups using the enhanced WFQ.

Attribute Name	Type	Default	Access	Explanation
WfqHighPriorityWeightAsPercent	Boolean	False	RCW	Indicates that you can implement PHB groups using the WFQ priority as percentage.
WfqLowPriorityWeightAsPercent	Boolean	False	RCW	Indicates that you can implement PHB groups using the WFQ low priority weight as percentage.
WfqPlusWredClassify	U32	0	RCW	Indicates that you can implement PHB groups using the WFQ plus Wred.
WfqPlusWredClassifyEnh	U32	0	RW	Indicates that you can implement PHB groups using the enhanced WFQ plus Wred.
WfqPlusWredSupport	Boolean	False	RCW	Indicates that you can implement PHB groups using the WFQ plus Wred.
WfqSupport	U32	0	RCW	Indicates that you can implement PHB groups using the WFQ.
WredClassify	U32	0	RCW	Indicates that you can implement classification rules on this interface using Wred.

Attribute Name	Type	Default	Access	Explanation
WredClassifyEnh	U32	0	RW	Indicates that you can implement classification rules on this interface using enhanced Wred.
WredEcn	Boolean	False	RW	Indicates support for WredEcn.
WredSupport	U32	0	RCW	Indicates that you can implement PHB groups using the WRED.
WrrClassify	U32	0	RCW	Indicates that you can implement classification rules on this interface using Wrr.
WrrClassifyEnh	U32	0	RW	Indicates that you can implement classification rules on this interface using enhanced Wrr.
WrrSupport	U32	0	RCW	Indicates that you can implement PHB groups using the WRR.
Name	String	"InterfaceCapabilities"	R	Name of the object.
Id	U32	490	R	The unique ID used to reference thsi object.

Object inheritnce

SubInterfaceCapabilities.Object

VCCapabilities Object

Description

A VCCapabilities object represents the capabilities and characteristics of a virtual circuit on the device. One object exists for each VC. Capabilities objects are re-used for objects which have the same set of capabilities parameters.

Object Diagram

See *Object diagram* on page 243.

Attributes

Attribute Name	Type	Default	Access	Explanation
CbqClassifyEnh	U32	0	RW	Indicates that you can implement enhanced classification rules on this interface using Cbq.
ClassificationEnhPolicingRules	U32	0	RW	Indicates that you can implement classification using enhanced policing rules.
ClassificationEnhServiceRules	U32	0	RW	Indicates that you can implement classification using enhanced service rules.
MqcClassifyEnh	U32	0	RW	Indicates that you can implement enhanced classification rules on this interface using Mqc.
MqcWredClassifyEnh	U32	0	RW	Indicates that you can implement enhanced classification rules on this interface using Mcq and Wred.
PqClassifyEnh	U32	0	RW	Indicates that you can implement enhanced classification rules on this interface using Pq.

Attribute Name	Type	Default	Access	Explanation
RlimClassifyEnh	U32	0	RW	Indicates that you can implement enhanced classification rules on this interface using Rlim.
WfqClassifyEnh	U32	0	RW	Indicates that you can implement enhanced classification rules on this interface using Wfq.
WfqPlusWredClassifyEnh	U32	0	RW	Indicates that you can implement enhanced classification rules on this interface using Wfq and Wred.
WredClassifyEnh	U32	0	RW	Indicates that you can implement enhanced classification rules on this interface using Wred.
WrrClassifyEnh	U32	0	RW	Indicates that you can implement enhanced classification rules on this interface using Wrr.

Object inheritance

VCCapabilities.InterfaceCapabilities.Object

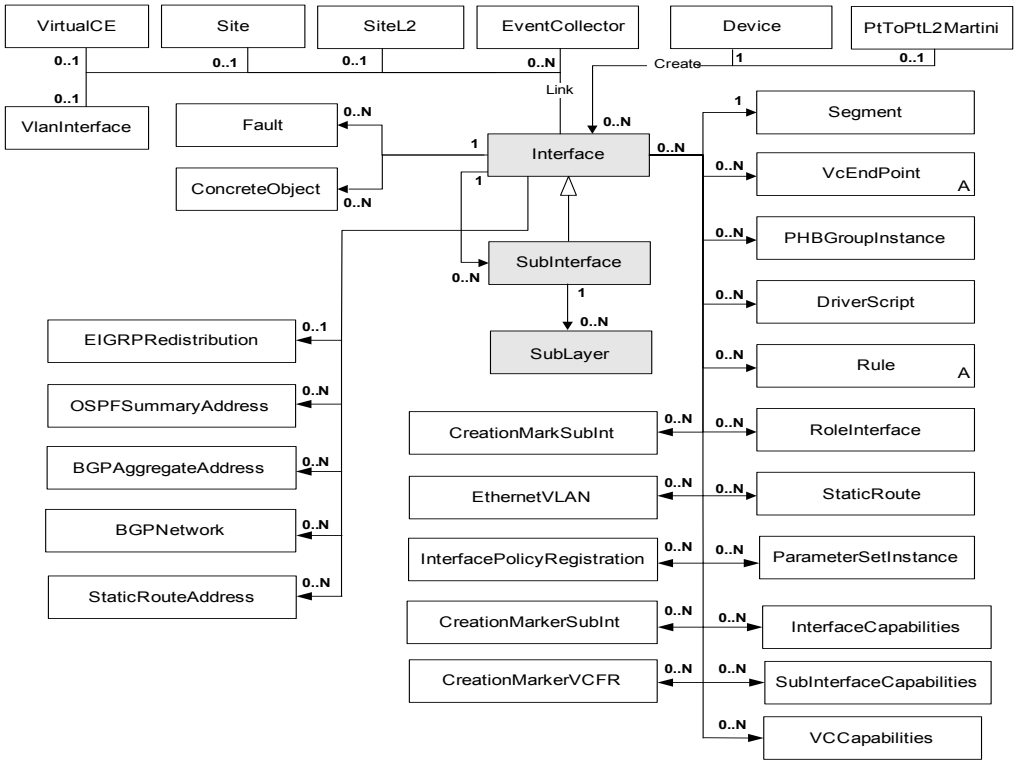
Interface object

Description

An Interface object represents an interface on the device. One object exists for each interface. These objects are created by reading the interface table of each device.

Some of the attributes on an Interface object are only meaningful when the Interface is linked as a child of a site object. The attributes cannot be modified when not linked to a site, and if the interface is unlinked from the site then the attributes will automatically be set to null values. The affected attributes are marked 'Site Only' in the table below.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RCW	Name of interface.
Description	String	""	RCW	SNMP ifDescription.
Number	U32	0	RCW	Number of interface (SNMP ifIndex).
Type	U32	1	RCW	SNMP ifType.
IpAddr	IPAddress	0.0.0.0	RCW	IP Address for interface if applicable.
SubnetMask	IPAddress	0.0.0.0	RCW	Subnet mask.
PhysicalAddress	String	""	RCW	Physical address of the interface (the MAC address).
Speed	U32	0	RCW	Speed of the interface in Kbits/s (SNMP ifSpeed parameter).
State	Enum	4	RCW	State of interface 0 = Down 1 = Up 2 = Testing 3 = Shutdown 4 = Unknown Constructed from the SNMP ifAdminStatus and ifOperStatus. Interfaces and sub-interfaces connected to a virtual device always have a state of "Unknown".
PublicPeIpAddr (Site only)	IPAddress	0.0.0.0	RO	PE interface address when not in a VRF.

Attribute Name	Type	Default	Access	Explanation
PublicPeMask (Site Only)	IPAddress	0.0.0.0	RO	PE interface address when not in a VRF.
PrivatePeIpAddr (Site Only)	IPAddress	0.0.0.0	RO	PE interface address when in a VRF.
PrivatePeMask (Site only)	IPAddress	0.0.0.0	RO	PE interface address when in a VRF.
PrivateCeIpAddr (Site only)	IPAddress	0.0.0.0	RO	CE interface address when using EBGP and in a VRF.
PrivateInterface Description	String	""	RO	Description of the interface.
VrfTableName (Site only)	String	""	RO	The name of the VRF routing table.
OverrideVrfTable Limit (VPN site only)	Boolean	False	RO	True = Use site-specific settings for VRF table limits False = Use domain defaults for VRF table limits.
VrfTableLimit (VPN site only)	U32	0	RO	Maximum number of routes allowed in a VRF (0=No limit).
VrfTableLimit Warning (VPN site only)	U32	0	RO	Percentage at which to warn of VRF table limits being exceeded. Range: 1–101, where 1–100 = percentage of VrfTableLimit reached warning. 101 = warning when VrfTableLimit reached

Attribute Name	Type	Default	Access	Explanation
RDHighOrder (VPN site only)	U32	0	RO	The top 32 bits of the Route Descriptor value.
RDLowOrder (VPN site only)	U32	0	RO	The bottom 32 bits of the route Descriptor value.
ActualRDHighOrder	U32	0	RO	The actual high-order RD number.
ActualRDLowOrder	U32	0	RO	The actual low-order RD number.
LocalPreference	U32	0xffffffff	RO	0xffffffff as default value in Cisco device this will equate to (100) when Local-preference command is not used. Can be in range 0 to $*2^{32}$ - 2) for EBGP only. A lower value indicates a high priority.
VrfExportFilter (Site only)	String	""	RO	Name of the pre-existing VRF export filter. "" = no filter applied.
EbgpPrefixFilterIn (Site only)	String	""	RO	Name of the pre-existing inbound prefix filter. "" = no filter applied.
EbgpPrefixFilterOut (Site only)	String	""	RO	Name of the pre-existing outbound prefix filter. "" = no filter applied.
EbgpPrefixLimit (Site only)	U32	0	RO	The maximum number of eBGP prefixes that a PE is allowed to receive. Range: $1–2^{32}$ 0 = no limit

Attribute Name	Type	Default	Access	Explanation
EbgpPrefixLimitWarning (Site only)	U32	0	RO	The percentage at which to warn of eBGP prefix limits being exceeded. Range: 1–101, where 1–100 = warning when percentage of EbgpPrefix limit reached. 101 = warning when EbgpPrefixLimit reached.
EbgpPrefixRestartDelay (Site only)	U32	0	RO	Delay, in minutes, before automatic restart of the EBGP session, after prefix limit was reached and session terminated. Range: 0–65535 0 = disabled
KeepAlive	U32	60	RO	Frequency, in seconds with which Keep Alive messages are sent to neighbors or a specific group of neighbors. Range: 0–65535 0 = disabled
HoldTime	U32	180	RO	Delay, in seconds, if a device does not receive a Keep Alive message till this time, it will declare its neighbor to be dead. Range: 0–65535 0 = disabled
EnableAdvertisementInterval	Boolean	False	RO	Allows the exchange of eBGP routing updates between two neighbors.

Attribute Name	Type	Default	Access	Explanation
AdvertisementInterval	U32	0	RO	Delay, in seconds, after which the subsequent eBGP routing updates to be exchanged between two neighbors. Range: 0–600
UpdateSourceInterface	String	PE Interface ID	RCW	Allows the definition of source interface to be used for eBGP updates on per neighbor basis.
NeighbourDescription	String	Site Name	RCW	User defined description of neighbors on a per neighbor basis. Maximum 80 characters allowed.
AsOverride (Site only)	Boolean	False	RO	True = Set AS Override for EBGP neighbors. False = Do not set AS override.
OverrideAllowAsIn (Site only)	Boolean	False	RO	True = Site setting overrides the domain default setting of Allow AS in. False = Site setting should not override the domain default setting of Allow AS in.
AllowAsIn (Site only)	U32	0	RO	The number of times the same AS can appear in the AS path list. Range: 0–10
BgpMd5Key (Site only)	Encrypted string	""	RO	BGP MD5 authentication key ""=not used.

Attribute Name	Type	Default	Access	Explanation
EbgpMd5InheritFromVPN	Boolean	False	RO	True = inherit the interface's BGP authentication settings from the parent VPN. False = Don't.
OverridePeCeSendCommunity (Site only)	Boolean	False	RO	True = use local settings for Send Community parameters. False = use domain-level settings.
PeCeSendStandardCommunity (Site only)	Boolean	False	RO	True = Use the PE-CE peering Standard send community tag. False = Don't.
PeCeSendExtendedCommunity (Site only)	Boolean	False	RO	True = Use the PE-CE peering Extended send community tag. False = Don't
Context (Site only)	String	""	RCW	Local context for driver scripts applied to interfaces (max 512 bytes).
VrfDesc (VPN site only)	String	""	RO	Optional string describes the VRF applied to router, if the router supports VRF description string.
EIBgpMaxPaths (VPN site only)	U32	1	RO	Maximum number of multipaths on both EBGP and IBGP. Range: 1–6
IBgpUnequalCost	Boolean	False	RO	Allows load balancing by selecting iBGP paths that do not have equal cost.

Attribute Name	Type	Default	Access	Explanation
VrfImport (VPN site only)	U32	1 (turned off)	RO	The number of device redundant path configurations. Range is 2^{16} plus default value.
UseVrfImport	Boolean	True	RO	True = Use VRF import False = Do not use VRF import
InternalName	String		RCW	
IsConfigurable	Boolean		RO	
IsController	Boolean	False	RC	Whether this interface has been tagged as controller during discovery
AutoInheritVrf	Boolean	False	RO	Try to inherit the VRF table parameters from the VRF template in the VPN this interface is connected to.
EBgpMaxPaths	U32	0	RO	Specification of the maximum number of parallel EBGP routes that can be installed in the routing table. This corresponds to the Cisco maximum-paths command. Range: 1–16
EbgpLocalAsn	U32	0	RO	Autonomous Systems path. Range: 1–65535

Attribute Name	Type	Default	Access	Explanation
EbgpLocalAsnEnable	Boolean	False	RO	Whether Local Autonomous Systems Numbers is enabled or not.
EbgpLocalAsnNoPrepend	Boolean	False	RO	If true, device does not prepend the local ASN to any routes received from the eBGP neighbor
EbgpMd5InheritFromVPN	Boolean	False	RO	True = inherit the interface's EBGP authentication settings from the parent VPN. False = Don't.
EbgpMultihop	Boolean	False	RO	Enables BGP connections to devices on networks that are not directly connected (that are more than one hop away from a local device).
EbgpSoftReconfig	Boolean	False	RO	Enables the EBGP soft reconfiguration setup command on Cisco and Juniper E-series devices. Setting this value does not issue a soft reconfiguration reset action - it enables the support for the reset action
EigrpDampHalfLife	U32	False	RO	Time, in minutes, when a penalty applying to a route is decreased by half. Range: 1–45, default is 15.

Attribute Name	Type	Default	Access	Explanation
EigrpDampMaxSuppressTime	U32	0	RO	Maximum time, in minutes, that a route can be suppressed. The valid range is 1–255 for Cisco and Juniper E-series devices, 1–720 for Juniper M-series devices. The default is 60 minutes.
EigrpDampReuse	U32	0	RO	Reuse threshold (default 750) Range: 1–20 000
EigrpDampSuppress	U32	0	RO	Cutoff (suppression) threshold (default 2000) Range: 1–20 000
EigrpMaxPaths	U32	0	RO	Specification of the maximum number of parallel EIGRP routes that can be installed in the routing table. This corresponds to the Cisco maximum-paths command. Range: 1–16
EigrpMd5Enable	Boolean	0	RO	Enable MD5 key authentication for EIGRP for the interface
EigrpMd5InheritFromVPN	Boolean	0	RO	True = inherit the interface's EIGRP authentication settings from the parent VPN. False = Don’t.
EigrpMd5KeyChainRef	String		RO	Key chain name to use with MD5 Authentication for EIGRP.

Attribute Name	Type	Default	Access	Explanation
EnableAdvertisementInterval	Boolean	False	RO	EBGP: Allows the exchange of eBGP routing updates between two neighbors.
EnableExternalInboundRouteMap	Boolean	False	RO	Enables the use of a specified route map name for inbound external route-map. Disabled for virtual-CE.
UseExternalOutboundRouteMap	Boolean	False	RO	Enables specification of an outbound external BGP route-map for the interface
ExternalInboundRouteMap	String		RO	Name of the inbound external route-map
ExternalOutboundRouteMap	String		RO	Name of the outbound external route-map
InboundRouteMap	String		RO	Inbound route map name.
InheritVRFRouteMapsFromVPN	Boolean	False	RO	Inherit the VRF table name and RD defined for the parent VPN
LoopbackIpAddr	IPAddress	0.0.0.0	RO	

Attribute Name	Type	Default	Access	Explanation
MultiVpnOverride	Boolean	False	RO	If a site is set to inherit VPN-wide VRF/RD details and participates in more than one VPN for which VPN-wide details are defined, Service Activator handles the conflict by applying site-specific automatically generated VRF/RD details to the site. If MultiVpnOverride and InheritVRFRouteMapsFromVPN are both set to true: - if the site is a member of only one VPN, the VRF table name and RD are derived from the parent VPN - if the site is a member of multiple VPNs, the VRF table name and RD are derived using the site specific options
NoRedistributeStaticMerge	Boolean	False	RW	Use Default Static Route Redistribution

Attribute Name	Type	Default	Access	Explanation
NoRedistributeBgp2Eigrp	Boolean	FALSE	RO	Used to turn off route redistribution from Bgp to Eigrp; when this attribute is set to true, routes will not be redistributed from Bgp to Eigrp. For this attribute access rights will change once this interface is linked to the site and connectivity is set as BGP.
NoRedistributeBgp2Ospf	Boolean	FALSE	RO	Used to turn off route redistribution from Bgp to Ospf; when this attribute is set to true, routes will not be redistributed from Bgp to Ospf. By default access will be "RO", when the interface is linked to some site and if the connectiviy changed to BGP then this attribute access changes to "RCW".

Attribute Name	Type	Default	Access	Explanation
NoRedistributeEigrp2Bgp	Boolean	FALSE	RO	Used to turn off route redistribution from Eigrp to Bgp; when this attribute is set to true, routes will not be redistributed from Eigrp to Bgp. Access is R only by default. For this attribute access rights will change once this interface is linked to the site and connectivity is EIGRP.
NoRedistributeEigrp2Ospf	Boolean	FALSE	RO	Used to turn off route redistribution from Eigrp to Ospf; when this attribute is set to true, routes will not be redistributed from Eigrp to Ospf. Access is RO by default. For this attribute access rights will change once this interface is linked to the site and connectivity is EIGRP.

Attribute Name	Type	Default	Access	Explanation
NoRedistributeEigrp2Rip	Boolean	FALSE	RO	Used to turn off route redistribution from Eigrp to Rip; when this attribute is set to true, routes will not be redistributed from Eigrp to Rip. Access for attribute NoRedistributeEigrp2Rip is RO only by default. For this attribute access rights will change once this interface is linked to the site and connectivity is EIGRP.
NoRedistributeOspf2Bgp	Boolean	FALSE	RO	Used to turn off route redistribution from Ospf to Bgp; when this attribute is set to true, routes will not be redistributed from Ospf to Bgp. Access type is not 'RW' by default for NoRedistributeOspf2Bgp attribute (i.e. Access is R only by default). For this attribute access rights will change once this interface is linked to the site and connectivity is OSPF.

Attribute Name	Type	Default	Access	Explanation
NoRedistributeOspf2Eigrp	Boolean	FALSE	RO	Used to turn off route redistribution from Ospf to Eigrp; when this attribute is set to true, routes will not be redistributed from Ospf to Eigrp. Access type is not 'RW' by default for NoRedistributeOspf2Eigrp attribute (i.e. access is R only by default). For this attribute access rights will change once this interface is linked to the site and connectivity is OSPF.
NoRedistributeOspf2Rip	Boolean	FALSE	RO	Used to turn off route redistribution from Ospf to Rip; when this attribute is set to true, routes will not be redistributed from Ospf to Rip. Access type is not 'RW' by default for NoRedistributeOspf2Rip attribute (i.e. access is R only by default). For this attribute access rights will change once this interface is linked to the site and connectivity is OSPF.

Attribute Name	Type	Default	Access	Explanation
NoRedistributeRip2Eigrp	Boolean	FALSE	RO	Used to turn off route redistribution from Rip to Eigrp; when this attribute is set to true, routes will not be redistributed from Rip to Eigrp. Access type is not 'RW' by default for NoRedistributeRip2Eigrp attribute (i.e. access is R only by default). For this attribute access rights will change once this interface is linked to the site and connectivity is RIP.
NoRedistributeRip2Ospf	Boolean	FALSE	RO	Used to turn off route redistribution from Rip to Ospf; when this attribute is set to true, routes will not be redistributed from Rip to Ospf. Access type is not 'RW' by default for NoRedistributeRip2Ospf attribute (i.e. access is R only by default). For this attribute access rights will change once this interface is linked to the site and connectivity is RIP.

Attribute Name	Type	Default	Access	Explanation
NoRedistributeStatic2Bgp	Boolean	FALSE	RO	Used to turn off route redistribution from Static to Bgp; when this attribute is set to true, routes will not be redistributed from Static to Bgp. Access type is not 'RW' by default for NoRedistributeStatic2Bgp attribute (i.e. access is R only by default). For this attribute access rights will change once this interface is linked to the site and connectivity is RIP/OSPF/EIGRP.
NoRedistributeStatic2Eigrp	Boolean	FALSE	RO	Used to turn off route redistribution from Static to Eigrp; when this attribute is set to true, routes will not be redistributed from Static to Eigrp. Access type is not 'RW' by default for NoRedistributeStatic2Eigrp attribute (i.e. access is R only by default). For this attribute access rights will change once this interface is linked to the site and connectivity is RIP/OSPF/EIGRP.

Attribute Name	Type	Default	Access	Explanation
NoRedistributeStatic2Ospf	Boolean	FALSE	RO	Used to turn off route redistribution from Static to Ospf; when this attribute is set to true, routes will not be redistributed from Static to Ospf. Access type is not 'RW' by default for NoRedistributeStatic2Ospf attribute (i.e. access is R only by default). For this attribute access rights will change once this interface is linked to the site and connectivity is RIP/OSPF/EIGRP.
NoRedistributeStatic2Rip	Boolean	FALSE	RO	Used to turn off route redistribution from Static to Rip; when this attribute is set to true, routes will not be redistributed from Static to Rip. Access type is not 'RW' by default for NoRedistributeStatic2Rip attribute (i.e. access is R only by default). For this attribute access rights will change once this interface is linked to the site and connectivity is RIP/OSPF/EIGRP.
OspfArea	IPAddressOrInt	0	RO	OSPF area ID for the selected interface

Attribute Name	Type	Default	Access	Explanation
OspfAreaIsIpAddress	Boolean	False	RO	Whether OspfArea is specified as an IP Address or as an integer
OspfCost	U32	0	RO	Cost of sending a packet on the selected interface. Range: 1–65535
OspfDistributeOutAcl	String		RO	Named ACL for Distribute Out filtering. The access list specified is applied to outgoing updates on the selected interface and suppresses networks from being advertised in updates.
OspfMaxPaths	U32	0	RO	Maximum redundant routes OSPF can use. Range: 1–6
OspfMd5AuthLocally	Boolean	False	RO	Enable MD5 key authentication for OSPF locally on the interface.
OspfMd5InheritFromVPN	Boolean	False	RO	Inherit the interface's OSPF authentication settings from the parent VPN.
OspfNSSANoRedistribution	Boolean	False	RO	Suppression of the NSSA behavior in which Type 7 LSAs are translated to Type 5 LSAs.
OspfRedistTagValueFromBgp	U32	0	RO	Tag value to identify routes redistributed into OSPF from BGP.

Attribute Name	Type	Default	Access	Explanation
OspfSpfHold	U32	0	RO	Minimum time in milliseconds between consecutive SPF recalculations.
OspfSpfMaxWait	U32	0	RO	Maximum wait time in milliseconds between consecutive SPF recalculations.
OspfSpfStart	U32		RO	Minimum delay in milliseconds between the reception of a topology change and the start of SPF recalculation.
OspfUseRedistTagFromBgp	Boolean	False	RO	Enable the use of the tag value to identify routes redistributed into OSPF from BGP.
OspfUseSpfThrottling	Boolean	False	RO	Enable control of timing and execution of SPF recalculations.
OverridePeCeSendCommunity	Boolean	False	RO	Override PE-CE community bit sharing
OverrideVrfTableLimit	Boolean	False	RO	Override the domain vrf table limit
PeCeSendExtendedCommunity	Boolean	False	RO	Share extended community between PE and CE devices on a VPN
PeCeSendStandardCommunity	Boolean	False	RO	Standard community between PE and CE devices on a VPN
PolicyServiceCustomer	String		RO	

Attribute Name	Type	Default	Access	Explanation
PrivateInterfaceDescription	String		RO	Description of the interface; such as the name of the customer that is associated with the interface.
PrivatePeIpAddrINA	Boolean	False	RO	Whether private PE IP address is allocated by INA.
PrivatePeIpUnnumberedRef	String		RO	Interface name to use for IP unnumbered Private PE addressing. This allows to enable IP on an interface and use it in a VPN without having to assign an explicit Private PE IP address and mask. Instead, the IP address of loopback address from the device is used.
Note: Use a naming scheme different from Service Activator's for external inbound and outbound route-maps. Service Activator will remove route-maps with the same naming as those which it generates when the device is unmanaged and re-managed.				
UseExternalInboundRouteMap (Site only)	Boolean	False	RO	True = Use the specified external inbound route-map. False = Don't
ExternalInboundRouteMap (Site only)	String	""	RO	Value for the external inbound route-map.
UseExternalOutboundRouteMap (Site only)	Boolean	False	RO	True = Use the specified external outbound route-map. False = Don't

Attribute Name	Type	Default	Access	Explanation
ExternalOutboundRouteMap (Site only)	String	""	RO	Value for the external outbound route-map.
EBgpDampHalfLife (VPN site only)	U32	0	RO	Time, in minutes, at which a penalty applying to a route is decreased by half. Range: 1–45 0 = No EBGP dampening. 15 = Default if EBGP dampening applied.
EBgpDampMaxSuppressTime (VPN site only)	U32	60	RO	Max. time, in minutes, that a route can be suppressed. in minutes. Range: 1–720
EBgpDampSuppress (VPN site only)	U32	2000	RO	A route is suppressed when its penalty exceeds this limit. Range: 1 –20 000. Must be greater than or equal to EBgpDampReuse.
EBgpDampReuse (VPN site only)	U32	750	RO	When the penalty applying to a route falls below this value, the route is unsuppressed. Range: 1–20 000
RedistributeDefaultRoute (Site only)	Boolean	False	RO	True = Redistribute the Default route. False = Don't.
RedistributeConnected (Site only)	Boolean	False	RO	True = Redistribute the connected routes. False = Don't.

Attribute Name	Type	Default	Access	Explanation
UseDefaultRedistribution (Site only)	Boolean	False	RO	True = use default redistribution metrics and policies and import RIP metric from VPN. False = Don't.
ForceVrfInstall (Site only)	Boolean	False	RO	True = VRF tables on corresponding interfaces must be installed and cannot be merged into other tables. False= VRF tables can be merged into other tables.
ShareableVrf (Site only)	Boolean	False	RO	True = Other tables can be merged into this VRF table. False = Other tables cannot be merged into this VRF table.
VrfExportFilter (Site only)	String	""	RO	Value for the external VRF import map name
VrfImportFilter (Site only)	String	""	RO	Value for the external VRF import map name ""=not used
RedistMetricConnected2Bgp	U32	0	RO	Values for redistribution metric:distribution into BGP. Range is 0 to 2^{32}-1 (VPN site only)
RedistMetricStatic2Bgp	U32	0	RO	
RedistMetricRip2Bgp	U32	0	RO	
RedistMetricOspf2Bgp	U32	0	RO	
RedistMetricEigrp2Bgp	U32	0	RO	

Attribute Name	Type	Default	Access	Explanation
RedistPolicyConnected2Bgp	String	""	RO	Values for redistribution routemap name (policy): distribution into BGP. (VPN site only)
RedistPolicyStatic2Bgp	String	""	RO	
RedistPolicyRip2Bgp	String	""	RO	
RedistPolicyOspf2Bgp	String	""	RO	
RedistPolicyEigrp2Bgp	String	""	RO	
RedistMetric Connected2Rip	U32	0	RO	Values for redistribution metric: distribution into RIP. Range: 0–16 (VPN site only)
RedistMetricStatic2Rip	U32	0	RO	
RedistMetricOspf2Rip	U32	2	RO	
RedistMetricBgp2Rip	U32	1	RO	
RedistMetricEigrp2Rip	U32	0	RO	
RedistPolicyEigrp2Rip	String	""	RO	Values for redistribution routemap name (policy): distribution into RIP. (VPN site only)
RedistPolicyConnected2Rip	String	""	RO	
RedistPolicyStatic2Rip	String	""	RO	
RedistPolicyOspf2Rip	String	""	RO	
RedistPolicyBgp2Rip	String	""	RO	
RedistMetricConnected2Ospf	U32	1	RO	Values for redistribution metric: distribution into OSPF. Range is 0 to 2^{24}-2 (VPN site only)
RedistMetricStatic2Ospf	U32	0	RO	
RedistMetricRip2Ospf	U32	2	RO	
RedistMetricBgp2Ospf	U32	1	RO	
RedistMetricEigrp2Ospf	U32	20	RO	

Attribute Name	Type	Default	Access	Explanation
RedistMetricTypeConnected2Ospf	String	"2"	RO	Value "1" or "2". (VPN site only)
RedistMetricTypeStatic2Ospf				
RedistMetricTypeRip2Ospf				
RedistMetricTypeBgp2Ospf				
RedistMetricTypeDefault2Ospf				
RedistPolicyConnected2Ospf	String	""	RO	Values for redistribution routemap name (policy): distribution into OSPF. (VPN site only)
RedistPolicyStatic2Ospf				
RedistPolicyRip2Ospf				
RedistPolicyBgp2Ospf				
RedistPolicyEigrp2Ospf				
RedistMetricRip2Eigrp	U32	0	RO	Values for redistribution metric: distribution into EIGRP. Range is 0 to 2^{24}-2 (VPN site only)
RedistMetricBgp2Eigrp				
RedistMetricConnected2Eigrp				
RedistMetricOspf2Eigrp				
RedistMetricStatic2Eigrp				
RedistMetricConnected2Bgp	U32	0	RO	Values for redistribution metric:distribution into BGP. Range is 0 to 2^{32}-1 (VPN site only)

Attribute Name	Type	Default	Access	Explanation
RedistMetricConnected2Eigrp RedistMetricStatic2Eigrp	U32	0	RO	Values for redistribution metric: distribution into EIGRP. Range is 0 to 2^{24}-2 (VPN site only)
RedistMetricConnected2Ospf RedistMetricDefault2Ospf	U32	0	RO	Values for redistribution metric: distribution into OSPF. Range is 0 to 2^{24}-2 (VPN site only)
RedistMetricTypeBgp2Ospf RedistMetricTypeConnected2Ospf RedistMetricTypeDefault2Ospf RedistMetricTypeRip2Ospf RedistMetricTypeStatic2Ospf	String		RO	Value "1" or "2". (VPN site only)
RedistPolicyConnected2Ospf RedistPolicyDefault2Ospf RedistPolicyConnected2Rip RedistPolicyDefault2Rip	String		RO	Values for redistribution routemap name (policy): distribution into OSPF. (VPN site only)

Attribute Name	Type	Default	Access	Explanation
RedistributeBgp2Rip	Boolean	False	RO	Redistribute BGP Routes into RIP
RedistributeRip2Bgp				Redistribute RIP Routes into BGP
RedistributeRip2Bgp				Redistribute Connected routes (Ospf / Rip)
RedistributeDefaultRoute				Redistribute Default route
RedistributeDefaultRouteOspf				Redistribute Default routes (OSPF)
RedistributeDefaultRouteRip				Redistribute Default routes (RIP)
RedistPolicyRip2Eigrp	String	""	RO	Values for redistribution routemap name (policy): distribution into EIGRP. (VPN site only)
RedistPolicyBgp2Eigrp				
RedistPolicyConnected2Eigrp				
RedistPolicyOspf2Eigrp				
RedistPolicyStatic2Eigrp				

Attribute Name	Type	Default	Access	Explanation
RemovePrivateAs	Boolean	False	RO	EBGP: turns on the removal of private autonomous system (AS) numbers from the autonomous system paths advertised by the neighbor WAN Address
UseDefaultRedistribution				When True, the default value will be used in Metrics and Policy fields for redistribution of protocols (EBGP, OSPF, RIP, and EIGRP).
UseVrfLabel				Enable vrf-table-label support on Juniper devices. When this value is set to true, the inner (VPN) label of a packet is removed as it arrives at a VRF so that it can be processed based on the contents of its IP header. When set to false, incoming packets are mapped directly onto an outgoing (CE-facing) interface based on the inner VPN label.
ttl	U32		RO	0 = ignored Range: 1–255 Only applicable if EbgpMultihop is true.

Attribute Name	Type	Default	Access	Explanation
InstallDhcp	Boolean	False	RW	True = Install DHCP support on the VRFs. (When applied at Interface level, overrides per-VPN settings.) False = Do not install DHCP support on VRFs.
PrimaryDhcpIpAddr	IPAddress	0.0.0.0	RW	Primary DHCP Server
SecondaryDhcpIpAddr	IPAddress	0.0.0.0	RW	Secondary DHCP Server
RIPIgnoreRoutes (VPN site only)	Boolean	False	RO	True=RIP routes from the specified IP address and mask are to be ignored. False=Do not ignore routes.
RIPIgnoreRoutesAddress (VPN site only)	IPAddress	0.0.0.0	RO	IP Address from which to ignore routes
RIPIgnoreRoutesMask (VPN site only)	IPAddress	0	RO	Mask of IP Address from which to ignore routes
RIPPassiveInterface (VPN site only)	Boolean	False	RO	True=Interface is configured as a passive (i.e. listen only) interface False=Not a passive interface
OSPFAreaType (VPN site only)	Enum	0	RO	0 = Normal 1 = Nssa 2 = NssaTotallyStub 3 = Stub 4 = StubTotallyStub

Attribute Name	Type	Default	Access	Explanation
OspfMd5Key (VPN site only)	Encrypted string	""	RO	OSPF MD5 authentication key ""=not used.
OspfMd5InheritFromVPN (VPN site only)	Boolean	False	RO	True = inherit the interface's OSPF authentication settings from the parent VPN. False = Don't.
OspfNSSANoRedistribution (VPN site only)	Boolean	False	RO	True = suppresses the NSSA behavior in which Type 7 LSAs are translated to Type 5 LSAs. False = Does not suppress the translation behavior.
BgpAsn (Virtual CE only)	U32	0	RO	BGP Autonomous System Number. Unique number for routing.
RoutingProtocol (Virtual CE only)	Enum	3	RO	Type of routing being used between the PE and CE, relevant to MPLS VPNs only: 0 = EBGP 1 = RIP 3 = None 4 = OSPF
InstallStatic (Virtual CE only)	Boolean	True	RO	True = Static routing is used in conjunction with relevant routing protocol. False = Static routing is not used.

Attribute Name	Type	Default	Access	Explanation
InstallLocalStatic This attribute is now unused.	Boolean	True	RO	True = Static routes defined in the site are not redistributed. False = Static routes defined in the site are redistributed.
InheritRouting	Boolean	False	RO	Always True for interfaces in VPN sites, so that the routing protocol and related attributes are inherited from the site object. Always False for VirtualCEs, so the routing protocol can be specified per interface.
EffectiveCommandDeliveryMode (This value is inherited when the command delivery mode is set at the Device or parent Interface level.)	Enum	0	RO	0 = Online: commands delivered to device 1 = OfflineMaintenance: commands not delivered, concretes and other states updated 2 = OfflineTest: commands not delivered
CommandDeliveryMode (This value is set when the command delivery mode is set at the Interface or SubInterface level.)	Enum	0	RCW	0 = Online: commands delivered to device 1 = OfflineMaintenance: commands not delivered, concretes and other states updated

Object inheritance

Interface.Object

EigrpRedistribution Object

Description

Redistribution attributes (delay, reliability, loading and mtu) from other protocols (connected, static, Bgp, Rip) into Eigrp.

Attributes

Attribute Name	Type	Default	Access	Explanation
RedistDelayFromBgp	U32	4294967295	RCW	Specify Delay in tens of microseconds. Default = 1000 (This is 10 milliseconds)
RedistDelayFromConnected				
RedistDelayFromRip				
RedistDelayFromStatic				
RedistLoadingFromBgp	U32	4294967295	RCW	Specify the effective load on the link. Range: 0–255, where 255 is 100% loading. Default = 1
RedistLoadingFromConnected				
RedistLoadingFromRip				
RedistLoadingFromStatic				
RedistMtuFromBgp	U32	4294967295	RCW	Maximum Transmission Unit of the path in bytes. Default = 1500 (typical for Ethernet interface.)
RedistMtuFromConnected				
RedistMtuFromRip				
RedistMtuFromStatic				

Attribute Name	Type	Default	Access	Explanation
RedistReliabilityFromBgp	U32	4294967295	RCW	Default = 255, represents 100% reliability.
RedistReliabilityFromConnected				
RedistReliabilityFromRip				
RedistReliabilityFromStatic				
Name	String		RO	

OspfSummaryAddress object

Description

The OspfSummaryAddress is used to configure the advertising of OSPF routes for redistribution as a summary address. It is an aggregate list of addresses represented by a single IP address and subnet mask.

The use of summary addressing reduces the overhead incurred to manage the link-state database. Instead of advertising all OSPF routes encompassed by the summary address, a single summary route is advertised. The use of this configuration corresponds to the following command:

```
Router(config-router)# summary-address <ip-address mask> | <prefix-mask> [not-advertise] [tag <tag>]
```

Each interface in the list can be configured with a summary address list. The SuppressAdvertise attribute configures the site not to advertise the OSPF routes encompassed by the summary address and subnet mask. In other words, these routes are filtered out. This corresponds to the use of the not-advertise flag in the Cisco IOS command above.

The tag parameter is supported by the UseTag attribute.

Attributes

Attribute Name	Type	Default	Access	Explanation
IpAddr	IPAddress		RCW	The base address to be used as the summary address.
SubnetMask	IPAddress		RCW	Use this field to specify the number of set prefix mask bits. 32 corresponds to 255.255.255.255
SuppressAdvertise	Boolean	False	RCW	To specify the 'no' form of the summary-address command. When this is set to true, the summary addresses specified will not be advertised.
TagValue	U32		RCW	Tag value to be used as the tag parameter in the summary-address command configured on the device.
UseTag	Boolean	False	RCW	Enables the use of TagValue as the tag parameter in the summary-address command configured on the device.
Name	String	Content of IpAddr used as a string	RO	

BgpAggregateAddress

Attributes

Attribute Name	Type	Default	Access	Explanation
IpAddr	IPAddress		RCW	Aggregated IP address for the route summary that BGP will advertise.
SubnetMask	IPAddress		RCW	Aggregated subnet mask for the route summary that BGP will advertise.
Name	String	Content of IpAddr used as a string	RO	

BgpNetwork

Attributes

Attribute Name	Type	Default	Access	Explanation
IpAddr	IPAddress		RCW	IP address for the network that BGP will advertise.
SubnetMask	IPAddress		RCW	Subnet mask for the network that BGP will advertise.
Name	String	Content of IpAddr used as a string	RO	

SAP object

Description

The SAP (Service Application Point) object represents the application of a service on a device, without involving an interface. (Many services in Service Activator are modeled to be implemented on interfaces. However, some services can be implemented on devices as well, such as the Layer3 VPN Site.)

SAP objects are also used when interface-less VRFs are configured. The SAP (Service Application Point) object behaves similarly to an interface in this context, and provides a way of accessing and manipulating the VRF in the object model.

SAP objects are not visible in the GUI, but are accessible through the integration manager. When using the GUI, the creation of a SAP object occurs when a PE device is attached to a site. The device object icon is displayed in the Access Points folder under the Site to represent the SAP. However, no SAP objects are visible or accessible through the GUI.

You must perform explicit lifecycle management on the SAP objects you manipulate through the integration manager.

You cannot directly link a device into a VPN site using the OIM. Instead, you must create an SAP object under the Device and then link the SAP to the Site.

SAP objects acquire most Interface object attributes. Although they can be modified through the integration manager, attributes that are not relevant for SAP objects (e.g. attributes other than Name, Description and IsSAP) are ignored.

Creation

SAP objects must be created under the appropriate device. At creation time, the SAP requires a **Name**. An optional **Description** can also be provided. A read-only boolean attribute **IsSAP** will be set to True (boolean) to indicate that the object is an SAP.

Unlike the GUI created SAPs, integration manager created SAPs will have user-specified names.

Deletion

Removal/deletion of the SAP object must be performed manually. A SAP object must be unlinked from all the services in which it participates. It can then be deleted in the context of the parent device.

Linking/Unlinking

Once a SAP object has been created, it can be linked into appropriate services (such as a VPN Site) in the same manner as an Interface object. SAP objects can also be

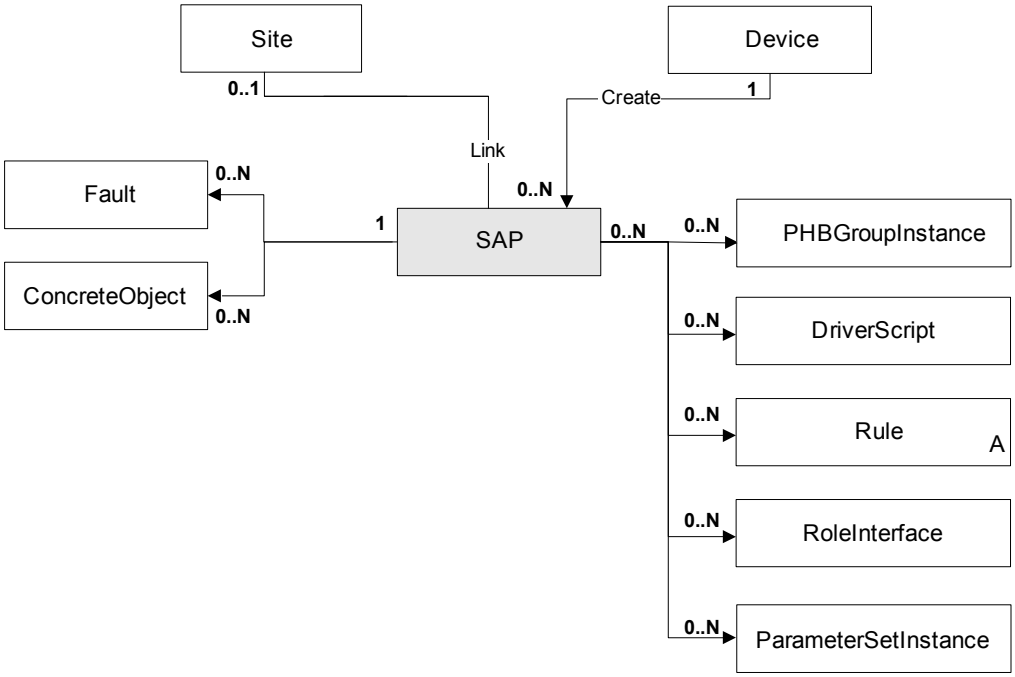
parents to **RoleInterface** objects. A SAP object must have a child RoleInterface of type **Access** in order to participate in a VPN service.

To create an interface-less VRF

To create an interface-less VRF using the OIM, follow these steps:

- Create the target VPN Site object under the appropriate customer
- Create a SAP object under the target device
- Link a RoleInterface of **Access** to the SAP object
- Link the SAP to the VPN site
- Modify the SAP attributes as required (e.g. static route redistribution)
- Modify the Site attributes as required (e.g. turn on static routing, specify the VRF name)
- Link the Site into the VPN

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RC	Name of SAP.
Description	String	""	RC	SNMP ifDescription.
IsSAP	Boolean	True	R	True if this is a Service Application Point (SAP) object.
Note: The remainder of the fields for the SAP object are identical to those of the Interface object. Many of these, however, have no meaning in the context of the SAP, even though they can be set through the Integration Manager.				

Object inheritance

SAP.Object

SubInterface object

Description

A SubInterface object represents a sub-interface on a device interface. One object exists for each sub-interface. Sub-interfaces are always linked to the parent interface. These objects are created by reading the interface table of each device.

Object diagram

See *Object diagram* on page 243.

Attributes

Inherited from interface object.

Object inheritance

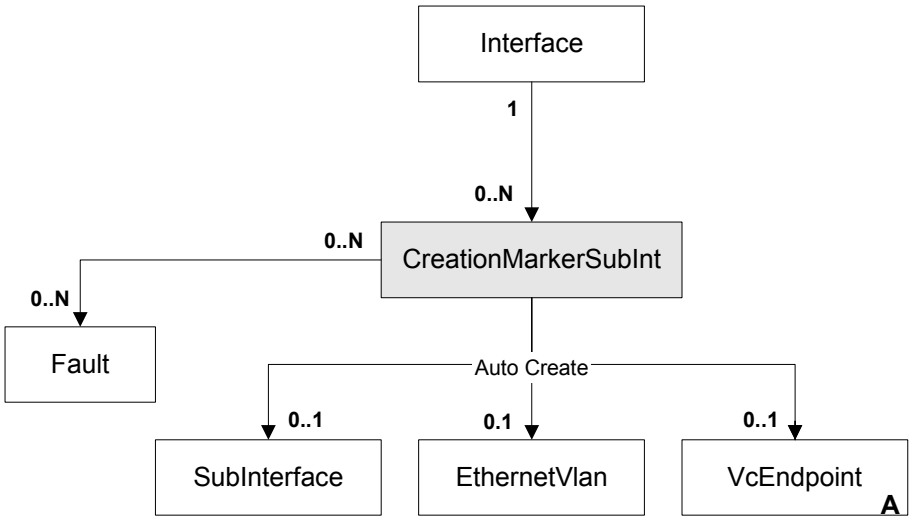
SubInterface.Interface.Object

CreationMarkerSubInt object

Description

A CreationMarkerSubInt object represents a with a PVC. When this object is created, a SubInterface and a VCEndPoint are created on the relevant device. When the object is deleted, the relevant interface gets removed.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String		RC	
SubInterfaceNumber	U32	0	RW	SubInterface number to be created.
Encapsulation	Enum	AtmAal5	RC	Encapsulation of the PVC created: 1 = AtmAal5 2 = AtmCell 3 = Frame 4 = Vlan

Attribute Name	Type	Default	Access	Explanation
CreationState	Enum	Inactive	RO	State of the SubInterface and PVC creation. This is comparable to the state of a concrete object. 0 = Inactive 1 = Active 2 = Installed 3 = Failed
Conflict	Boolean	False	RO	Indicates whether this object is in conflict with another policy element.

Object inheritance

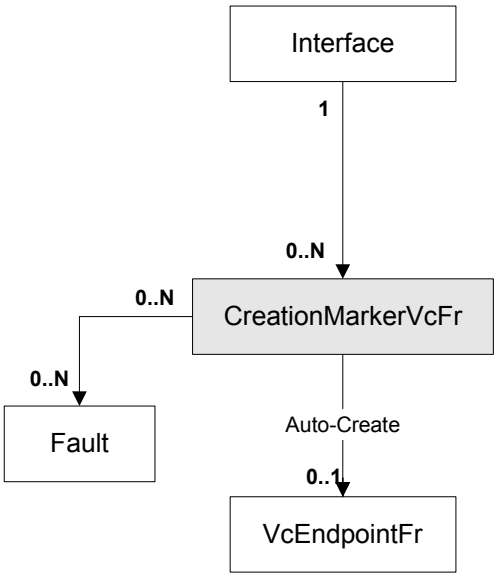
SubInterface.CreationMarkerSubInt.Object

CreationMarkerVcFr object

Description

A CreationMarkerVcFr object represents a Frame Relay VC endpoint, which is created directly under the interface. This can only be applied to some devices, depending on their capabilities.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RC	Simple accessory to this object.

Attribute Name	Type	Default	Access	Explanation
CreationState	Enum	Inactive	RO	State of the SubInterface and PVC creation. This is comparable to the state of a concrete object. 0 = Inactive 1 = Active 2 = Installed 3 = Failed
Conflict	Boolean	False	RO	Indicates whether this object is in conflict with another policy element.

Object inheritance

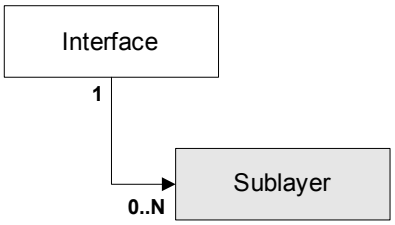
SubInterface.CreationMarkerVcFr.Object

SubLayer object

Description

A SubLayer object represents one protocol sublayer of an interface, as returned by the device during the discovery process. This is especially used for AAL5 sublayers on ATM interfaces.

Object diagram



Attributes

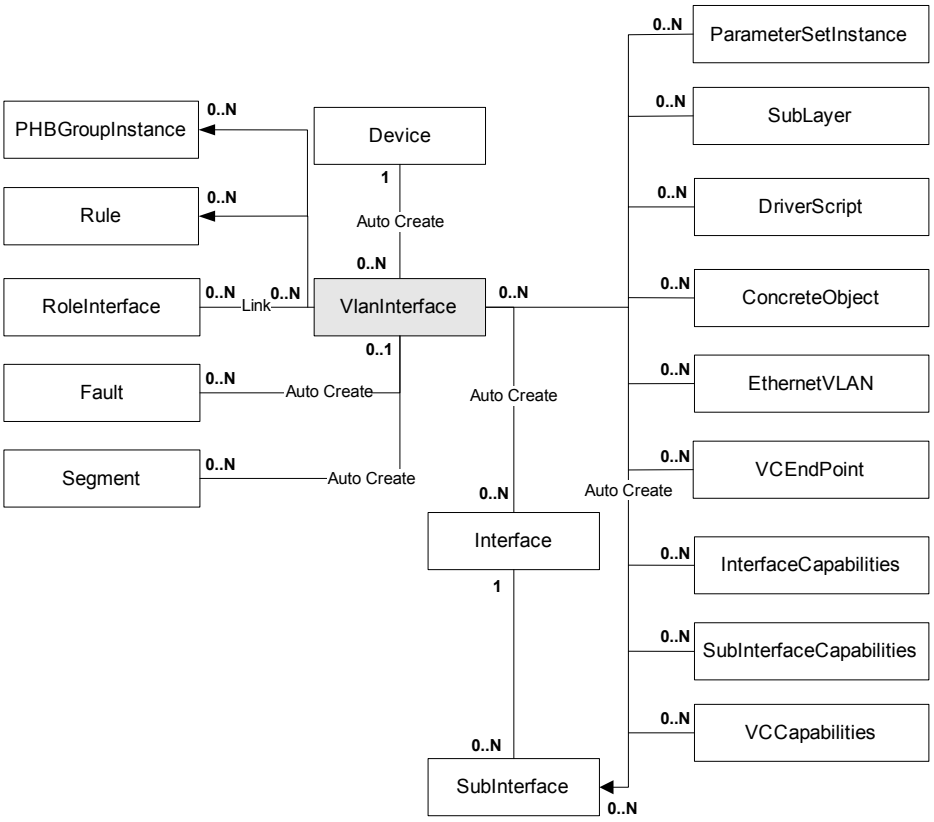
Attribute Name	Type	Default	Access	Explanation
Name	String	""	RO	Name of sublayer (SNMP ifDescr).
Number	U32	0	RO	Number of sublayer (SNMP ifIndex).
Speed	U32	0	RO	Speed of the sublayer in Kbits/s (SNMP ifSpeed parameter).
Type	U32	0	RO	Type of layer (ifType). This follows the IANA ifType numbering.

VlanInterface object

Description

A VlanInterface object represents a VLAN interface. One object is created for each VLAN present on a device when the device is discovered. All relevant interfaces, sub-interfaces and segments are linked to it. Note that although the object can have a Role, you cannot apply any policy element to it.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RC	Name of VLAN interface.
Description	String	""	RC	SNMP ifDescription.
Number	U32	0	RC	Number of interface (SNMP ifIndex).
Type	U32	1	RC	SNMP ifType.

Attribute Name	Type	Default	Access	Explanation
IpAddr	IPAddress	0.0.0.0	RC	IP Address of the VLAN interface.
SubnetMask	IPAddress	0.0.0.0	RC	Subnet mask.
PhysicalAddress	String	""	RC	Physical address of the interface (the MAC address).
Speed	U32	0	RC	Speed of the interface in Kbits/s (SNMP ifSpeed parameter).
VLANId	U32	0	RC	Vlan Number. Range: 0–4096

Object inheritance

VlanInterface.Object

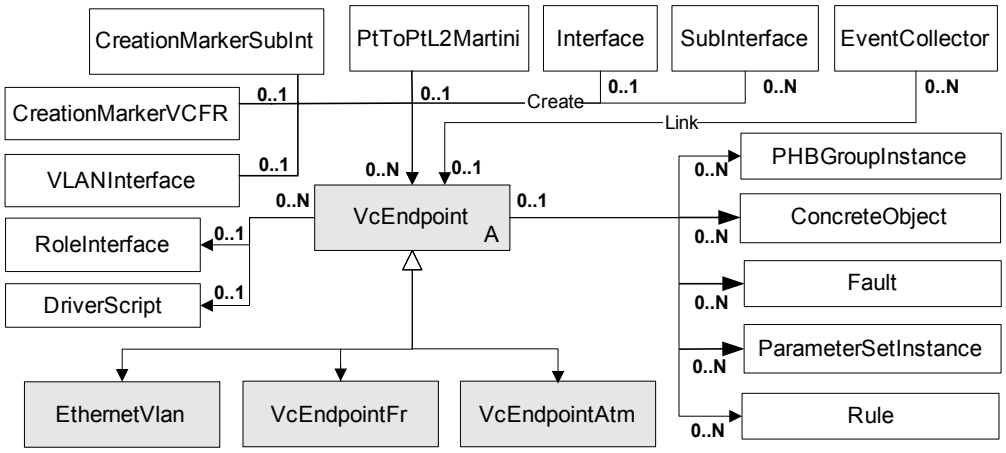
VcEndpoint objects

VcEndpoint object (abstract)

Description

The VcEndpoint abstract object maintains attributes common to all virtual circuit endpoints.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RC	Name of VC endpoint. For ATM constructed from VPI and VCI. For Frame Relay constructed from DCLI.
Context	String	""	RW	Local context for driver scripts applied to VC endpoints (max 512 bytes).
EffectiveCommand DeliveryMode (This value is inherited when the command delivery mode is set at the Device, parent Interface, or parent SubInterface level.)	Enum	0	RO	0 = Online: commands delivered to device 1 = OfflineMaintenance: commands not delivered, concretes and other states updated 2 = OfflineTest: commands not delivered.

Object inheritance

VcEndpoint.Object

VcEndpointFr object

Description

VcEndPointFr objects represent Frame Relay virtual circuit endpoints.

Object diagram

See *Object diagram* on page 289.

Attributes

Attribute Name	Type	Default	Access	Explanation
DLCI	U32	18	RC	Data Link Connection Identifier.
CircuitState	Enum	Inactive	RCW	State of the circuit: 1 = Invalid 2 = Active 3 = Inactive
CircuitType	Enum	PVC	RCW	Type of circuit: 0 = Unknown (Should never happen) 1 = PVC 2 = SVC (Service Activator only configures PVCs)
Context	String	""	RCW	Local context for driver scripts applied to VC endpoints (max 512 bytes).

Attribute Name	Type	Default	Access	Explanation
EffectiveCommandDeliveryMode (This value is inherited when the command delivery mode is set at the Device, parent Interface, or parent SubInterface level.)	U32	0	RCW	0 = Online: commands delivered to device 1 = OfflineMaintenance: commands not delivered, concretes and other states updated 2 = OfflineTest: commands not delivered.
Name	String	"DLCI: 18"	RC	Name of VcEndpointFr object.
ID	U32	11902	RO	

Object inheritance

VcEndpointFr.VcEndpoint.Object

VcEndpointAtm object

Description

VcEndPointAtm objects represent Asynchronous Transfer Mode (ATM) virtual circuit endpoints.

Object diagram

See *Object diagram* on page 289.

Attributes

Attribute Name	Type	Default	Access	Explanation
Vpi	U32	0	RC	Virtual Path Identifier.
Vci	U32	0	RC	Virtual Channel Identifier.
AdminStatus	Enum	3	RC	Desired administrative status of VC: 0 = Invalid (Should never happen) 1 = Up 2 = Down 3 = Unknown
OperStatus	Enum	3	RC	Current operational status of VC: 0 = Invalid (Should never happen) 1 = Up 2 = Down 3 = Unknown

Object inheritance

VcEndpointAtm.VcEndpoint.Object

EthernetVlan object

Description

An EthernetVlan object represents a virtual circuit endpoint on an Ethernet virtual local area network (VLAN).

Object diagram

See *Object diagram* on page 289.

Attributes

Attribute Name	Type	Default	Access	Explanation
VlanId	U32	0	RC	Vlan Number. Range: 1–4096
State	Enum	4 = Unknown	RC	0 = Down 1 = Up 2 = Testing 3 = Shutdown 4 = Unknown 5 = Dormant 6 = NotPresent 7 = LowerLayerDown 8 = NotFound 9 = NotYetDiscovered
Context	String	""	RCW	Local context for driver scripts applied to VC endpoints (max 512 bytes).
EffectiveCommandDeliveryMode (This value is inherited when the command delivery mode is set at the Device, parent Interface, or parent SubInterface level.)	U32	0	RCW	0 = Online: commands delivered to device 1 = OfflineMaintenance: commands not delivered, concretes and other states updated 2 = OfflineTest: commands not delivered.
Name	String	VLAN: 25	RC	Name of EthernetVlan object. For EthernetVlan constructed from VPI and VCI.
ID	U32	11854	RO	

Object inheritance

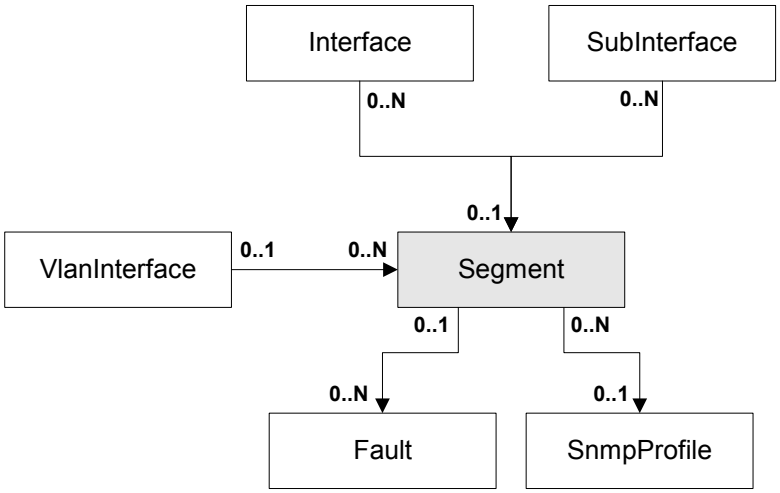
EthernetVlan.VcEndpoint.Object

Segment object

Description

For each interface which is not a point-to-point connection there will be one segment object representing the locally connected network segment.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RO	Name of segment.
SubnetAddr	IPAddress	0.0.0.0	RC	Subnet address.
SubnetMask	IPAddress	0.0.0.0	RC	Subnet mask.
Community	String	""	RC	SNMP Read Community string used when running autodiscovery on this segment.

Attribute Name	Type	Default	Access	Explanation
SnmpRetries	U32	2	RCW	The number of retry attempts when sending a PDU (Protocol Data Unit), Range: 0–20
SnmpTimeout	U32	3	RCW	Timeout of PDU response. Range: 1–30 seconds
Type	Enum	0	RC	Type of network segment: 0 = Unknown 1 = Other 2 = Serial 3 = BusSegment 4 = StarSegment 5 = TokenRing 6 = FddiRing 7 = AtmCloud 8 = FrCloud
Discovered	Boolean	""	RCW	True for segments created during discovery. False for segments created by the user, such as those linking a Virtual CE interface to the appropriate PE interface.

Object inheritance

Segment.Object

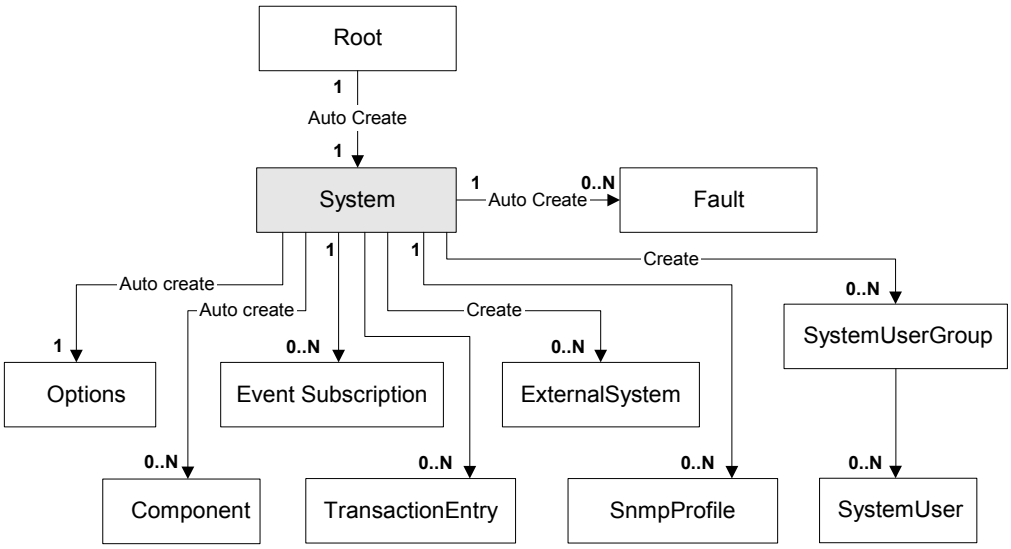
The System model

System object

Description

The System object represents the system root of the EOM. This object exists above all system objects.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	"System"	RO	Always "System".
ProxyPushState	Enum	0	RO	The current state of the push to the proxy agents (not the state of the proxy): 0 = Ready 1 = Push 2 = Pushing 3 = Invalid
ProxyPushProgress	U32	100	RO	When a proxy push is in progress, shows the percentage of the push that is complete.

Object inheritance

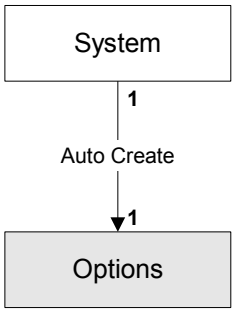
System.Object

Options object

Description

The Options object represents system-wide global options.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	"Options"	RO	Always "Options".
MajorVersion	U32	3	RO	Object Model Major version.
MinorVersion	U32	0	RO	Object Model Minor version.
ProxyAssign	Enum	1	RW	Global option defining the strategy for assigning devices to Proxy Agents automatically: 0 = Off 1 = AssignFirst 2 = LoadBalance
Revision	U32	0	RO	Object Model Revision Version
UnmanagedAction	Enum	1	RW	Default behavior to apply when unmanaging a device, unless overridden at device level: 0 = RemoveConfiguration 1 = LeaveConfiguration
TransactionLimit	U32	200	RO	Number of applied transactions to keep.

Object inheritance

Options.Object

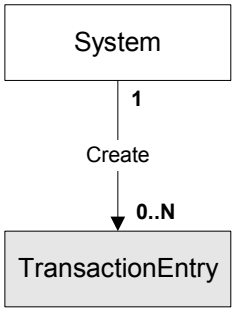
TransactionEntry object

Description

A TransactionEntry object represents each queued transaction, allowing the user to see a list of these transactions and to execute or roll back a transaction.

A TransactionEntry object can be created with the `commit` command, and manipulated with the `schedule`, `rollback` and `merge` commands.

Object diagram



Attributes

Attribute	Type	Default	Access	Explanation
Name	String	""	RW	Unique name for transaction.
Description	String	""	RW	Description of transaction entered by user, entered as a summary to describe the transaction.
Username	String	Name of user logged in	RO	Username of the user that created this transaction. Left empty if applicable to stored automatically-generated system transactions.
RollbackOnFailure	Boolean	True	RC	True unmerges (rolls back) the transaction if it fails. False does not roll back the transaction.
ReasonForFailure	String	""	RW	This attribute contains the messages from any faults raised for this transaction if the transaction monitor is running.

Attribute	Type	Default	Access	Explanation
State	Enum	0	RC	Status of this object: 0 = Pending 1 = Installed 2 = Uninstalled 3 = Scheduled 4 = ScheduledUninstall 5 = FailedScheduled 6 = FailedScheduleUninstall 7 = Committed 8 = ScheduleCommitted 9 = SucceededScheduleUninstall
ProvisioningStatus	Enum		RW	Provisioning status of this object: 0 = Pending — initial state on commit of the transaction. 1 = Succeeded — all concretes are in the Installed state or deleted. 2 = Failed — at least one concrete is in Rejected state, or the transaction timed out and the transaction monitor is configured to fail transactions on timeout. 3 = Timedout — the maximum time a transaction is allowed to stay in pending state awaiting completion notifications was exceeded. More details on these states are provided in the *Administrator's Guide*.
Schedule	DateTime	Now	RW	Date and Time to schedule this transaction for or when transaction was changed to current static state.

Attribute	Type	Default	Access	Explanation
Operations	String	-	RC	The data appearing below the data operations field is a deserialized representation of objects stored in a string type within the TransactionEntry object. The fields available for viewing represent the following: CID = ID of the concrete operated on Status = Inactive, Pending, Succeded, Failed, or Timedout Operation = Type of operation (Create, Delete or Link) PClass = Parent class PName = Parent name ParentID = ID of the parent object for the operation
NumberofConcretes	U32	-	R	The number of concretes for the transaction.

Object inheritance

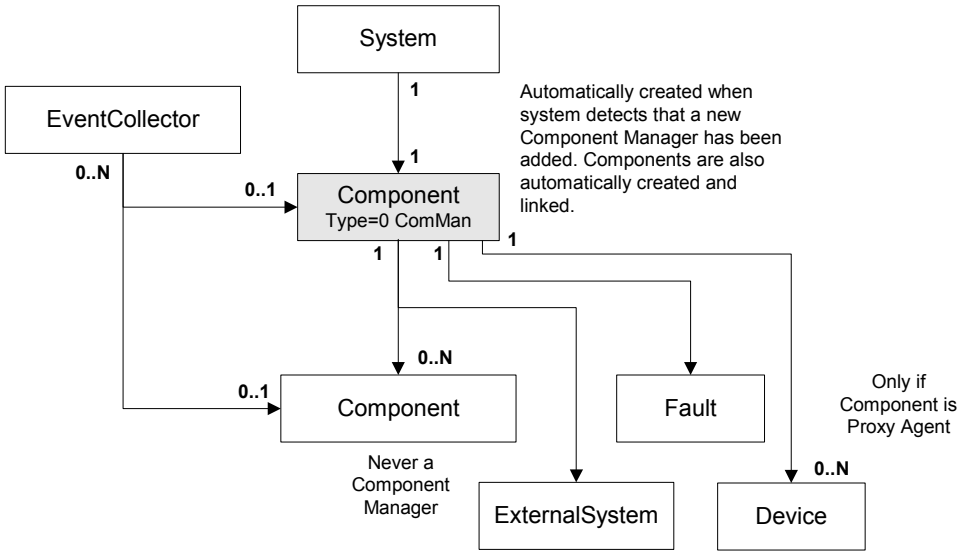
TransactionEntry.Object

Component object

Description

A Component object represents a Service Activator component: Component Manager, Policy Server, Event Handler, OIM, Proxy Agent, Device Driver, System Logger or Integration Component. The Type attribute identifies the type of component; individual components are not accessible via the EOM.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RO	Name of component.
Remarks	String	""	RO	Optional additional comments.
CorbaName	String	Auto	RO	CORBA name of component.
CorbaKind	String	Auto	RO	CORBA type of component.
CorbaLocation	String	Auto	RO	CORBA location of component.
CorbaMajor	U32	0	RO	Current version of the CORBA interface - major number.

Attribute Name	Type	Default	Access	Explanation
CorbaMinor	U32	0	RO	Current version of the CORBA interface - minor number.
RunState	Enum	0	RO	Allows triggering shutdown of components on a single host. Can be one of: 0-2 = running 3-5 = shutdown
StartTime	U32		RO	Time since this component started, in seconds.
Restarts	U32	0	RO	The number of times this component has been restarted.
Type	String	""	RO	Identifies type of component; one of: "ComMan" "PolicyServer" "ProxyAgent" "Driver" "SysLog" "ServiceComponent" "OssIntegrationManager" "EventHandler" "IntegrationComponent"
UpTime	String	0	RO	Elapsed time since the component started, specifying days, hours, minutes and seconds.
IsNetworkProcessor	Boolean	False	RO	Indicate if the proxy is the Networkprocessor or a regular proxy agent.

Attribute Name	Type	Default	Access	Explanation
Id	U32		RO	Unique ID used to reference this object.
Full Version	String	""	RO	The FullVersion attribute contains the values for the MajorVersion, MinorVersion, ServicePack and BuildNo.

Object inheritance

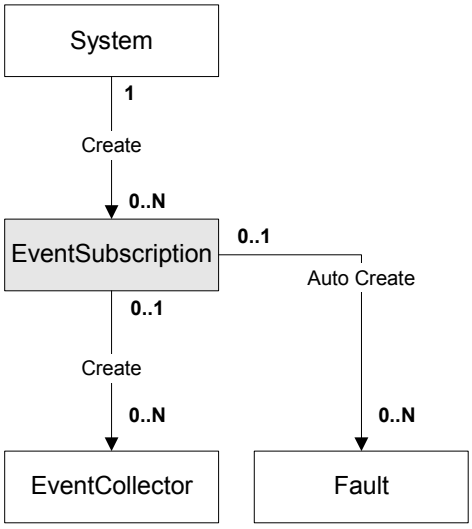
Component.Object

EventSubscription object

Description

An Event Subscription represents the definition of an external system subscribing to events and faults reported by Service Activator. The subscription defines the delivery type such as via an SNMP trap, NetCool, CORBA interface, etc.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	"InfoVista"	RCW	Name of subscription.
Description	String	""	RCW	A longer description.
Enable	Boolean	True	RCW	True = subscription enabled. False = subscription disabled.
SendPendingEvents	Boolean	False	RCW	True = currently pending events are to be delivered. False = pending events are not delivered.
DeliveryType	U32	4	RCW	Delivery method: 0=SnmpTrap 1=UpgradedSnmpTrap 2=Netcool 3=CorbaChannel 4=DatabaseOnly
DeliveryDetails	String	""	RCW	String formatted according to type. For SNMP, a comma-separated list of IP address, port and version, for example, "192.168.1.2,42,2". For CORBA and Netcool, the name of service. For DatabaseOnly: blank.

Attribute Name	Type	Default	Access	Explanation
ImpactTargets	U32	0	RCW	Bitwise value, 1 bit per impact target: Bit 0 = Customer, Site, VPN Bit 1 = Device Bit 2 = Interface
HighestSeverity	Boolean	False	RCW	True = delivers highest severity faults only. False = all faults delivered.
TransactionEvents	U32	0	RCW	Identifies transaction start/end events to be reported: 0 = None 1 = User 2 = System 3 = All

Object inheritance

EventSubscription.Object

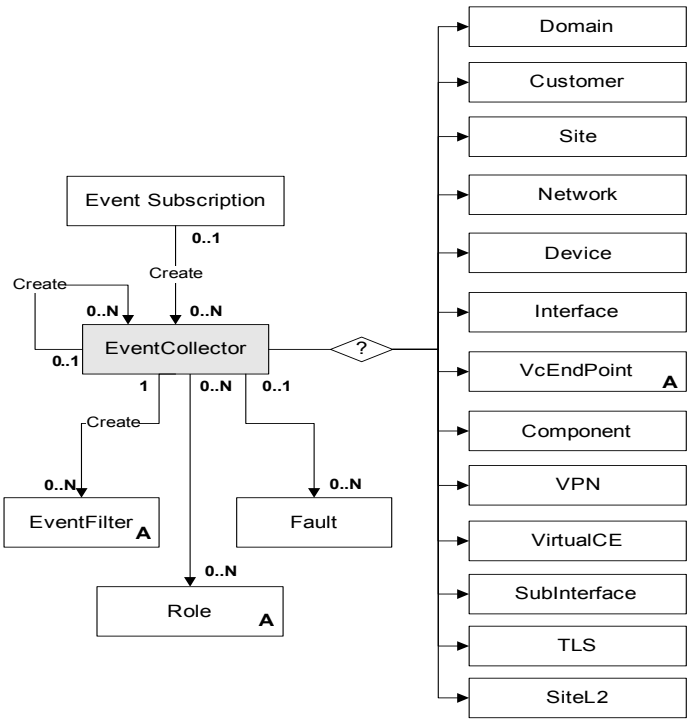
EventCollector object

Description

Event Collector objects are linked to other objects in the EOM to specify that the object is to be monitored.

To set the collection point to a particular object then you should link the object to the EventCollector. If the collection point is the Root object, then no object should be linked, but the RootCollection attribute should be set to True. If the collection point is the Policy or System objects, then no object should be linked, but the CollectionPoint attribute should be set to 1 or 2 respectively.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RW	Name of collector, must be unique.
Description	String	""	RW	Remarks.
CollectEvents	Boolean	True	RW	True = events are collected. False = events are not collected.

Attribute Name	Type	Default	Access	Explanation
CollectionPoint	U32	0	RW	Used if the collection point is the Policy (1) or System (2), as these objects cannot be linked directly.
RootCollection	Boolean	False	RW	True = Area of interest is the entire object model. False = Area of interest is defined by CollectionPoint.
HierarchicalCollection	Boolean	True	RW	True = collection is expanded below the local scope from the CollectionPoint. False = collection affects CollectionPoint only.
Type	Enum	1	RW	Specifies the relevant Event Type: 0 = CreateAndDelete 1 = Fault 2 = AttributeChange 3 = StateChange 4 = Link/Unlink
Classname	String	""	RW	Name of class being subscribed to. Must be a valid class name. Blank if subscription relates to a specific object.

Object inheritance

Subscription.Object

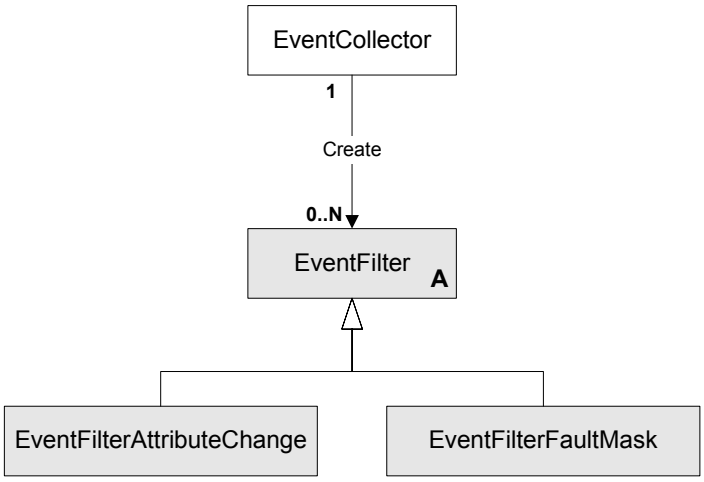
EventFilter objects

EventFilter object (Abstract)

Description

The EventFilter object is used by the Event Collector to determine which events are to be generated. It operates on event specific details such as the attribute that has changed on an object described in the event, or the fault ID raised.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	R	Name of event filter
Permit	Boolean	True	RCW	False=Deny matching events. True=Permit matching events.
Priority	U32	0	RCW	Ordering of filter.

Object inheritance

EventFilter.Object

EventFilterAttributeChange object

Description

The EventFilterAttributeChange object performs a match between an attribute name and the changes on an object to decide whether the event should be generated. The name corresponds to the type of filter.

Object diagram

See on page 310.

Attributes

Attribute Name	Type	Default	Access	Explanation
AttributeName	String	""	RCW	String to match against attribute name.

Object inheritance

EventFilterAttributeChange.EventFilter.Object

EventFilterFaultMask object

Description

The EventFilterFaultMask object performs a match between a triggered event and details of the faults to be reported. The name corresponds to the type of filter.

Object diagram

See page 310.

Attributes

Attribute Name	Type	Default	Access	Explanation
FaultCode	U32	0	RCW	Fault code; only used if FaultCategory is set to SingleFaultCode.
FaultCategory	U32	0	RCW	Category of faults to be filtered: 0=SingleFaultCode 1=ComponentFaultClears 2=DatabaseAccessFaultClears 3=ProvisioningFaultClears 4=PolicyRuleFaultClears 5=CommunicationFaultClears 6=DeviceOrInterfaceFaultClears 7=UpgradedComponentFaults 8=UpgradedDatabaseAccessFaults 9=UpgradedPolicyOrServiceFaults 10=UpgradedDeviceFaults 11=UpgradedInterfaceFaults 12=UpgradedLinkFaults 13=UpgradedSystemFaults 14=UpgradedConfigurationFaults

Object inheritance

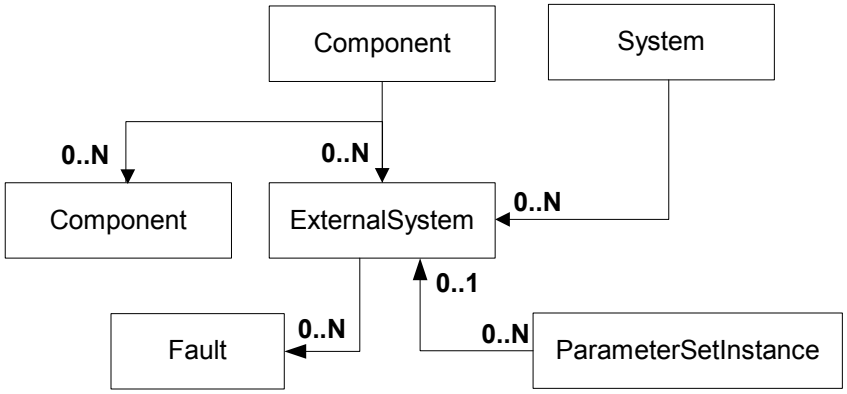
EventFilterFaultMask.EventFilter.Object

ExternalSystem object

Description

The ExternalSystem object represents an external system, and can be linked to a component of type IntegrationComponent.

Object diagram



Attributes

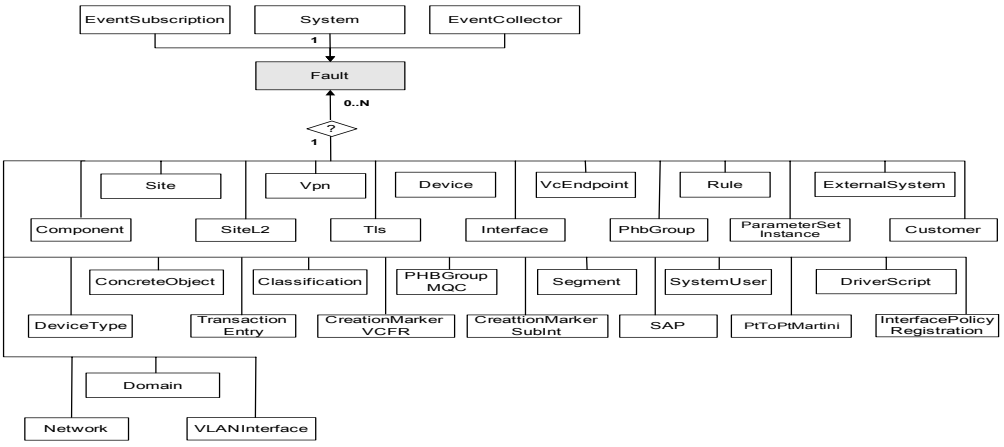
Attribute Name	Type	Default	Access	Explanation
Name	String	""	RW	Name of the specific instance of external system.
Remarks	String	""	RW	Optional additional comments.
Type	String	""	RW	Type of external system.
IpAddr1	IPAddress	0.0.0.0	RW	Primary IP address of external system.
IpAddr2	IPAddress	0.0.0.0	RW	Secondary IP address of external system.
Port1	U32	0	RW	Primary Port number.
Port2	U32	0	RW	Secondary Port number.
UserName	String	""	WO	User login name (encrypted).
Password	String	""	WO	User password (encrypted).
URL	String	""	RW	URL on which to contact external system.

Fault object

Description

A Fault object represents a system message indicating a fault reported from a Service Activator component or a network object. Fault objects exist while a component has an outstanding fault. When the fault is fixed the fault object is automatically removed. A fault object is linked to the object representing the component that currently has a fault.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RO	Name of the fault.
TimeStamp	DateTime	Now	RC	Time of fault creation.
MajorCode	U32	0	RC	Major error code (this is 100 for faults raised by Service Activator).
MinorCode	U32	0	RC	Minor error code.

Attribute Name	Type	Default	Access	Explanation
Severity	Enum	0	RC	The severity level of the fault. One of the following: 0 = Info 1 = Notice 2 = Warning 3 = Error 4 = Critical
Error	String	""	RO	Description text with parameter place holders.

Object inheritance

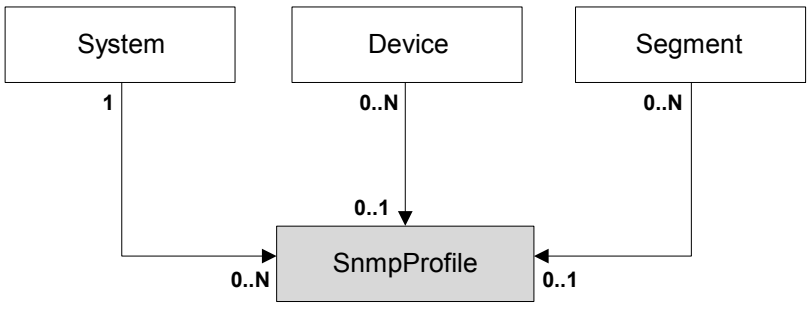
Fault.Object

SnmpProfile object

Description

An SnmpProfile object represents a user-defined profile of SNMP attributes that can be assigned to manage the discovery of a group of devices, or to the re-discovery of individual devices.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
Name	String	""	RW	Name given to this SNMP Profile
Description	String	""	RW	Text field describing this SNMP Profile
SnmpRetries	U32	2	RW	The number of retry attempts when sending a PDU (Protocol Data Unit). Range: 0–20
SnmpTimeout	U32	3	RW	Timeout of PDU response. Range: 1–30 seconds
SnmpVn	Enum	3	RW	Version of SNMP to use to interrogate the device: 0 = None 1 = SnmpV1 2 = SnmpV2c 3 = SnmpV1V2c 4 = SnmpV3 5 = SnmpV1V3 6 = SnmpV2cV3 7 = SnmpV1V2cV3
Community	Secure	public	RW	SNMP Community string used for Read access to network devices, when running autodiscovery
Userid	Secure	noAuth User	RCW	UserID authorized to set authentication and privacy passwords

Attribute Name	Type	Default	Access	Explanation
Authentication	Enum	None	RW	0 = None 1 = SHA 2 = MD5
Authentication Password	String	""	RW	Password length: 8-127 characters
Privacy	Enum	None	RW	0 = None 1 = DES 2 = AES
PrivacyPassword	String	""	RW	Password length: 8-127 characters
Id	U32	541	RO	Unique ID used to reference this object

Object inheritance

SnmpProfile.Object

SystemUserGroup objects

SystemUserGroup object

Description

The SystemUserGroup object is read only. The administrator must log into the GUI to create a system user group.

A system user group defines the access level that its members have within Service Activator. Every system user is a member of only one system user group. All users in the same system user group have the same access privileges.

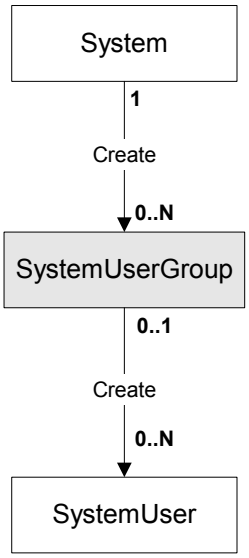
A system user group can have one of the following access levels:

- Super User – users can access the user interface, view information and perform any operation. Only users with Super User access can set up other users and perform configuration tasks associated with the policy server.
- Read Write – users can access the user interface, view information, and create and/or modify objects. The operations that can be performed by group members depend on the permissions that are set for the group.

- Read Only – users can access the user interface and view objects but cannot create new or modify existing objects.

For system user groups with Read Write access, you can specify exactly which operations its members can perform. For example, you can specify that members cannot create new networks, but can discover new devices. These controls are referred to as permissions.

Object diagram



Attributes

Attribute Name	Type	Default	Access	Explanation
AccessRights				None = 0, SystemWrite = 1, SystemRead = 2, WorldWrite = 4, WorldRead = 8, ReadOnly = 8, ReadWrite = 8 + 4, SuperUser = 8 + 4 + 2 + 1 Only a Super user sees System objects.
Name	String	""	R	Name of the user group.
Id	U32	0	R	This is a unique ID used to reference this object.

Object inheritance

SystemUserGroup.Object

SystemUser object

Description

The SystemUser object is used to create new users and set security restrictions. System users are set up within a system user group. All users in the same system user group have the same access privileges. A user can be a member of only one group.

Users with Super User access can set up new system users, specifying their user name and password details. It is possible to force users to change their passwords when they first log into Service Activator.

Object diagram

See page 318.

Attributes

Attribute Name	Type	Default	Access	Explanation
Active	U32	0	R	This value indicates the number of concurrent sessions that the user is running. 0 indicates that the user is not currently logged on.
Concurrent	Boolean	False	RCW	False prevents more than one concurrent logon by this user
Enabled	Boolean	True	RCW	False disables user logon.
ExpireDate	Date	[date]	R	Password expiry date is calculated from the value entered for PasswordExpires. The default [date] is 30 days after the user is created.
FailedLogins	U32	0	R	The number of failed logon attempts before the user is denied access (locked out).
Name	Secure	none	RCW	Name of the user.
Password	String	""	RCW	User password string.
PasswordExpires	U32	30	RCW	Number of days between password change and password expiry. A value of 0 means "password never expires". When login is blocked due to password expired, the user must log into the GUI and change the password. This will enable login to the OSS Integration Manager.
Remarks	String	""	RCW	Add any desired text.

Attribute Name	Type	Default	Access	Explanation
ResetPassword	Boolean	False	RCW	True requires the user to reset the password at initial logon. When this attribute is True (that is, when the password is expired), the user cannot log into the OSS Integration Manager. When login is blocked due to reset required by the system administrator, the user must log into the GUI and change the password. This will enable login to the OSS Integration Manager.
Id	U32	0	R	A unique reference number for each object.

Object inheritance

SystemUser.SystemUserGroup.Object

Chapter 4f

Examples of Using OIM

This chapter provides examples of using the OIM commands to perform a number of functions within Service Activator. It includes the following examples:

- Browsing the topology
- Discovering devices
- Pre-provisioning devices
- Creating roles and assigning them to devices and interface
- Creating a site
- Creating a management VPN
- Creating a customer VPN
- Creating policy rules and applying them to an object
- Creating and applying standard and MQC PHB groups
- Setting up a subscription to report events occurring in the network
- Applying parameter sets to configuration targets
- Creating an SAA template and applying it to VPN
- Determining when transactions have been completed
- Managing system users
- *Configuration Thresholding feature - modifying the regular expression*

Explanation of the examples

This chapter consists of a number of worked examples showing some of the most common uses of OIM, from discovering devices to setting up management and customer VPNs and applying policy rules to them. The examples are based on the use of the OIM CLI but as they are intended as general illustrations, they can be adapted for use in scripts.

For illustrative purposes, example screens are included showing the corresponding appearance of the user interface.

The examples are based on a simple example network, initially with a single domain called "iisp".

Browsing the topology

A set of commands are available to allow users to navigate through the system and display information.

Pre-requisites

None

Command syntax

The following commands allow you to navigate through the network topology. See *The Navigation Module* on page 46 for full details.

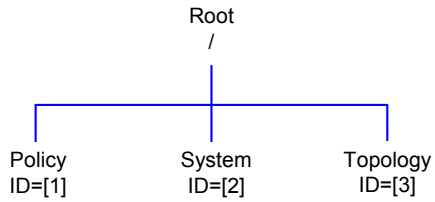
Command	Purpose
`getPath [`*`object-path`*`]`	Returns the path to the current position in the EOM
`setPath [`*`object-path`*`]`	Sets the current path within the EOM
`getChildren [`*`object-path`*`]`	Lists the children of the object
`getName [`*`object-path`*`]`	Returns the name of an object
`getparents [`*`object-path`*`]`	Lists the parents of an object
`getAttributes [`*`object-path`*`]`	Lists the attributes of an object
`getID [`*`object-path`*`]`	Returns the ID of an object

Object hierarchy

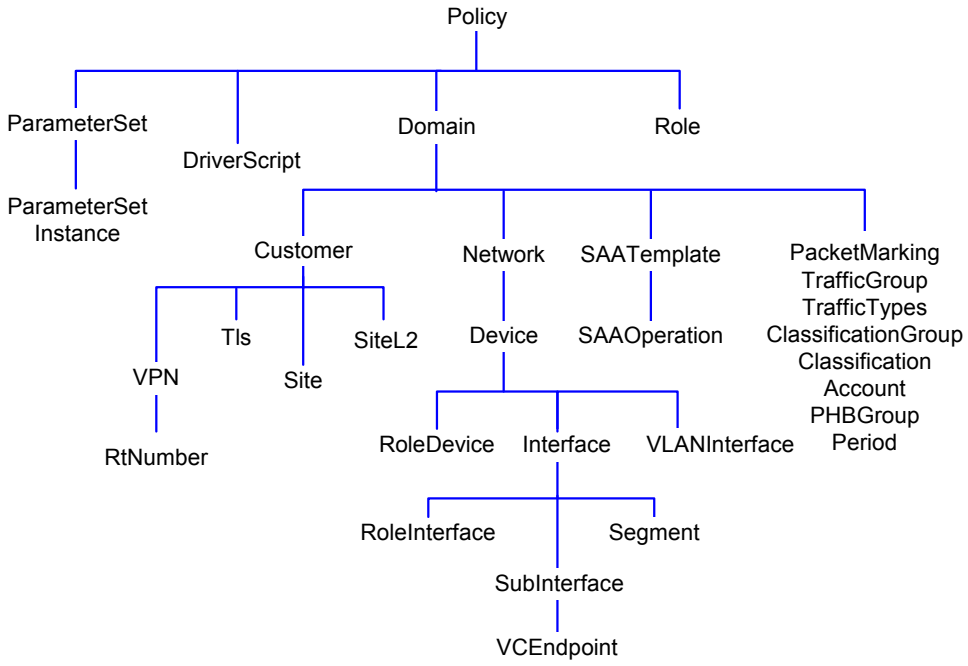
The following diagrams illustrate the basic structure of the object hierarchy for navigation purposes.

Root object hierarchy

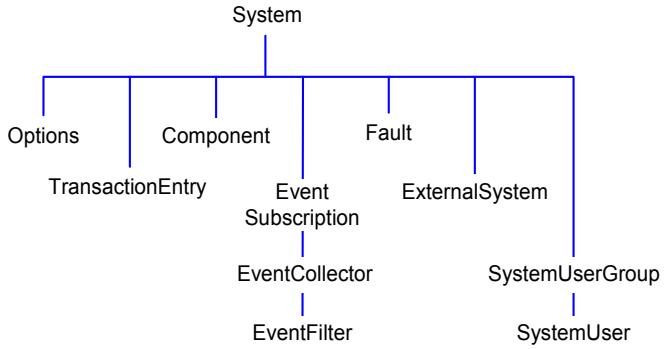
Note the standard ID numbers for these objects:



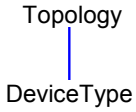
Policy object hierarchy



System object hierarchy



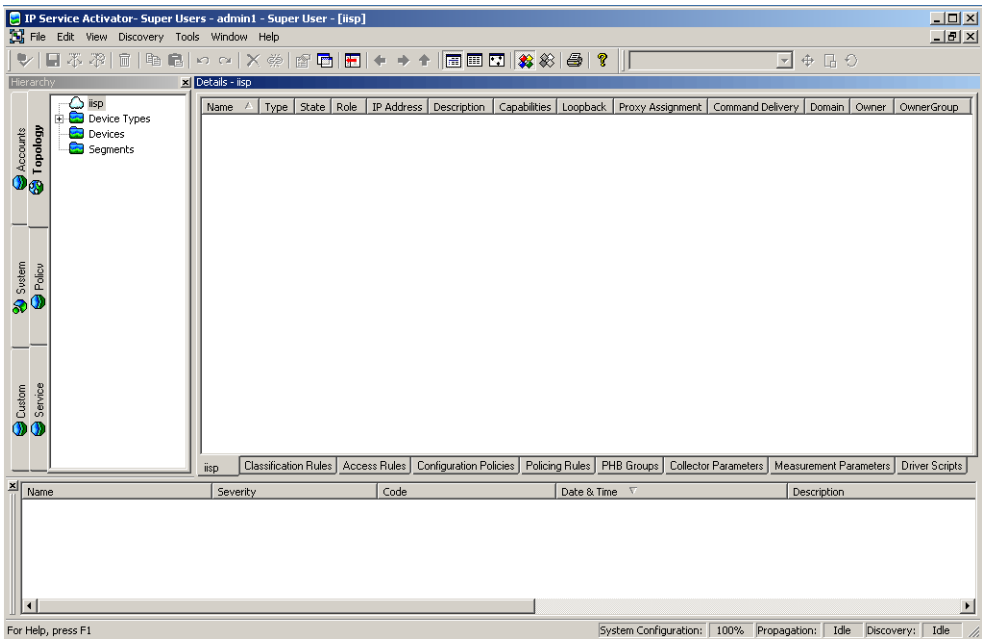
Topology object hierarchy



Note that whenever further objects are created or existing objects linked they appear as children of the appropriate object. For example, if a rule is created at the device level it appears as a child of the device, or if a PE interface is linked to a site, it will appear as a child of the site.

Example

The following examples show commands to navigate through the system and list the pre-existing objects. This assumes one domain called "iisp":



To browse the policy part of the EOM, set the path to "policy":

```
setPath policy:policy
    /Policy:"Policy"
/Policy:"Policy">getchildren
    [4] RoleDevice:"Access"
    [5] RoleDevice:"Gateway"
    [6] RoleDevice:"Core"
    [7] RoleDevice:"Any"
    [8] RoleInterface:"Core"
    [9] RoleInterface:"Customer"
    [10] RoleInterface:"Local"
    [11] RoleInterface:"Access"
    [12] RoleInterface:"Any"
    [272] Domain:"iisp"
```

You can then view the domain:

```
/Policy:"Policy">setPath domain:iisp
    /Policy:"Policy"/Domain:"iisp"
/Policy:"Policy"/Domain:"iisp">getchildren
    [273] Network:"OrchDemo"
    [285] PacketMarking:"IP Precedence 0"
    [286] PacketMarking:"IP Precedence 1"
    [287] PacketMarking:"IP Precedence 2"
    [288] PacketMarking:"IP Precedence 3"
    [289] PacketMarking:"IP Precedence 4"
    [290] PacketMarking:"IP Precedence 5"
    [291] PacketMarking:"IP Precedence 6"
    [292] PacketMarking:"IP Precedence 7"
    [293] TrafficGroup:"Standard Traffic Types"
    [297] TrafficGroup:"Standard Port Numbers"
    [647] ClassificationGroup:"Standard Traffic Types"
    [651] ClassificationGroup:"Standard Port Numbers"
    [1749] Customer:"Customer A"
    [1756] Customer:"Management"
```

You can then view the network and view the devices that have been set up:

```
/Policy:"Policy"/Domain:"iisp">setPath network:"OrchDemo"
    /Policy:"Policy"/Domain:"iisp"/Network:""

/Policy:"Policy"/Domain:"iisp"/Network:"">getChildren
    [1212] Device:"G3640-4.Orchestream.com"
    [1213] Device:"G3640-6.Orchestream.com"
    [1214] Device:"G3640-3.Orchestream.com"
    [1570] Network:"CORE NET"
    [1847] Device:"A2501-3.Orchestream.com"
```

Finding objects

The find command allows you to search for a particular character string. The syntax is as follows.

Pre-requisites

The objects must exist.

Command syntax

The basic syntax of the `find` command is:

`find [`*`object-path`*`] `*`search-string attributes`* `finddirection=`*`direction`*

object-path	The ID or path of the object from which to start the search. If this is omitted, the search will start from the current point in the path.
search-string	A string, enclosed in quotes, containing the object to look for. It may contain wildcards.
attributes	Any number of attributes may be specified. An object will only satisfy the search criteria if it matches the search string and has the same attributes as those given.
direction	Indicates the direction of the search – `child` or `parent`. By default the Find operation searches downwards through the hierarchy, from parent to child, but you can set it to search upward, from child to parent.

Examples

To find all objects from the current location:

```
find . "*"
```

To search for a domain in a current directory:

```
find . Domain:"*"
```

To find a specific object from the current location:

```
find . Device:"DeviceName"
```

To search up the tree from the current location:

```
find . Customer:"*" finddirection=parent
```

If you are unsure of where an object is in the object model, it is a good idea to start the search from the root. For example:

```
find / device:"G*"
    [1152] Device:"G4500-1"
    [1178] Device:"G4500-3"
    [1225] Device:"G3640-3"
    [1266] Device:"G7204-1"
    [1817] Device:"GCAT5509"
    [1837] Device:"G3640-7"
    [1852] Device:"G-6509"
```

Discovering devices

This example describes how to discover devices using OIM. It is possible to discover new devices, set security/authentication parameters and fetch the capabilities of interfaces using OIM in exactly the same way as from the user interface.

Note that the discovery can take some time especially if a large number of devices are discovered.

Pre-requisites

- A domain must previously have been created using the UI.
- The ID or name of the network object which will be the parent of the discovered devices must be known.

Command syntax

The basic syntax for the `discover` command is:

`discover` *`network-id parameters`*

The *`network-id`* is the ID of the network object which will be the parent of the discovered devices.

The *`parameters`* specify the type of discovery, the IP addresses (or DNS names) to discover, and the security settings that will be required to authenticate to the device. The Type parameter defaults to Discover, so for an ordinary discovery this can be omitted. The security settings are required in order to configure the device or to obtain the capabilities of specific interfaces, so if they are not set on discovery they must be set afterwards and the capabilities re-fetched.

Discovering a single device

To discover a single device, use the `discover` command specifying the IP address of the device to be discovered.

```
discover network-id ipaddress=10.0.0.22 mask=32
discover network-id DnsName=foo
```

This will discover the device, creating it as a child of the network object.

Discovering a single device using an SNMP Profile

To discover a single device using an SNMP Profile, when the authentication option is selected:

```
discover network-id ipaddress=10.0.0.22 mask=32
SnmpProfileName=xxxxxxxx
```

Discovering a subnet

To discover a whole subnet, use the `discover` command with the following format:

```
discover network-id ipaddress=subnet-address mask=24
```

Fetching capabilities

To fetch device and interface capabilities, you need to provide security information for the devices. The exact command syntax depends on the type of security required.

Anonymous Telnet

```
discover network-id IpAddress=IPaddress AccessStyle=Anonymous
LoginPassword=loginpassword EnablePassword=enablepassword
```

Named User

```
discover network-id IpAddress=IPaddress AccessStyle=NamedUser
UserName=username LoginPassword=loginpassword
EnablePassword=enablepassword
```

TACACS

To discover using TACACS, the UserName, ResponseStrings, and EnablePassword attributes of the Network object must be set first:

```
modify network-id UserName=username ResponseStrings=responsestrings
EnablePassword=enablepassword
commit
```

The device can then be discovered using a command as follows:

```
discover network-id IpAddress=IPaddress AccessStyle=TACACS
```

Rediscovering a device

To rediscover an existing device, simply use the `discover` command with the ID of the device:

```
discover device-id
```

Stopping discovery

To stop all discoveries in progress, use:

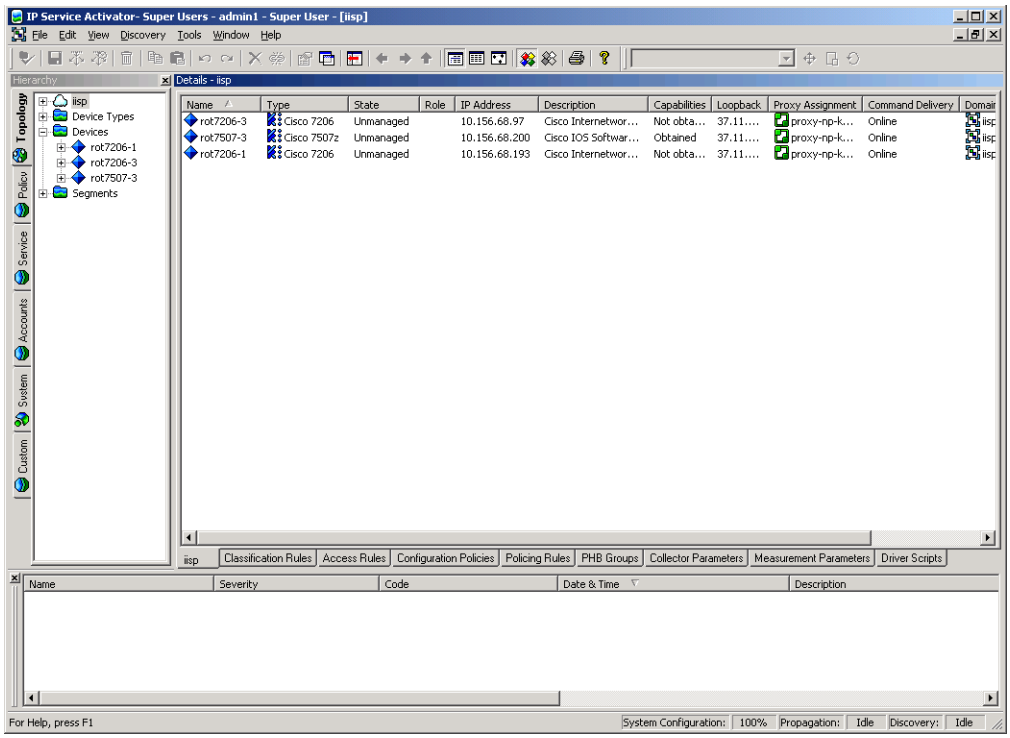
```
discover Type=Stop
```

Example

In this example, three devices are discovered within a network (ID=18)

```
discover [18] IpAddress=10.0.0.91 AccessStyle=anonymous
LoginPassword=cisco EnablePassword=cisco
discover [18] IpAddress=10.0.0.92 AccessStyle=anonymous
LoginPassword=cisco EnablePassword=cisco
discover [18] IpAddress=10.0.0.93 AccessStyle=anonymous
LoginPassword=cisco EnablePassword=cisco
/>GetChildren [18]
   [2707] Device:"rot7206-1"
   [2731] Device:"rot7206-3"
   [2799] Device:"rot7507-3"
```

The user interface is as follows:



The three devices appear as children of the network object. As part of the discovery, any role assignment rules that have been set up using the user interface will also be applied to the newly-discovered objects.

Creating Device and Interface objects for pre-provisioning

Device objects, and interfaces on those devices, can be created directly using the OIM to permit pre-provisioning.

Pre-requisites

- A domain must have been created using the user interface.

Command syntax

If the required DeviceType object for the device does not exist, then it must be created as a child of the Topology object:

```
setPath /Topology:"Topology"
create DeviceType:"name"
modify DeviceType:"name" attributes
```

Devices and interfaces may be created in the same way as any other object using the `create` command. Note that Device and Interface objects contain a number of attributes whose values can only be set on creation. See *DeviceType object* on page 227, *Device object* on page 220, and *Interface object* on page 243

```
setPath network-id
create Device:"device-name" IpAddr=IPaddress DeviceType=device-
type-name IsVirtual=False ...
modify Device:"device-name" Name=new-device-name
commit
```

device-type-name is the name of a DeviceType object that is a child of the Topology object.

Interface objects can then be created as children of the Device object:

```
setPath device-id
create Interface:"interface-name" IpAddr=IPaddress SubnetMask
=subnet-mask ...
modify Interface:"interface-name" Speed=new-interface-speed
commit
```

Once the Device and Interface objects have been created, and the correct security attributes set, the real devices can be discovered and associated with the Device object by using the discover command:

```
discover device-id
```

Example

In this example a device and an interface are created. The device is created as a child of the network object (ID=18). This example assumes that the DeviceType object already exists as a child of the Topology object.

```
setPath [18]
create Device:"New Device" IpAddr=1.2.3.4 DeviceType="New Type"
IsVirtual=False
```

```
commit
setPath Device:"New Device"
create Interface:"NewInterface" ipAddr=5.6.7.8
SubnetMask=255.255.255.0
commit
```

Creating and assigning roles to devices and interfaces

Before devices and interfaces can be configured, you need to assign roles to them in order to define the services and policies that can be applied to them. You can assign both system-defined roles and user-defined roles, and create new user-defined roles if required.

Pre-requisites

- The Device and Interface objects to which you are applying roles must already exist.

Command syntax

Creating a new role

To create a new device or interface role, set the path to the Policy object, and use the `create` command to create the roles:

```
setPath /Policy:"Policy"
create RoleDevice:"new-device-role"
create RoleInterface:"new-interface-role"
```

Assigning a role to a device, interface, or subInterface

To assign a role (either a newly-created one, or an existing one) to a device, interface, or sub-interface simply link the role to the object.

```
link [object-id] [role-id]
```

Note that RoleDevice objects can only be children of Device objects, and RoleInterface objects can only be children of Interface or SubInterface objects.

Example

In this example, roles are assigned to the three devices discovered in *Discovering devices* on page 330. The PE device is assigned a role of "Gateway" while the two CE

devices are assigned the role of "Access". The interfaces linking the devices are all assigned the "Access" interface role.

Object IDs are as follows:

```
RoleDevice:"Gateway"=[5]
RoleDevice:"Access"=[4]
RoleInterface:"Access"=[11]
```

To assign appropriate roles to the devices:

```
link [2799] /Policy:"Policy"/RoleDevice:"Gateway"
link [2707] /Policy:"Policy"/RoleDevice:"Access"
link [2731] /Policy:"Policy"/RoleDevice:"Access"
commit
```

Alternatively, use the object IDs, such as:

```
link [2731] [4]
```

To assign appropriate roles to the interfaces:

```
link [2865] [11]
link [2867] [11]
link [2838] [11]
link [2962] [11]
commit
```

Creating a site

This section explains the steps involved in creating a site and setting up the routing parameters.

Pre-requisites

- Sites are associated with PE Access interfaces. They can also be associated with CE devices if they are to be managed, and PE devices, when creating an interface-less site. These devices and interfaces must have been discovered and appropriate roles must be assigned.

Command syntax

Create Customer object

Sites must be created as children of Customer objects, so you need to create a customer object as a child of the Domain if it does not already exist.

```
setPath domain-id
create Customer:"name"
commit
```

Create Site object

The site can then be created as a child of the required customer:

```
setPath customer-id
create Site:"name"
commit
```

Link PE Access interface to site

The appropriate interface on the PE device needs to be linked to the site:

```
link site-id interface-id
commit
```

Link CE router to management site

If the site is to contain a CE router, then that must also be linked to the site:

```
link site-id ce-device-id
commit
```

Manage the CE router

If the site contains the CE router, then it should be managed.

```
manage ce-device-id
commit
```

Setting up PE-CE routing parameters

The routing parameters between the PE and the CE can then be set. This involves modifying certain attributes on the site.

Routing protocol

To configure the appropriate protocol, the Routing Protocol must be set to **RIP, EBGP, OSPF**, or **NONE**.

```
modify site-id RoutingProtocol=value
commit
```

For RIP and OSPF, no further parameters need to be set. If the RoutingProtocol is set to **EBGP**, then additional attributes should be set.

- The BgpAsn attribute on the Site object should be set to the ASN of this site.
- The PrivateCeIpAddress attribute on the Interface object should be set to the IP address of the access interface on the CE.

```
modify management-site-id BgpAsn=value
modify interface-id PrivateCeIpAddress=value
commit
```

Static routing

Setting the InstallStatic attribute on the Site object to True will specify static routing. The destinations that can be reached from the PE Access interface can then be set by creating StaticRoute objects as children of the PE Access interface and setting their attributes:

```
setPath interface-id
create StaticRoute:""
modify static-route-id ipaddr=ce-ipaddr subnetmask=mask
nexthop=ce-access-interface-ipaddr
commit
```

If multiple static routes are available then a StaticRoute object should be created for each destination accessible from the PE Access interface.

Example

In this example a site is created that contains the PE access interface "Ethernet 0/1" and the CE device "A1720-1". This site is used as the management site of a management VPN.

The routing protocol for the site is EBGP, and the address of the access interface on the CE device is 10.136.135.2. Static routing to the CE device is also configured.

OIM Commands

First, the customer object is created as a child of the Domain object (ID=30):

```
setPath [30]
create Customer:"Management"
commit
```

Next the Site object is created as a child of the newly-created customer:

```
setPath Customer:"Management"
create Site:"Management Hub"
commit
```

Now the PE access interface (ID=2867) and the CE device (ID=2837) must be linked to the site:

```
link Site:"Management Hub" [2867]
link Site:"Management Hub" [2837]
commit
```

Now the CE and PE devices can be managed:

```
manage [2837]
manage [2861]
commit
```

Next the routing protocol for the site should be set to EBGP, and the BgpAsn attribute is set to a nominal value (in this case 60).

```
modify Site:"Management Hub" RoutingProtocol=EBGP BgpAsn=60
commit
```

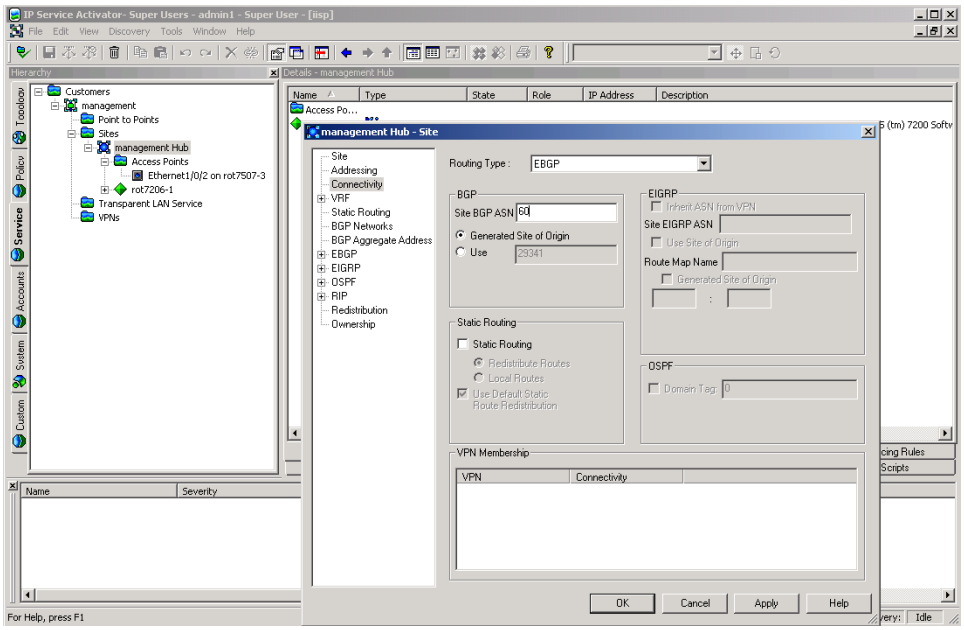
The PrivateCeIpAddr attribute on the PE access interface (ID=2867) in the site should be set to the IP address of the CE's access interface.

```
modify [2867] PrivateCeIpAddr=10.135.135.2
commit
```

Now the static route is configured. A StaticRoute object is created as a child of the PE access interface (ID=2867), and the attributes of the object are set accordingly.

```
setPath [2867]
create StaticRoute:"0.0.0.0"
modify StaticRoute:"0.0.0.0" ipaddr=10.0.0.91
subnetmask=255.255.255.255 nexthop=10.135.136.2
commit
```

The following screen shows that a customer and a site have been created. The **Connectivity** properties page shows that the ASN has been correctly set.



Creating a management VPN

This section explains the steps involved in setting up a management VPN.

Pre-requisites

- The management VPN must contain a site that will be the hub of the VPN, and any customer sites that are also present in their own VPNs. These sites must have been previously set up. It is usual to create a management customer that will contain the management site and the management VPN object.

Command syntax

Creating the VPN object

The first step is to create a VPN object as a child of the management customer:

```
setPath customer-id
create VPN:"ManagementVPN"
commit
```

The VPN must be configured as a management VPN:

```
modify vpn-id MplsVpnType=Management
commit
```

Link the management site to the VPN, and make it a hub site

The Management Site object can now be linked to the VPN. This site should be set up as a hub site of the management VPN. To do this, after linking the site to the VPN, create a SiteHub object as a child of the VPN with the same name as the management site:

```
link vpn-id management-site-id
setPath vpn-id
create SiteHub:"management-site-name"
commit
```

Link customer sites to the VPN

So far only one site has been added to the management VPN. Any other sites linked to customer VPNs should also be linked to the management VPN object:

```
link vpn-id customer-site-id
commit
```

Example

In this example, a management VPN is created for the Management customer. The Management site is added to the VPN, and configured to be a hub. A single customer site is then added to the management VPN as a spoke.

Firstly the management VPN object should be created as a child of the Management customer (ID=3000).

```
setPath [3000]
create VPN:"Management VPN"
modify VPN:"Management VPN" MplsVpntype=Management
commit
```

Now the management site (ID=3002) should be linked to the VPN object:

```
link VPN:"Management VPN" [3002]
commit
```

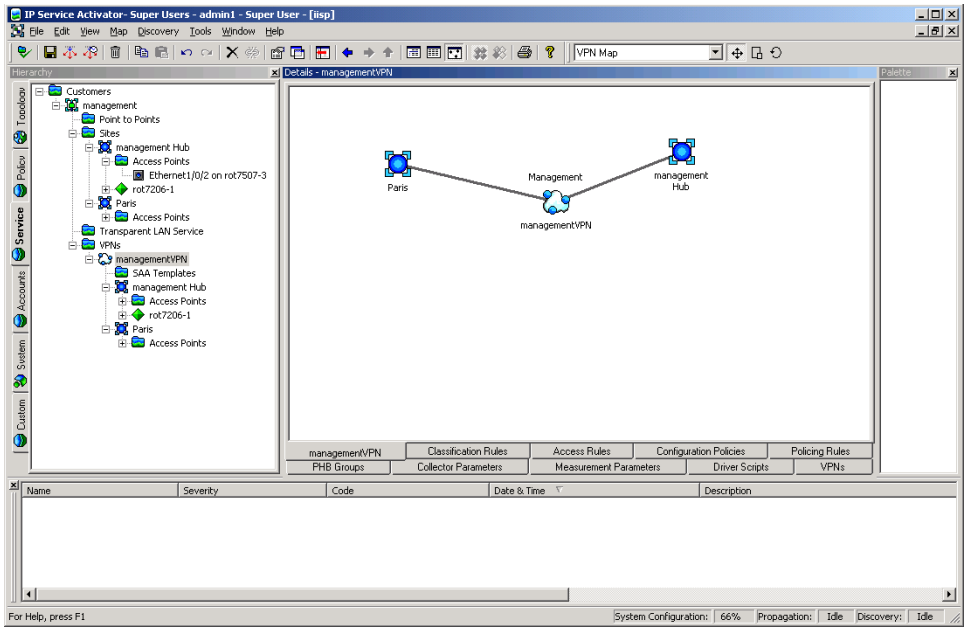
The management site is made a hub of the VPN, by creating a SiteHub object with the name "Management Site" as a child of the VPN.

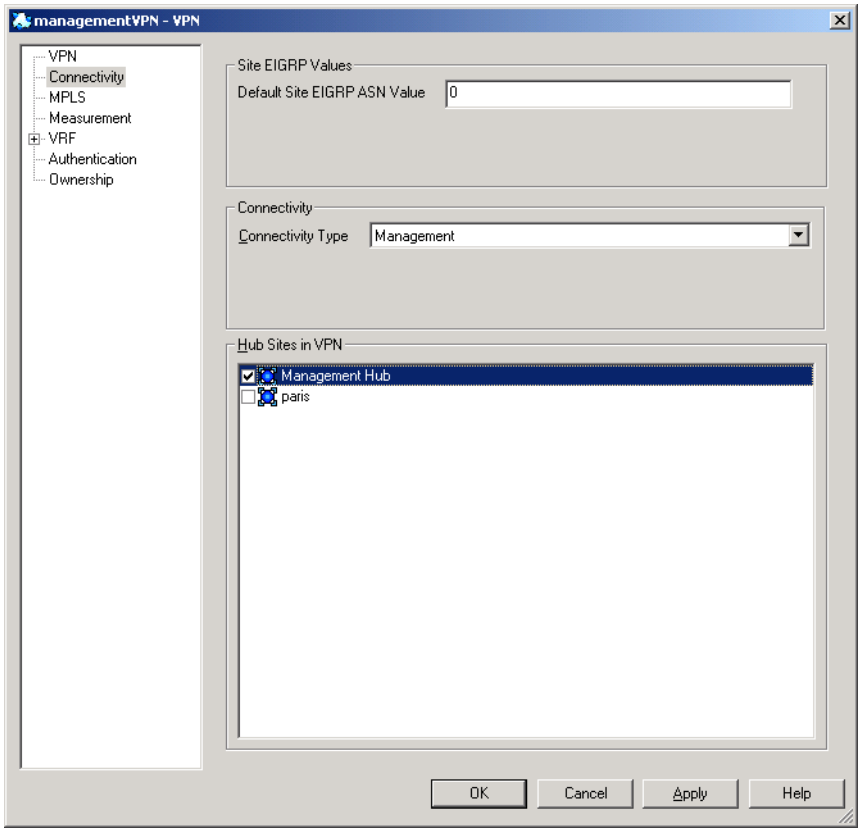
```
setPath VPN:"Management VPN"
create SiteHub:"Management Hub"
commit
```

Finally any customer sites should be linked to the VPN object. In this example a single customer site object (ID=3013) is linked to the VPN object (ID=3020).

```
link [3020] [3013]
commit
```

The following screen shows the management VPN containing the two sites. Also shown is the properties box of the VPN, showing that "Management Site" is a hub, and "Paris" is a spoke.


IP Service Activator- Super Users - admin1 - Super User - [iisp]
File Edit View Map Discovery Tools Window Help
VPN Map
Hierarchy
Details - managementVPN
Palette
Customers
management
Point to Points
Sites
management Hub
Access Points
Ethernet1/0/2 on rot7507-3
rot7206-1
Paris
Access Points
Transparent LAN Service
VPNs
managementVPN
SAA Templates
management Hub
Access Points
rot7206-1
Paris
Access Points
Paris
Management
management Hub
managementVPN
managementVPN
Classification Rules
Access Rules
Configuration Policies
Policing Rules
PHB Groups
Collector Parameters
Measurement Parameters
Driver Scripts
VPNs
Name
Severity
Code
Date & Time
Description
For Help, press F1
System Configuration: 66%
Propagation: Idle
Discovery: Idle


managementVPN - VPN
VPN
Connectivity
MPLS
Measurement
VRF
Authentication
Ownership
Site EIGRP Values
Default Site EIGRP ASN Value
0
Connectivity
Connectivity Type
Management
Hub Sites in VPN
Management Hub
paris
OK
Cancel
Apply
Help

Creating a customer VPN

This section explains the steps involved in creating a customer VPN.

Pre-requisites

- Sites to be included in the VPN must be created and configured as described above.

Command syntax

Create the VPN object

A VPN object must be created as a child of the customer:

```
setPath customer-id
create VPN:"vpn-name"
commit
```

The VPN object must be configured as a either a hub and spoke or a full mesh VPN:

```
modify [vpn-id] MplsVpnType=type
commit
```

Link the customer site to the VPN

Any sites that are to be in the VPN must be linked to the VPN object:

```
link customer-vpn-id customer-site-id
commit
```

Repeat this step for all sites to be included in the VPN.

Set hub sites

If the customer VPN is hub and spoke, one of the linked sites must be defined as the hub site. This is achieved by creating a SiteHub object as a child of the VPN object, with the same name as the site. The *hub-site-name* must match the name of the site under the hub and spoke VPN.

```
setPath customer-vpn-id
create SiteHub:"hub-site-name"
commit
```

If the VPN is full mesh VPN, this step is not necessary.

To remove a SiteHub which is linked to a hub in a hub and spoke VPN:

```
unlink [vpn id] [sitehub id]
delete [sitehub id]
commit together
```

Link customer sites to the management VPN

Any sites linked to any customer VPN must also be linked to the management VPN object, if this has not already been done:

```
link management-vpn-id customer-site-id
commit
```

This should be repeated so all the customer sites are linked to the management VPN object.

Example

This example creates a full mesh VPN containing three customer sites (assumed to be already created, but not yet linked to the management VPN).

OIM commands

First a VPN object is created as a child of Customer 1 (ID=3012).

```
setPath [3012]
create VPN:"Acme Customer VPN"
commit
```

The VPN is defined as full mesh:

```
modify VPN:"Acme Customer VPN" MplsVpnType=FullMesh
commit
```

The three customer sites are linked to the VPN object:

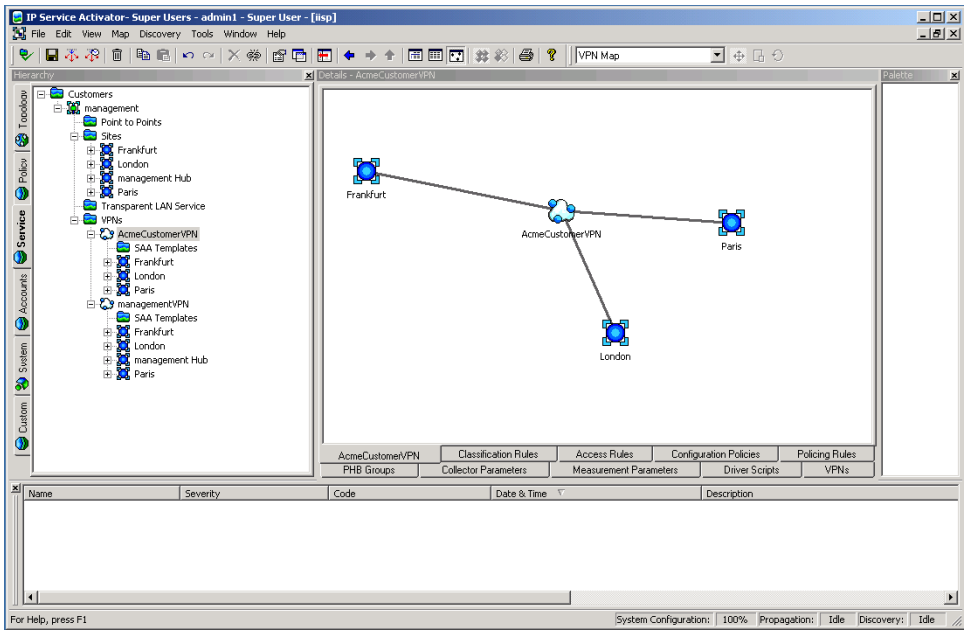
```
link VPN:"Acme Customer VPN" Site:"London"
link VPN:"Acme Customer VPN" Site:"Paris"
link VPN:"Acme Customer VPN" Site:"Frankfurt"
commit
```

The three customer sites are also linked to the management VPN (ID=3084).

```
link [3084] Site:"London"
link [3084] Site:"Paris"
link [3084] Site:"Frankfurt"
```

```
commit
```

The user interface now shows the Customer VPN containing three sites:



Creating and applying rules

This section describes how to create and configure classification rules, access rules and policing rules.

Creating a rule involves the following steps:

- Create the RuleClassification, RuleAccess, or RulePolicing object.
- Link RoleDevice and RoleInterface objects to the Rule to set the role(s) that the rule will apply to.
- Set the attributes of the Rule object.

Once all these steps have been performed, and the final commit executed, then ConcreteRule objects appear as children of any interfaces that match the role criteria.

Pre-requisites

- The devices and interfaces on which the rules are applied must have already been discovered, and must have roles assigned to them.

Command Syntax

Creating the rule

Consider where the rule is to be applied. If it is to apply to all devices and interfaces that match the role(s), the rule should be created as a child of the Domain object. If the rule is only to apply on the interfaces of one specific device, then the rule should be created as a child of that device:

```
setPath domain-id
create RuleType:"rule-name"
commit
setPath device-id
create RuleType:"rule-name"
commit
```

RuleType may be RuleClassification, RuleAccess, or RulePolicing.

Note: The rule applies to global domain only when that domain contains VPN and its applying to only that device which is part of the VPN.

Assigning roles to the rule

To assign roles that the rule will apply to, simply link the appropriate RoleDevice and RoleInterface objects to the rule:

```
link rule-id role-id
commit
```

Rule attributes - inbound/outbound

To specify whether the rule applies inbound, outbound or both, set the InBound and OutBound flags to true or false using the `modify` command:

```
modify rule-id InBound=true
modify rule-id OutBound=true
commit
```

Rule attributes - Date and Time Period

You can set an applicable date and time period in one of two ways.

One method is to modify the StartTime, EndTime, and DaysOfWeek attributes using the `modify` command. StartTime and EndTime must be set to a string with the format "YYYY/MM/DD HH:MM:SS". The DaysOfWeek attribute uses an unsigned integer where each bit represents a day of the week. For example:

```
modify rule-id StartTime="2001/10/10 10:00:00"
modify rule-id EndTime="2001/10/10 10:00:00"
modify rule-id DaysOfWeek="127"
commit
```

The alternative is to modify the PeriodName attribute to match the name of an existing Period object (which must be a child of the Domain object). When this method of setting the data and time is used, any change to the Period object changes the start and end time of the Rule object.

```
modify rule-id PeriodName="period-object-name"
commit
```

Rule attributes - Classification

The Classification of the rule can be set in one of two ways, using local classification or global classification.

To set up local classification, modify the Classification object (which must be a child of the Rule object) to ensure the appropriate source/destination IP address and account name are set. Also modify the TrafficName attribute to the name of an existing Traffic object if required. For example, to set the Rule to work on traffic from IP address 10.0.0.3, to a destination account *dest-account* and traffic type *traffic-name* use the following commands:

```
modify classification-id SourceIpAddr=10.0.0.3
```

```
modify classification-id SourceMask=255.255.255.255
modify classification-id DstAccountName="dest-account"
modify classification-id TrafficName="traffic-name"
commit
```

To set up global classification, first delete the Classification object that is a child of the Rule. Then create a ClassificationGroup object as a child of the Rule. Finally, any number of existing Classification objects or ClassificationGroup objects may be linked to the newly created ClassificationGroup object. For example:

```
setPath rule-id
delete classification-id
create ClassificationGroup:"name"
link classification-group-id classification-object
commit
```

The remaining attributes differ depending on the type of rule.

Classification Rule

To set packet marking on a classification rule, modify the PacketMarking attribute to the name of a PacketMarking object that is a child of the Domain object:

```
modify rule-id PacketMarkingName="packet-marking-name"
commit
```

Creating and linking a packet marking to a traffic type

To create a packet marking and link it to a traffic type:

```
create TrafficPacketMarking:"TrafficTypeTT1"
create PacketMarking:“TrafficPM1”
link [traffic-type-id] [packet-marking-id]
commit
```

Note: the traffic type specified must match the name of the existing traffic type.

To unlink a packet marking from a traffic type:

```
modify [classification-id] TrafficName=""
commit
```

Access Rule

Set the Permit attribute to True to permit the identified traffic, or to False to deny the identified traffic:

```
modify rule-id Permit=false
commit
```

Policing Rule

Set the CommittedRate, NormalBurstSize, and ExcessBurstSize attributes using the `modify` command:

```
modify rule-id CommittedRate=8000
modify rule-id NormalBurstSize=1000
modify rule-id ExcessBurstSize=2000
commit
```

Example

This example creates rules to apply QoS on "Acme Customer VPN" which was set up in an earlier example.

A classification rule is created that marks real-time traffic within Acme Customer VPN is marked with a a specific IP precedence value, and an access rule is created to prevent any Game traffic (Quake) from traveling on the VPN.

Classification Rule

A RuleClassification object is created as a child of the VPN (ID= 3105):

```
setPath [3105]
create RuleClassification:"Real-time"
commit
```

The device and interface roles are set up to indicate which interfaces the rule will apply to. In this case the rule is applied to the Access interfaces on the PE. Therefore the RuleClassification object is linked to the Gateway device role (ID=5) and the Access interface role (ID=11):

```
link RuleClassification:"Real-time" [5]
link RuleClassification:"Real-time" [11]
commit
```

The rule is to apply to inbound traffic on the interfaces, so the Rule's InBound attribute is set to True, and the OutBound attribute is set to False.

```
modify RuleClassification:"Real-time" InBound=True OutBound=False
commit
```

Next the packet marking is set. The PacketMarkingName attribute is set to "IP Prec 5", which is the name of the PacketMarking object that has been assigned to the identified traffic.

```
modify RuleClassification:"Real-time" PacketMarkingName="IP Prec 5"
commit
```

Concrete rule objects will now appear as children of the three PE access interfaces in the VPN.

Access Rule

An access rule is set up to prevent Quake traffic from using the VPN. This rule will be configured to apply to the PE Access interfaces in the VPN.

A RuleAccess object is created as a child of the customer VPN (ID=3105):

```
setPath [3105]
create RuleAccess:"Quake Deny"
commit
```

Now the RuleAccess object is linked to the Gateway device role (ID=5) and the Access interface role (ID=11).

```
link RuleAccess:"Quake Deny" [5]
link RuleAccess:"Quake Deny" [11]
commit
```

The rule is to apply to inbound traffic on the interfaces, so the rule's InBound attribute is set to True, and the OutBound attribute is set to False.

```
modify RuleAccess:"Quake Deny" InBound=True OutBound=False
commit
```

As the access rule will deny traffic, the Permit attribute should be set to False:

```
modify RuleAccess:"Quake Deny" Permit=False
commit
```

Finally, the traffic type which identifies the traffic to be denied is set. To do this the TrafficName attribute of the Classification object that is a child of the RuleAccess object is set to the name of the traffic to be denied:

```
modify RuleAccess:"Quake Deny"/Classification:""
TrafficName="quake-c"
commit
```

Creating and applying PHB groups

Creating and apply PHB groups involves the following steps:

- Create a PHBGroup object
- Modify the PhbGroup object's PHB child object

  The OIM automatically creates a PHB child object when you create a PHBGroup object.
- Create a PHB object for each class of service to be handled by the PHBGroup object

Pre-requisites

- Any Cos object to be used by an MQC PHB group must be linked to a Classification object

Creating a standard PHB group

A PHB group must always be created as a child of the Domain object. When a PHBGroup object is created, a PHB child object is automatically created and its ClassName attribute set to 'Default Class of Service'. The Name attribute is automatically taken from the ClassName attribute – the two are inextricably linked.

```
setpath /Policy:"Policy"/Domain:"USA"
create PHBGroup:"WRRPHB"
modify PHBGroup:"WRRPHB" Action=8
commit
```

The device and interface roles are set up to indicate which interfaces the PHB group will apply to. In this case the PHB group is applied to the Access interfaces on the PE. Therefore the PHBGroup object is linked to the Gateway device role (ID=5) and the Access interface role (ID=11):

```
link PHBGroup:"WRRPHB" [5]
link PHBGroup:"WRRPHB" [11]
commit
```

PHBAtm and PHBFrts objects are also automatically created as children of the PHBGroup object:

```
setpath PHBGroup:"WRRPHB"
getchildren
    [1197] PhbAtm:"PhbAtm"
    [1196] PhbFrts:"PhbFrts"
```

```
    [1190] Phb:"Default Class of Service"
```

Creating additional PHB children

You can create additional PHB objects as children of the PHBGroup object. You must set the ClassName attribute at creation:

```
setpath /Policy:"Policy"/Domain:"USA"/PHBGroup:"WRRPHB"
create PHB:"name" ClassName="Silver"
commit
modify PHB:"Silver" ClassName="Gold"
commit
```

Note that the value assigned to the PHB object's `name` attribute is ignored by the OIM. The OIM automatically assigns a name to the PHB object, using the value specified in the ClassName attribute – that is, the PHB object's name is taken from the CoS to which it is linked.

If no ClassName is specified at creation, the attribute is automatically assigned the string 'Default Class of Service'. Assigning the same ClassName value to two PHB objects results in an error:

`CommandExecutionError.ERR_ObjectModelViolation: Object Cos is linked to more than one Phb under the same PHBGroup [`*`PhbGroupId`*`].`

Use the `getchildren` command to view the children of the PHBGroup object 'WRRPHB':

```
/Policy:"Policy"/Domain:"USA"/PHBGroup:"WRRPHB">getchildren
    [1197] PhbAtm:"PhbAtm"
    [1196] PhbFrts:"PhbFrts"
    [1190] Phb:"Default Class of Service"
    [1205] Phb:"Gold"
```

Creating an MQC PHB group

An MQC PHB group must be created as a child of the Domain object, in the same way as a standard PHB group.

When a PHBGroupMqc object is created, the OIM automatically creates a PhbMqc child object. The child object's ClassName attribute is set to 'Default Class of Service', which associates the object with this CoS. It is essential to modify the CosName attribute and specify the QoS action:

```
setpath /Policy:"Policy"/Domain:"USA"
create PHBGroupMqc:"CBWFQMQC"
```

```
modify PHBGroupMqc:"CBWFQMQC"/PhbMqc:"Default Class of Service" …
CosName="Gold" Action=2
commit
```

Refer to *PHBGroupMqc objects* on page 190 for a list of applicable Action types.

Note that if a value is not specified for the Action attribute the following error is generated:

```
CommandExecutionError.ERR_ObjectModelViolation: [3653], No MQC Action
has been selected.
```

The device and interface roles are set up to indicate which interfaces the MQC PHB group will apply to. In this case the MQC PHB group is applied to the Access interfaces on the PE. Therefore the PHBGroupMqc object is linked to the Gateway device role (ID=5) and the Access interface role (ID=11):

```
link PHBGroupMqc:"CBWFQMQC" [5]
link PHBGroupMqc:"CBWFQMQC" [11]
commit
```

Creating additional PHBMqc children

You can create additional PHBMqc objects as children of the PHBGroupMqc object. You must set the CosName attribute at creation:

```
setpath /Policy:"Policy"/Domain:"Test"/PhbGroupMqc:"CBWFQMQC">
create PHBMqc:"PHBMQCSilver" CosName="Silver" Action=16
commit
```

If no CosName is specified at creation, the attribute is automatically assigned the string ‘Default Class of Service’. Assigning the same CosName value to two PHBMqc objects results in an error.

Implementing single or two-rate policing with a PHBMqc object

Implementing single or two-rate policing with a PHBMqc object involves the following attributes:

- Action – set to 4 for single-rate policing, 8 for two-rate policing
- ConformAction – specify the name of a PHBPolicingAction object
- ExceedAction – specify the name of a PHBPolicingAction object
- ViolateAction – specify the name of a PHBPolicingAction object

For example, to implement two-rate policing:

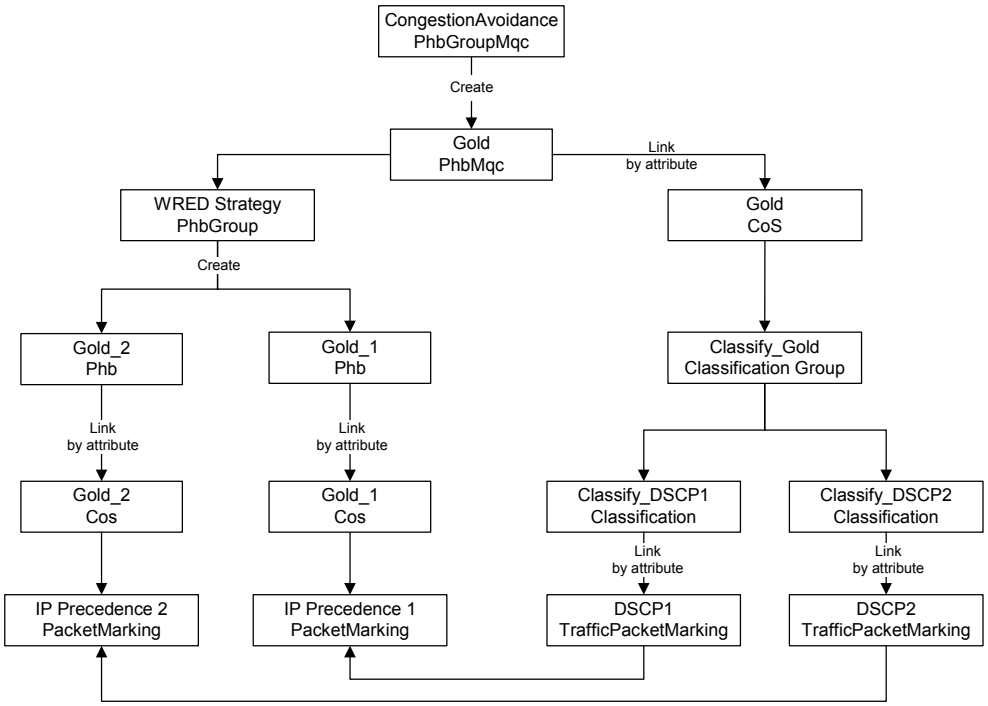
```
modify PhbMqc:"Default Class of Service" Action=8
```

```
ConformAction="Default Conform Action"
ExceedAction="Default Exceed Action"
ViolateAction="Default Violate Action"
```

For a list of applicable Action values, refer to *PHBGroup objects* on page 179.

Configuring WFQ using WRED as the drop strategy

It is possible to configure WFQ using WRED as the drop strategy. The following diagram illustrates the object relationships that must be created to configure WFQ+WRED.



Pre-requisites

The example assumes the following:

- PacketMarking objects IP Precedence 1 and IP Precedence 2 have been created through the user interface

  It is also possible to perform this step via the OIM. However, these packet markings are created automatically by loading the **default.dscp.policy** file into Service Activator.

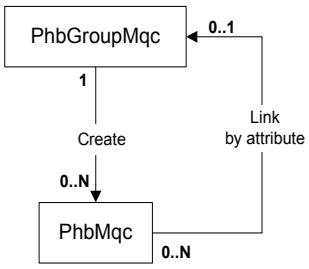
Sample code

```
*** The PhbMqc requires that any Cos it applies to is linked to a
*** Classification so we must create an appropriate Classification. In
*** this case it will be a ClassificationGroup containing
*** TrafficPacketMarkings DSCP1 and DSCP2.
create TrafficPacketMarking:"DSCP1"
create TrafficPacketMarking:"DSCP2"
link TrafficPacketMarking:"DSCP1" PacketMarking:"IP Precedence 1"
link TrafficPacketMarking:"DSCP2" PacketMarking:"IP Precedence 2"

create Classification:"Classify_DSCP1" TrafficName="DSCP1"
create Classification:"Classify_DSCP2" TrafficName="DSCP2"

create ClassificationGroup:"Classify_Gold"

link ClassificationGroup:"Classify_Gold"
Classification:"Classify_DSCP1"
link ClassificationGroup:"Classify_Gold"
Classification:"Classify_DSCP2"

create Cos:"Gold"
link Cos:"Gold" ClassificationGroup:"Classify_Gold"



*** Create the 2 Cos used in the WRED Phbs that will determine the WRED
*** strategy.

create Cos:"Gold_1"
create Cos:"Gold_2"

link Cos:"Gold_1" PacketMarking:"IP Precedence 1"
link Cos:"Gold_2" PacketMarking:"IP Precedence 2"

*** Create the WRED PhbGroup
```

```
create PhbGroup:"WRED Strategy" Action=8

create PhbGroup:"WRED Strategy"/Phb:"Gold_1" ClassName="Gold_1"
create PhbGroup:"WRED Strategy"/Phb:"Gold_2" ClassName="Gold_2"

*** Commit the transaction
commit

*** These are the children of the WRED PhbGroup.
*** One can modify the attributes for the Phbs as required (or could set
*** them during creation).

/Policy:"Policy"/Domain:"Domain1"/PHBGroup:"WRED Strategy">getchildren
    [1583] PhbAtm:"PhbAtm"
    [1582] PhbFrts:"PhbFrts"
    [1576] Phb:"Default Class of Service"
    [1590] Phb:"Gold_2"
    [1584] Phb:"Gold_1"

*** Create the PhbGroupMqc (Action = 66 = 2 + 64 = CBWFQ + Congestion)

setpath /Policy:"Policy"/Domain:"Domain1"
create PhbGroupMqc:"Congestion Avoidance"
modify PhbGroupMqc:"Congestion Avoidance"/PhbMqc:"Default Class of
Service" CosName="Gold" Action=66
modify PhbGroupMqc:"Congestion Avoidance"/PhbMqc:"Gold"
WredStrategy=WRED
link PhbGroupMqc:"Congestion Avoidance"/PhbMqc:"Gold" PHBGroup:"WRED
Strategy"

commit
```

Nesting MQC PHB groups

A PHBGroupMqc object that specifies CBWFQ, Shaping or Policing may nest a child PHBGroupMqc and so configure what Cisco refers to as a ‘hierarchical service policy’. The following diagram illustrates the object relationships that must be formed to nest MQC PHB groups.



Pre-requisites

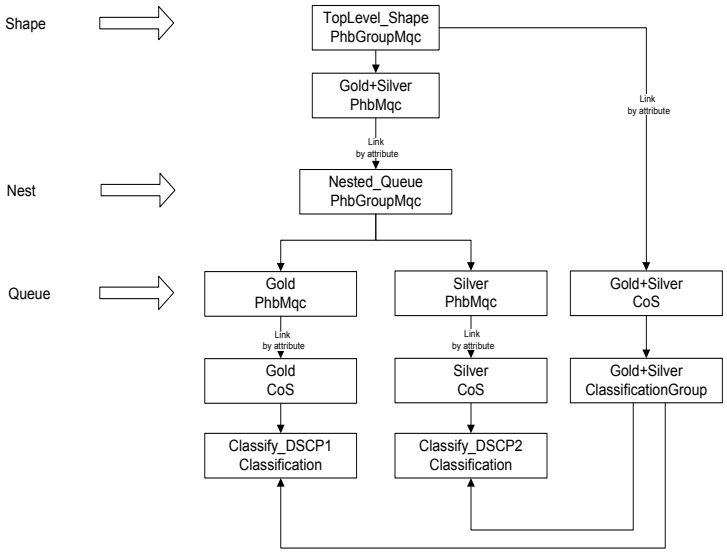
The example assumes the following:

- PacketMarking objects IP Precedence 1 and IP Precedence 2 have been created through the user interface

  It is also possible to perform this step via the OIM. However, these packet markings are created automatically by loading the **default.dscp.policy** file into Service Activator.

Overview of the example

This example uses nested MQC PHB groups to apply traffic shaping to incoming Gold and Silver traffic and different queuing treatments to each class of service individually. The object relationships that are created and the policies that are applied are shown in the following diagram.



Sample code

```
*** The PhbMqc requires that the Cos is linked to a Classification so we
*** must create an appropriate classification. To apply shaping to
*** Gold and Silver traffic we need to create a ClassificationGroup
*** containing 2 types of traffic - DSCP1 and DSCP2.

create TrafficPacketMarking:"DSCP1"
create TrafficPacketMarking:"DSCP2"
link TrafficPacketMarking:"DSCP1" PacketMarking:"IP Precedence 1"
link TrafficPacketMarking:"DSCP2" PacketMarking:"IP Precedence 2"

create Classification:"Classify_DSCP1" TrafficName="DSCP1"
create Classification:"Classify_DSCP2" TrafficName="DSCP2"

create ClassificationGroup:"Gold+Silver"
```

```
link ClassificationGroup:"Gold+Silver" Classification:"Classify_DSCP1"
link ClassificationGroup:"Gold+Silver" Classification:"Classify_DSCP2"

create Cos:"Gold+Silver"
link Cos:"Gold+Silver" ClassificationGroup:"Gold+Silver"


*** We also need to create a Cos for each traffic type we
*** want to apply individual queuing parameters to.
create Cos:"Gold"
create Cos:"Silver"
link Cos:"Gold" Classification:"Classify_DSCP1"
link Cos:"Silver" Classification:"Classify_DSCP2"

*** Create a PhbGroupMqc specifying CB-WFQ as the treatment. This
*** PhbGroupMqc will be nested in a parent PhbGroupMqc that defines
*** traffic shaping parameters for the Cos's.

create PhbGroupMqc:"Nested_Queue"
modify PhbGroupMqc:"Nested_Queue" WfqBandwidthType=Percentage

*** We need to create 2 PhbMqcs that define queuing parameters for
*** each Cos - PhbMqc:"Gold" (renames the automatically created
*** "Default Class of Service" PhbMqc) and PhbMqc:"Silver"
*** Action=2 (CBWFQ) and we allocate different bandwidth for each
*** stream (35% and 20%)

modify PhbGroupMqc:"Nested_Queue"/PhbMqc:"Default Class of Service"
CosName="Gold" Action=2 WFQWeight=35
create PhbGroupMqc:"Nested_Queue"/PhbMqc:"Silver" CosName="Silver"
Action=2 WFQWeight=20

*** Create the top-level PhbGroupMqc to shape both Cos's
*** (Gold+Silver) and select the nested PhbGroupMqc (linked by
*** attribute). Action= 144 (Shape + Nest)
```

```
create PhbGroupMqc:"TopLevel_Shape"
modify PhbGroupMqc:"TopLevel_Shape"/PhbMqc:"Default Class of Service"
CosName="Gold+Silver" Action=144 NestedPhbGroup="Nested_Queue"

commit
```

Creating an event subscription

This section describes how to create an event subscription that will collect events of a specified type occurring on specified objects.

Pre-requisites

- The objects on which events are to be collected must already exist and have been configured.

Command Syntax

Creating a subscription requires two steps:

1. An EventSubscription object must be created as a child of the System object. The EventSubscription specifies the delivery target.
2. One or more EventCollector objects must be created as a child of the EventSubscription. The EventCollector objects determine which type of events should be captured, and on which object(s).

Creating the Event Subscription

First the EventSubscription object is created as a child of the System object:

```
setPath /System:"System"
create EventSubscription:"name"
commit
```

Then the attributes of the EventSubscription must be set, namely the DeliveryType and the DeliveryDetails. The DeliveryType attribute indicates what sort of trap (if any) should be sent, and the DeliveryDetails attribute tells where to deliver the trap.

```
modify event-subscription-id DeliveryType=delivery-type
DeliveryDetails=delivery-details
commit
```

delivery-type can take the values SnmpTrap, UpgradedSnmpTrap, NetCool, CORBA Channel, or DatabaseOnly.

delivery-details is a string containing delivery details. For SNMP traps this is a comma-separated list specifying the IP address, port to which the traps are sent, and SNMP version (for example, "192.168.1.2,162,2"). For CORBA or NetCool, the string indicates the name of the service. If DeliveryType is DatabaseOnly the string should be blank.

Creating the Event Collector

Multiple EventCollector objects may be created as children of the EventSubscription object. Each EventCollector object traps one particular type of event on a group of objects. Therefore if, for example, you need to trap attribute changes and fault events occurring on a single object, two EventCollector objects would be required.

```
setpath /System:"System"/EventSubscription:"name"
create EventCollector:"name"
commit
```

After creation, set the type of event to be reported:

```
modify event-subscription-id Type=type
commit
```

type takes the values CreateAndDelete, Fault, AttributeChange, or StateChange.

Then set the object(s) to trap on. There are several ways to do this.

To collect events from a single object only, link that object to the EventCollector, and set the attribute HierarchicalCollection to False:

```
link event-collector-id target-object-id
modify event-collector-id HierarchicalCollection=False
commit
```

To collect events from an object and all items below it in the hierarchy, link the top object to the EventCollector object and set the HierarchicalCollection attribute to True.

```
link event-collector-id target-object-id
modify event-collector-id HierarchicalCollection=True
commit
```

If any event from any object in the entire EOM is required, simply set the attributes RootCollection and HierarchicalCollection to True.

```
modify event-collector-id RootCollection=True
HierarchicalCollection=True
commit
```

Note that you cannot set the RootCollection to be True while an object linked to the EventCollector object or CollectionPoint is non-zero.

The scope can be further narrowed down by restricting it by device role and/or interface role. To do this, simply link any DeviceRole and/or InterfaceRole object to the EventCollector.

```
link event-collector-id [role-id]
commit
```

Once the subscription is set up, any events occurring that match one of the EventCollector's criteria will be sent as specified in the EventSubscription's DeliveryDetails attribute.

Event Filters

EventFilter objects can be created as children of EventCollector objects, in order to filter on faults and/or object attributes. You do not need to set the name of an EventFilter object as it will be named automatically according to the attributes set.

The two types of EventFilter objects, EventFilterFaultMask and EventFilterAttributeChange, work in similar ways.

EventFilterFaultMask

The EventFilterFaultMask allows you to set up a filter to report on specific categories of faults or individual faults only. When you create the EventFaultFilterMask object you must set the Permit attribute to True or False and the FaultCategory attribute to to match the type of fault to be reported. To report a single fault only, set the FaultCategory to "SingleFaultCode", and set the FaultCode attribute to the relevant fault number (see *EventFilterFaultMask object* on page 311).

```
setPath event-collector-id
create EventFilterFaultMask:"" Permit={true|false}
FaultCategory=fault-category FaultCode=fault-code
commit
```

EventFilterAttributeChange

The EventFilterFaultMask allows you to set up a filter to report on a specific attribute change. You must set the Permit attribute to True or False, and the AttributeName attribute to the name of the attribute to be reported. the attribute can be any attribute of any object within the EOM.

```
setPath event-collector-id
create EventFilterAttributeChange:"" Permit={true|false}
AttributeName=attribute-name
commit
```

Example

In this example, an EventSubscription object will be created and configured to send Upgraded SNMP Traps to an IP address. Two EventCollector objects will be created

as children of the EventSubscription. One will catch all faults; the other will catch attribute changes of any objects in the "Customer One" hierarchy.

It assumes that initially no EventSubscription objects exist.

EventSubscription

The EventSubscription object is created as a child of /System:"System":

```
setPath /System:"System"
create EventSubscription:"Subscription 1"
commit
```

The DeliveryType attribute is set to UpgradedSNMPTrap, and the DeliveryDetails should be set to "10.100.100.10, 162, 1" which signifies that the traps should be sent to 10.100.100.10, on port 162, and version 1 SNMP traps should be sent:

```
modify EventSubscription:"Subscription 1"
DeliveryType=UpgradedSnmpTrap DeliveryDetails="10.100.100.10,162,1"
commit
```

EventCollector

The EventCollector objects are created as children of the EventSubscription object:

```
setPath EventSubscription:"Subscription 1"
create EventCollector:"Fault Collector"
create EventCollector:"Attribute Changes Collector"
commit
```

Firstly, configure the event collector that collects faults, which collects all faults in the system:

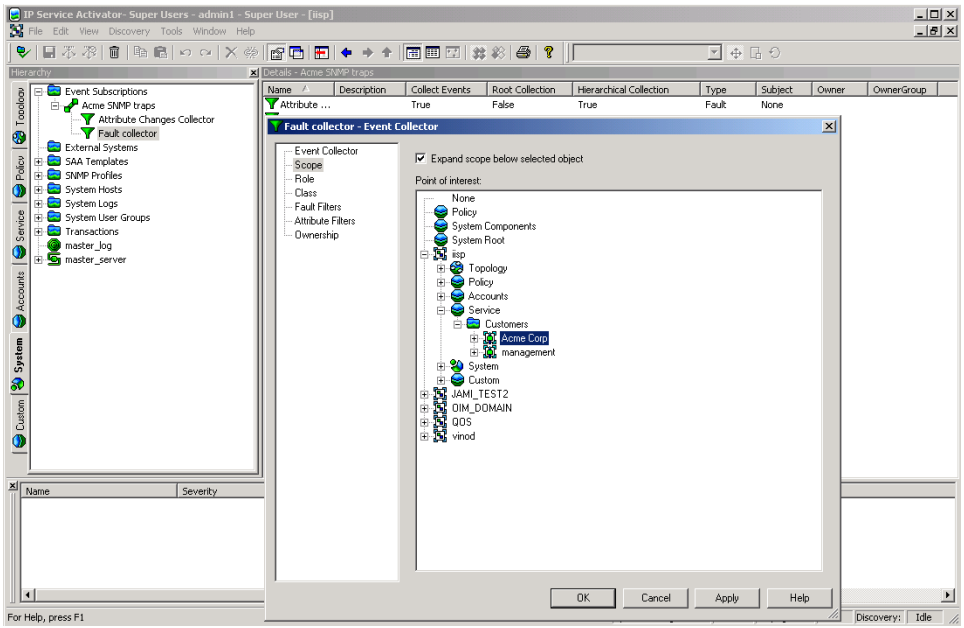
```
modify EventCollector:"Fault Collector" Type=Fault
modify EventCollector:"Fault Collector" RootCollection=True
modify EventCollector:"Fault Collector" HierarchicalCollection=True
commit
```

Then configure an event collector to collect attribute changes, which collects events for the ACME Corp customer (ID=3012) and its children:

```
modify EventCollector:"Attribute Changes Collector"
Classname=<Class-name>
modify EventCollector:"Attribute Changes Collector"
Type=AttributeChange
```

```
modify EventCollector:"Attribute Changes Collector"
HierarchicalCollection=True
link EventCollector:"Attribute Changes Collector" [3012]
commit
```

The following screen shows that an EventSubscription object and two EventCollector objects have been created. The dialog box shows that the "Attribute Changes" EventCollector object's scope is defined as the Acme Corp customer.



Applying parameter sets

This example describes how to apply parameter sets to configuration targets.

Pre-requisites

- Devices, interfaces, sub-interfaces, and VC endpoints must have already been discovered, and configured with roles assigned to them.

Command syntax

ParameterSets are applied using the `use` command. The syntax is:

```
use object-id parameter-set-id [attribute-list]
```

The above command will create a ParameterSetInstance object that is a child of both the object and the parameter set specified. The instance will also be inherited by all applicable objects, but the ParameterSetInstance object will only appear as a child of the specified object the ParameterSet object has been "used" with.

The attributes of the ParameterSetInstance to be created can be set by supplying optional attributes, or the ParameterSetInstance object can be modified after creation.

If a CollectorParameterSet is being applied, then the relevant ExternalSystem object should be linked to the created ParameterSetInstance:

```
link parameter-set-instance-id external-system-id
```

To remove the ParameterSetInstance, delete the ParameterSetInstance object:

```
delete object-id parameter-set-id
commit
```

Example

In this example, a Measurement ParameterSetInstance object (ID=15) will be applied to a Network object (ID=25) with one device. The example shows how the parameters are applied to all the interfaces belonging to the device:

```
use [25][15] Name="MeasurementInstance 1" Levels="4"
OCH_MeasureCBQoS="True"
commit
```

The `use` command sets the Levels attribute of the ParameterSetInstance to 4. Levels is a bitwise attribute, and 4 means Interface, so the parameters will only be applied to interfaces.

The ParameterSetInstance object can be seen as a child of the Network object:

```
getchildren[25]
[701] Device:"A2504-2"
[1400] ParameterSetInstance:"Measurement Instance 1"
```

Device roles and interface roles can then be linked to the ParameterSetInstance object to indicate which interfaces the parameters should be applied to. For this example a device role of “Any”, and an interface role of "Any" are linked to the ParameterSetInstance:

```
link [1400] /Policy:"Policy"/RoleDevice:"Any"
link [1400] /Policy:"Policy"/RoleInterface:"Any"
```

Using the `getParameters` command on one of the interfaces that is a child of the device (ID=701) will list the parameters from the ParameterSetInstance:

```
getchildren [701]
[702]Interface:"Ethernet0"
[705]Interface:"Serial0"
[714]Interface:"Serial1"
[715]Interface:"Null0"
[716]Interface:"Loopback0"
[740]RuleClassification:"ca1"
[5]RoleDevice:"Gateway"
getparameters [702]
r-- BooleanOCH_MeasureCBQoS = True
r-- BooleanOCH_MeasureCarQoSMIB=False
r-- BooleanOCH_MeasureMIB2Stats=False
r-- BooleanOCH_NetflowEnabled=False
r-- U32OCH_NetflowVersion=1
r-- U32OCH_NetflowAggregation=0
r-- U32OCH_NetflowCacheSize=0
r-- U32OCH_NetflowTimeoutActive=0
r-- U32OCH_NetflowTimeoutInactive=0
```

Using the `getTargets` command on the ParameterSetInstance will display all the interface children of the device, as long as they have a role assigned.

```
gettargets [1400]
[702]Interface:"Ethernet0"
```

```
[705]Interface:"Serial0"
[714]Interface:"Serial1"
[715]Interface:"Null0"
[716]Interface:"Loopback0"
[740]RuleClassification:"ca1"
```

Applying a Collector ParameterSet is similar, except that ExternalSystem object should be linked to the ParameterSetInstance object after creation to provide the values for the parameters.

Provisioning SAA

This section describes how to create an SAA template with a Jitter operation and apply it to a VPN.

Creating an SAA template involves the following steps:

- Create an SAATemplate object
- Create the required SAAOperation objects as children of the template
- Link the SAA template object to an existing VPN

Creating an SAA Template

An SAATemplate must be created as a child of the Policy object:

```
setPath /Policy:Policy
create SAATemplate:"JitterTemplate" DeviceBits=9 TosBits=3
TypeBits=2
commit
```

Note that the combined values of the DeviceBits, TosBits and TypeBits attributes must be exactly 14. For more information, see *SAATemplate object* on page 107.

Creating an SAA Operation

Only one SAA operation per operation type can be created as a child of an SAATemplate object.

The name of an SAAOperation object is derived from its Type attribute value. When creating an SAAOperation object, the Name attribute must be left as an empty string ("") and the Type attribute must be specified:

```
setpath /Policy:Policy/SAATemplate:JitterTemplate
create SAAOperation:"" Type=TcpConnect
modify SAAOperation:"TcpConnect" Period=120 Timeout=25
modify SAAOperation:"TcpConnect" EnableErrorChecking=True
commit
```

Applying an SAA template to a VPN

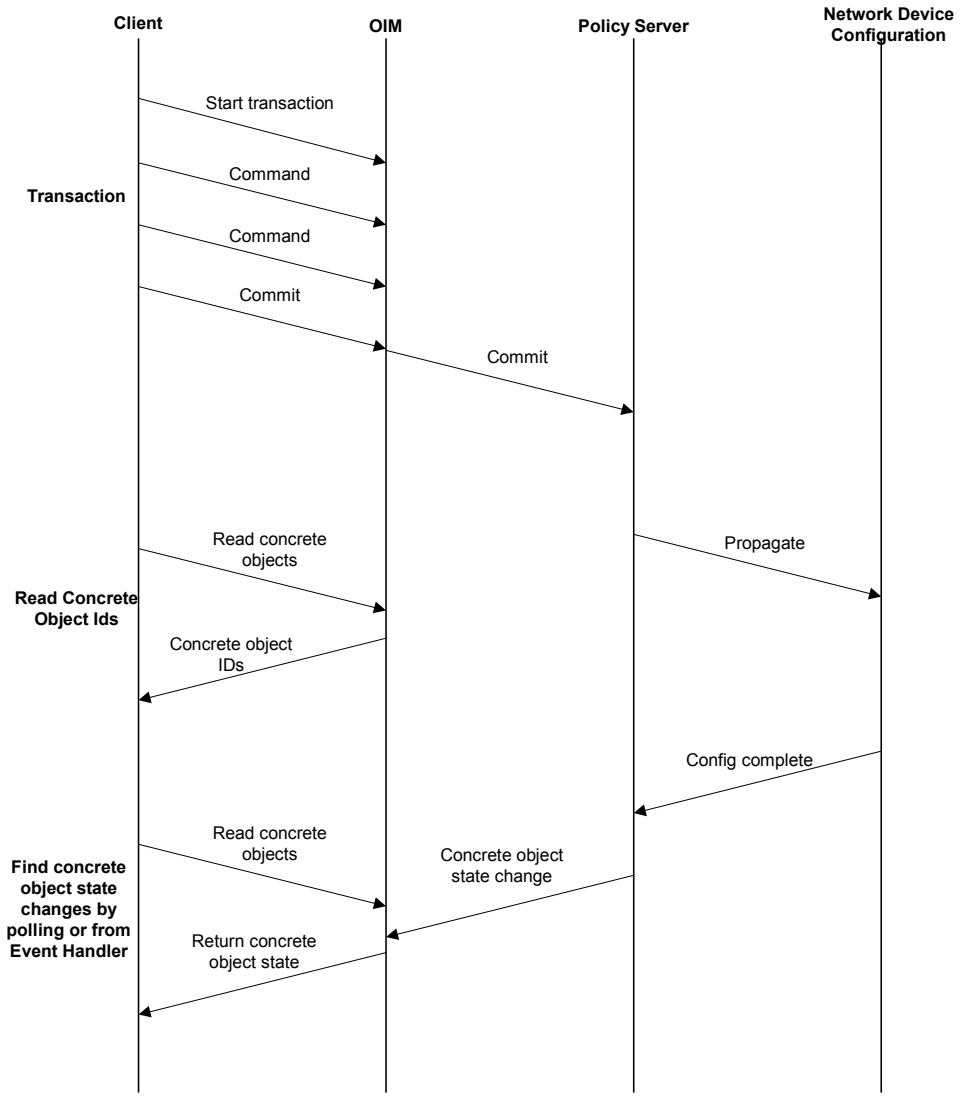
An SAA template is applied by linking it as a child to a VPN:

```
setpath /Policy:Policy
link Domain:"Global"/Customer:"Acme"/Vpn:"Acme VPN"
SAATemplate:"JitterTemplate”
commit
```

An SAATemplate object may be linked to multiple VPNs.

Determining when transactions have been completed

It is useful to be able to determine when a transaction has been completed and to find out if configuration has been successfully applied to a device. The following diagram shows the workflow of a transaction to update device configuration:



When a transaction is committed, the policy server propagates the new configuration and the device drivers attempt to apply it to the devices. A concrete

object is created for each implementation of the rule, PHB group, VPN or Driver Script to the network. For example, if a rule is applied, a separate concrete rule is created on each interface where it is to be implemented.

To find out whether the configuration has been successfully installed you need to check the State attribute of all the concrete objects involved in the transaction. The State attribute is initially "Inactive" and is updated when the device is configured:

- A state of "Installed" means that the transaction is complete and the configuration has been successfully propagated to the device.
- A state of "Failed" means the transaction is complete and the configuration could not be applied.
- Other states indicate that the transaction is still being processed. Note that states "Finished", "RunFailed", and "RunOnceFailed" only apply to DriverScripts.

In order to determine whether configuration has been applied, you therefore need to do the following:

1. Locate the relevant concrete objects
2. Check for changes in status

Finding relevant concrete objects

To find out the instances where an object has been applied, you can find all the concrete objects descended from the parent object (rule, VPN, etc), and all the concrete objects descended from the object the rule is applied to (network, customer, etc). This can be done using the command:

```
find objectID "ConcreteObject:*"
```

The intersection of the two sequences will give the set of concrete objects to examine. See *The find command* on page 47.

An alternative method is to use the `events` command on the transaction that applied the parent object. This will give a list of operations performed in the transaction, including creating, linking, or modifying concrete objects. This can be used with the `subscribe` command to hide operations on other objects. See *The events command* on page 21 and *The subscribe/unsubscribe commands* on page 36.

Checking concrete object status

There are two alternative methods for checking the concrete objects:

- Poll the concrete objects, monitoring the State attributes until the value is either "Installed" or "Failed".
- Set up an event subscription to collect AttributeChange events on the relevant concrete objects. See *Creating an event subscription* on page 363 for more details

Example

The following pseudo-code illustrates a script for a rule being applied to a VPN. Similar processing could be used for other scenarios:

```
link vpnID ruleID;commit
vpnConcretes = find vpnID "ConcreteObject:*"
ruleConcretes = find ruleID "ConcreteObject:*"
activeConcrete = intersection(vpnConcretes, ruleConcretes)
done = false
While not done:
   Processing = 0
   Installed = 0
   Failed = 0
   For concreteObj in activeConcretes:
      getAttributes concreteObj
      get State attribute from the returned list
         If State == "Installed":


            Installed++
         Else if State == "Failed":
            Failed++
         Else:
            Processing++
         If Processing != 0:
            done = true
         Do something with the Installed & Failed counts...
```

Managing system users

The commands in this example illustrate how to view all users and user groups, how to add, delete, or modify a user, and how to change a user's password.

SystemUserGroups cannot be created using the OSS Integration Manager (OIM). You must use the graphical user interface to create a new SystemUserGroup.

Viewing all users and user groups

Super User access is required to view all users and user groups.

1. Log in to the OSS Integration Manager (OIM) using either the OIM CLI or a python OIM script.
   For example: `run integration_manager_cli`
2. To view all user groups and their IDs, enter:

```
>find / SystemUserGroup:"*"
  [30] SystemUserGroup:"Super Users"
```

where 30 is the ID of the Super Users group

To view all users and their IDs, enter:

```
>find / SystemUser:"*"
  [33] SystemUser:"admin"
  [2249] SystemUser:"jlastname"
```

where 33 is the ID of user admin

Adding a user

Super User access is required to add a user.

1. Log in to OIM.
2. Find the ID of the desired SystemUserGroup.
3. Create the new user in the desired SystemUserGroup:

```
>setpath <[ID]>
>create SystemUser:"<alastname>" password="<password>"
>commit
```

where ID is the desired SystemUserGroup ID

Modifying a user

Users can change their own password and view their own attributes. The Super User can change any user's password, and view the attributes of any user.

To change a user's password:

1. Log in to OIM.

2. Find the ID of the desired SystemUser: `>find / SystemUser:"jl*"`
3. Modify the password:

   `>modify [ID] password="password"` where ID is the desired SystemUser ID

   `>commit`

To view user attributes:

4. View the desired user's attributes:

   `>getattributes [2249]` where ID=2249 is the desired SystemUser ID

```
r--      U32       Active            = 0
rcw      Boolean   Concurrent        = False
rcw      Boolean   Enabled           = True
r--      Date      ExpireDate        = 2004/06/16 04:59:59
r--      U32       FailedLogins      = 0
rcw      Secure    Name              = jlastname
rcw      String    Password          = "*****"
rcw      U32       PasswordExpires   = 0
rcw      String    Remarks           = ""
rcw      Boolean   ResetPassword     = False
r--      U32       Id                = 2249
```

Explanation: `r--` indicates a read-only attribute, while `rcw` indicates a modifiable attribute. For additional explanation of attributes, refer to page 320.

Deleting a user

Super User access is required to delete a user.

1. Log in to OIM.
2. Find the ID of the desired SystemUser.
3. Delete the user:

   `>delete [2249]` where ID=2249 is the desired SystemUser ID

   `>commit`

```
    [2256] TransactionEntry:"20040528164422"
```

Configuration Thresholding feature - modifying the regular expression

When applying Configuration Thresholding to pushing configuration commands to cartridge-driven devices, you can use this procedure to help you specify the regular expression. The provisioning of the regular expression cannot be done through the GUI, only through the OSS Integration Manager.

The Configuration Thresholding feature checks the commands to be pushed against a provisioned regular expression (**MatchesPatternTransactionSize**), counts the number of matches, and verifies that the count does not exceed the value of **MaxTransactionSize**. If the limit is exceeded, the commands are not sent and errors are raised against the concretes and the device. The regular expression and the count limit can be provisioned at a network or a device level. Configuration Thresholding is applied per device, per device transaction. For more information about this feature, refer to the *Network Discovery & Basic Setup* guide.

A different expression is used for different vendor devices to achieve the same purpose:

Cisco regular expression

For Cisco, you can count, for example, the number of "no" occurrences at the beginning of commands, to limit the amount of configuration removed in one transaction.

Huawei regular expression

For Huawei cartridge units, you can achieve the same function by counting the number of "undo" occurrences at the beginning of commands.

Juniper regular expression

For Juniper, the whole XML text (which is one large command) is considered, and the system counts the number of times that the regular expression is found in it. On JuniperXml, the system counts occurrences of "delete="delete"" attributes anywhere in the XML text.

In most cases, it gives the same results as for other vendor cartridges, except if you use "greedy" regular expressions. A "greedy" expression on Juniper can potentially match a big part of the XML text (several or many commands). On other cartridges, it only matches to a whole command at most for each count.

Limitation

A general limitation of the feature is that counting occurrences of "no" or "undo" or "delete="delete"" may not give an accurate idea of how much configuration will be removed, because if the match occurs at the top of a large code hierarchy, all code under it would be removed.

In the following sample of device code, if the command to be pushed is "no Interface 3", then MatchesPatternTransactionSize="^no" counts one removal command. But because of the code hierarchy, three commands would actually be removed from the device:

```
Interface 3
    Speed
    PhysicalAddress
```

An inaccurate count is particularly likely on JuniperXml code.

General procedure for changing the regular expression

1. Start a command line session and find the desired network or device objectID.
2. Modify the regular expression (regex) as follows:

   ```
   modify [objectID]
   MatchesPatternTransactionSize="<reg_expression>"
   commit
   ```

3. Check if the regular expression is accepted:

   ```
   cat
   ```

   excerpt from response:

   rcw String MatchesPatternTransactionSize = "<reg_expression>"

Specific examples

The default global regular expression checks for "no" at the beginning of each command line. You can change it at the device level with a specific expression.

To count removal commands on a Huawei device, search for "undo" at the front of each command:

```
modify [objectID] MatchesPatternTransactionSize="^undo"
```

To count removal commands on a Juniper device, search for "delete="delete"" on the command line:

```
modify [objectID] MatchesPatternTransactionSize="delete=\"delete\""
```

At the network level, you can use a regular expression that counts removal commands for several device types:

```
modify [objectID]
MatchesPatternTransactionSize="(^no|^undo|delete=\"delete\")"
```

Chapter 5

Error Handling

This chapter provides details of error handling.

Exceptions

The API returns four general exception types. Specific information concerning the cause of the exception is contained in a descriptive string parameter called Reason.

CommandSyntaxException

This exception type occurs if the command syntax is not recognised.

- The command issued is unknown
- The path entered is malformed
- The attribute entered is unknown
- The command is malformed and has not been recognized
- Unexpected regular expression
- The attribute appears more than once in the parameter list
- The parameter supplied is not valid in this context

CommandExecutionError

Indicates that a command could not execute. The returned string is used to indicate the specific error condition:

- The command is invalid
- The system could not resolve the object supplied to an existing or a valid object
- A required parameter is missing from the argument list
- Too many parameters to the command
- The command requires a path
- The command requires precisely two path arguments
- The attribute value specified is out of range
- The specified EOM object type is invalid and cannot be created
- Cannot change the value for this attribute
- Command cannot be applied on this object or you do not have permission to perform this command
- Command has resulted in Object Model integrity violation
- The find command requires at least one Object, Attribute value or Regular Expression

- The find command does not accept a Regular Expression and an Object simultaneously
- This object is not a valid child of the specified class
- This object cannot be unlinked from the specified object
- This object cannot be linked to the specified object
- This object cannot be copied
- The copied object cannot be of the type specified
- The attribute values of the Global Template are read-only
- The type of the object is read-only and cannot be modified
- The link to a Global Template and the local values cannot be changed in the same time
- No child matching the given attributes could be found
- An attempt to merge a TransactionEntry has failed
- An attempt to rollback a TransactionEntry has failed
- The user does not have permissions to perform this command

OimSystemException

A general system error has occurred. The returned string is used to indicate the specific error condition. Examples are:

- Failure to communicate with the Policy Server
- Memory Error
- Shutdown is in progress
- This command cannot be executed

LoginException

Indicates errors occurring whilst trying to login to the API. The returned string is used to indicate the specific error condition:

- Invalid user name or password
- The login procedure has repeatedly failed

Appendix A

Command Grammar

This appendix provides full details of the command grammar.

The basic grammar for commands is described here using Regular Expressions:

token*	means zero to many appearances of the token
token+	means at least one appearance of the token
token?	means zero or one appearance of the token
(tokenA | tokenB)	means either tokenA or tokenB

```
command ::= COMMAND_NAME (' ' parameter )*
```

The command name is one of the following:

abort | alias | commit | copy | create | delete | discover | events | getAccess | find | findParameters | getAttributes | getChildren | getID | getName | getPath | getParents | gettargets | link | listTrans | login | logout | manage | merge | modify | rollback | schedule | setPath | subscribe | unlink | unsubscribe | unmanage | use | unuse | xmlexport

```
parameter ::= relative_path | absolute_path | attribute_couple | reg_exp
absolute_path ::= '/'  relative_path*
relative_path ::=  object  ('/' object )*
object ::= prefixed_name | object_id | ".." | '.'
object_id ::= '[' digit+ ']'
prefixed_name ::= EOM_TYPE ':' object_name
```

The EOM Type strings recognized are the objects accessible in the EOM. See *The External Object Model* on page 63 for full details.

```
object_name ::= string | letter+
attribute_couple ::= ATTRIBUTE_NAME  '='  attribute_value
Attribute names are defined in the OIM External Object Model Reference.
attribute_value ::=  string |  (letter | digit | '.' | '-' | '_' )+
regexp ::= '"' str_content * ( '?' | '*' ) str_content* '"'
string ::= '"' str_content * '"'
str_content::= <any non escaped character> | "\\" | "\?" | "\n" | "\*"
| "\""
```

> The quote characters may appear inside a string if it is escaped with a backslash. The backslash character in a quoted string needs to be escaped with a further backslash. For example:
>
> Domain:"Some domain"
>
> Domain:"A domain with \"embedded\" quote marks"

```
digit ::= [0-9]
letter ::= [a-z] | [A-Z]
```

Each command follows the same basic command format. For example, a command string using the `unmanage` command would look as follows:

```
unmanage domain:"Europe"/site:"London"/device:"UK120"
```

This example unmanages device 'UK120' within domain 'europe' and site 'London'.

Appendix B

Command Correlation

This appendix summarizes the commands that you can perform on each object in the EOM.

The following table indicates the commands that you can perform on each object in the EOM.

Object	create delete	modify	link unlink	copy	use unuse	discover	manage unmanage	merge rollback
Account			✓					
Classification	✓	✓	✓					
ClassificationGroup	✓	✓	✓					
Component								
ConcreteObject								
CoS	✓	✓						
Customer	✓	✓	✓					
Device	✓	✓	✓			✓	✓	
DeviceType	✓	✓						
Domain			✓					
DriverScript			✓					

Object	create delete	modify	link unlink	copy	use unuse	discover	manage unmanage	merge rollback
EthernetVlan	✓	✓	✓					
EventCollector	✓	✓	✓					
EventFilter	✓	✓	✓					
EventSubscription	✓	✓	✓					
ExternalSystem**	✓	✓	✓					
Fault								
Interface	✓		✓					
Network	✓	✓	✓					
Options		✓						
PacketMarking	✓	✓	✓					
ParameterSet*					✓			
ParameterSetInstance**		✓						
Period								
PHB	✓	✓	✓					
PHBAtm	✓	✓	✓					
PHBFrts	✓	✓	✓					
PHBGroup					✓			
PHBGroupInstance		✓						
PHBGroupMqc*	✓	✓	✓		✓			
PHBMqc	✓	✓	✓					
PHBPolicingAction	✓	✓	✓					

Object	create delete	modify	link unlink	copy	use unuse	discover	manage unmanage	merge rollback
Policy								
RoleDevice	✓		✓					
RoleInterface	✓		✓					
Root								
RtNumber	✓	✓	✓					
RuleAccess	✓	✓	✓	✓				
RuleClassification	✓	✓	✓	✓				
RulePolicing	✓	✓	✓	✓				
SAATemplate		✓	✓					
SAAOperation	✓	✓	✓					
Segment	✓		✓			✓		
Site	✓	✓	✓					
SiteHub	✓		✓					
SiteL2	✓	✓	✓					
StaticRoute	✓	✓	✓					
SubLayer	✓		✓					
SubInterface	✓		✓					
System								
Tls	✓	✓	✓					
Topology								
TrafficApplication			✓					

Object	create delete	modify	link unlink	copy	use unuse	discover	manage unmanage	merge rollback
TrafficCompound			✓					
TrafficDomainName			✓					
TrafficGroup			✓					
TrafficMime			✓					
TrafficPacketMarking			✓					
TrafficPort			✓					
TrafficSubApplication			✓					
TrafficURL	✓		✓					
TransactionEntry								✓
VcEndpointAtm	✓	✓	✓					
VcEndpointFr	✓	✓	✓					
VlanInterface	✓	✓	✓					
Vpn	✓	✓	✓					

* not unuse

** not create, but delete

Appendix C

Sample Service Activator transaction publisher class

This appendix illustrates how to use Service Activator External Object Model (EOM) events that are generated by Service Activator OSS Integration Manager (OIM) on TransactionEntry objects. The sample OSS Java Development Library (OJDL) code, shown on page 392 and explained with examples page 401, transforms the EOM events into Java Message Service (JMS) events, which are then published on a JMS topic.

The events are meant to be used by an external JMS-enabled application to track the activation status of its Service Activator transactions. Activation status information in Service Activator is available for all transactions through TransactionEntry's ProvisioningStatus and ReasonForFailure attributes.

Changes to these two attributes, as well as TransactionEntry's State attribute, are encoded in the JMS messages and can be easily decoded by the external client. The format of the messages can also be customized.

Two types of JMS events are generated: TransactionCreation and AttributeChange. These events are described in the following sections.

TransactionCreation

This event is generated when a transaction is created by a Service Activator client. The transaction for the generated create event is treated as active in the sample. All active transactions are stored in a container, so that subsequent events on them can be analyzed and translated into AttributeChange events.

AttributeChange

This event is generated when one or more attributes of an active transaction change. The event indicates that one of the following has occurred:

(1) Transaction's State attribute changed. This is an indication that a scheduled transaction has been merged into the Service Activator object model and its activation has begun.

(2) Transaction's ProvisioningStatus attribute changed. Three cases are possible:

(2.1) Transaction's ProvisioningStatus changed to Success. This is an indication that directly committed or scheduled transaction completed activation successfully.

(2.2) Transaction's ProvisiongStatus changed to Failed. This is an indication that directly committed or scheduled transaction has failed activation. The same event should contain new value for the ReasonForFailure attribute.

(2.3) Transaction's ProvisiongStatus changed to Timedout. This is an indication that activation status of directly committed or scheduled transaction was not confirmed during the configured timeout period.

Notes:

If AttributeChange is generated and indicates that (1) occurred, it is followed by another AttributeChange indicating that either (2.1), (2.2) or (2.3) occurred.

An AttributeChange indicating (2.1), (2.2) or (2.3) is the final event for a transaction.

Sample OJDL file

The following is a sample OJDL file. Various parts of the sample file are explained in the section *Coding examples* starting on page 401.

```
/*

Copyright (c) 2007,2008 Oracle. All rights reserved. Oracle is a trademark of
Oracle Corporation and/or its affiliates. Other names may be trademarks of their
respective owners.

 */
import java.util.HashMap;
import java.util.Hashtable;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

import com.mslv.osa.ojdl.eom.EomException;
import com.mslv.osa.ojdl.eom.EomSession;
```

```
import com.mslv.osa.ojdl.eom.EomSessionException;

import javax.jms.JMSException;
import javax.jms.Session;
import javax.jms.TextMessage;
import javax.jms.Topic;
import javax.jms.TopicConnection;
import javax.jms.TopicConnectionFactory;
import javax.jms.TopicPublisher;
import javax.jms.TopicSession;

import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;

public class SampleTransactionStatusPublisher implements Runnable {
/**
* Main method which allows running the sample publisher as a stand-alone command
line utility.
* Usage:
* java SampleTransactionStatusPublisher
*
* @param argv array of command line parameters
*/
public static void main( String [] argv) {
       SampleTransactionStatusPublisher samplePublisher = new
               SampleTransactionStatusPublisher();
       samplePublisher.run();
    }

/**
* Implements run method from the java.lang.Runnable interface. Allows running
* the publisher on its own detached thread within another application.
*
* @see java.lang.Runnable#run()
*/
public void run() {
       try {
              openEomSession();
              while( true) {

                     String [] events = getEomEvents();

                     for (int i = 0; i < events.length; i++) {
                     System.out.println(events[i]);
                     processEomEvent( events[i]);
                     }
```

```
                    Thread.sleep( IPSA_EVENT_POLLING_INTERVAL);
             }
             }
       catch (InterruptedException e) {}
             catch (Exception e) {
                    e.printStackTrace();
             }
             finally {
                    closeEomSession();
             }
    }

       // IPSA parameters
       protected final String  IPSA_HOST = "192.168.149.128";
       protected final short   IPSA_PORT = 2809;
       protected final String  IPSA_USER = "admin";
       protected final String  IPSA_PASSWORD = "orchestream";
       protected final int     IPSA_NUM_RECONNECT_ATTEMPTS = 3;
       protected final int     IPSA_EVENT_POLLING_INTERVAL = 5000; // 5 seconds
       protected final boolean IPSA_EXIT_ON_RECONNECT_FAILURE = true;

    // JMS parameters
       protected final String JMS_SERVER_URL = "t3://127.0.0.1:7001";
       protected final String JMS_USER="weblogic";
       protected final String JMS_PASSWORD="weblogic";
       protected final String
JMS_CONTEXTFACTORY="weblogic.jndi.WLInitialContextFactory";
       protected final String
JMS_CONNECTIONFACTORY="com.bea.wli.b2b.server.TopicConnectionFactory";
       protected final String JMS_TOPICNAME="ipsa.transaction.event.topic";

    // pattern for matching transaction creation events
       protected final Pattern createEomEventPattern = Pattern.compile( "create
TransactionEntry:\"(.+?)\"" +
             ".*\\sId=\"(.+?)\"" +
             ".*\\sNumberOfConcretes=\"(.+?)\"" +
             ".*\\sProvisioningStatus=\"(.+?)\"" +
             ".*\\sReasonForFailure=\"(.*?)\"" +
             ".*\\sSchedule=\"(.+?)\"" +
             ".*\\sState=\"(.+?)\"", Pattern.DOTALL);

// pattern for matching modify events
       protected final Pattern modifyEomEventPattern = Pattern.compile( "modify
\\[(.+?)\\]");

// patterns for matching transaction attributes in the modify events
```

```
protected final Pattern provisioningStatusPattern = Pattern.compile(
"\\sProvisioningStatus=\"(.*?)\"");
protected final Pattern reasonForFailurePattern = Pattern.compile(
"\\sReasonForFailure=\"(.*?)\"");
protected final Pattern schedulePattern = Pattern.compile(
"\\sSchedule=\"(.*?)\"");
protected final Pattern statePattern = Pattern.compile( "\\sState=\"(.*?)\"");

// template for the generated transaction creation events protected final String
createEventTemplate =
       "TransactionCreation(\n" +
       "Id: \"%Id%\",\n" +
       "Name: \"%Name%\",\n" +
       "NumberOfConcretes: \"%NumberOfConcretes%\",\n" +
       "ScheduleTime: \"%ScheduleTime%\",\n" +
       "State: \"%State%\"\n" +
       ")\n";

// template for the generated attribute change events protected final String
attributeChangeEventTemplate =
       "AttributeChangeEvent(\n" +
       "Id: \"%Id%\",\n" +
       "Name: \"%Name%\",\n" +
       "%AttributeChanges%)\n";

// template for single attribute change within attribute change events protected
final String attributeChangeTemplate =
       "%AttributeName%: ( \"%OldValue%\", \"%NewValue%\")\n";

// class with all transaction attributes of interest protected class
TransactionInfo {
       public String Id;
       public String Name;
       public String NumberOfConcretes;
       public String ProvisioningStatus;
       public String ReasonForFailure;
       public String ScheduleTime;
       public String State;
    }

/** Initializes JMS session and publisher objects
* @throws NamingException
* @throws JMSException
*/
protected void initializeJms() throws NamingException, JMSException {
      Hashtable<String,Object> ht = new Hashtable<String,Object>();
      ht.put(Context.INITIAL_CONTEXT_FACTORY, JMS_CONTEXTFACTORY);
```

```
      ht.put(Context.PROVIDER_URL, JMS_SERVER_URL);
      ht.put(Context.SECURITY_PRINCIPAL, JMS_USER);
      ht.put(Context.SECURITY_CREDENTIALS, JMS_PASSWORD);

      Context jndiContext = new InitialContext(ht);

      TopicConnectionFactory topicConnectionFactory =
           TopicConnectionFactory) jndiContext.lookup(JMS_CONNECTIONFACTORY);
      Topic topic = (Topic) jndiContext.lookup(JMS_TOPICNAME);
      TopicConnection topicConnection =
         topicConnectionFactory.createTopicConnection();
      m_jmsSession = topicConnection.createTopicSession(false,
         Session.AUTO_ACKNOWLEDGE);
      m_jmsPublisher = m_jmsSession.createPublisher(topic);
    }

/** Opens EOM session
* @throws EomException
*/
protected void openEomSession() throws EomException {
        m_eomSession = new EomSession( IPSA_HOST, IPSA_PORT,
           "master_integration_manager");
        m_eomSession.login( IPSA_USER, IPSA_PASSWORD);
}

/**
*
*/
protected void closeEomSession() {
     try {
             m_eomSession.logout();
             m_eomSession.close();
           }
      catch (Exception e)
      {}
    }

/** Retrieves an array of the events that queued by the OJDL library since the *
previous call to this method. If any exception is raised, due to connectivity *
or other problems, the method will retry up to IPSA_NUM_RECONNECT_ATTEMPTS * each
time instantiating a new EOM session.
*
*  Called periodically by the run method
*
* @return array of Strings
* @throws InterruptedException
*/
```

```
    protected String [] getEomEvents() throws InterruptedException {
       String [] events = null;
       for( int numAttempts = 0; numAttempts <= IPSA_NUM_RECONNECT_ATTEMPTS;
          ++numAttempts) {
              try {
                     if( m_eomSession == null && numAttempts == 0) {
                            // first time opening of the EOM session
                            openEomSession();
                     }
                     else if( numAttempts > 0) {
                            // reconnection attempt
                     System.err.println("Re-attempting to establish EOM
                            session
                        to host: " + IPSA_HOST +
                                    " port: " + IPSA_PORT + ". Attempt # " +
                                           numAttempts);
                     openEomSession();
              }
                     events = m_eomSession.getEvents(false);
                     break;
              }
              catch(Exception e) {
                     System.err.println( e.toString());
                     if( IPSA_EXIT_ON_RECONNECT_FAILURE && numAttempts ==
                        IPSA_NUM_RECONNECT_ATTEMPTS) {
                            System.err.println("Cannot establish EOM session
after " +
                                           IPSA_NUM_RECONNECT_ATTEMPTS + "attempts.
                                                  Exiting.");
                            System.exit(1);
                     }
              }
                     Thread.sleep( IPSA_EVENT_POLLING_INTERVAL);
       }
       return events;
    }

    /** Process create event for a TransactionEntry object
     *
     * @param match
     */
    protected void onCreateEvent( Matcher match) {
        TransactionInfo infoObject = new TransactionInfo();
        infoObject.Name = match.group( 1);
        infoObject.Id = match.group( 2);
        infoObject.NumberOfConcretes = match.group( 3);
        infoObject.ProvisioningStatus = match.group( 4);
```

```
        infoObject.ReasonForFailure = match.group( 5);
        infoObject.ScheduleTime = match.group( 6);
        infoObject.State = match.group( 7);

      m_activeTransactions.put( infoObject.Id, infoObject);

      String messageText = createEventTemplate
            .replaceFirst("%Id%", infoObject.Id)
            .replaceFirst("%Name%", infoObject.Name)
            .replaceFirst("%NumberOfConcretes%", infoObject.NumberOfConcretes)
            .replaceFirst("%ProvisioningStatus%",
                    infoObject.ProvisioningStatus)
            .replaceFirst("%ReasonForFailure%", infoObject.ReasonForFailure)
            .replaceFirst("%ScheduleTime%", infoObject.ScheduleTime)
            .replaceFirst("%State%", infoObject.State);

      publishJmsMessage( messageText);
    }

/** Process a modify event on a TransactionEntry object for which a create
* event has been previously processed.
*
* @param match Matcher object that matched the event
* @param modifyEvent modify event matched by the match parameter
*/
protected void onModifyEvent( Matcher match, String modifyEvent) {
      String objectId = match.group(1);
      // is the modify event on one of the active transactions
      if( m_activeTransactions.containsKey( objectId)) {
            Matcher provisioningStatusMatcher =
              provisioningStatusPattern.matcher( modifyEvent);
            Matcher reasonForFailureMatcher = reasonForFailurePattern.matcher(
modifyEvent);
            Matcher scheduleMatcher = schedulePattern.matcher( modifyEvent);
            Matcher stateMatcher = statePattern.matcher( modifyEvent);

            TransactionInfo infoObject = m_activeTransactions.get(objectId);
         StringBuffer attributeChanges = new StringBuffer();

           if( scheduleMatcher.find()) {
              attributeChanges.append( attributeChangeTemplate
                     .replaceFirst( "%AttributeName%", "ScheduleTime")
                     .replaceFirst( "%NewValue%", scheduleMatcher.group(1))
                       .replaceFirst( "%OldValue%", infoObject.ScheduleTime));
            }
            if( stateMatcher.find()) {
              attributeChanges.append( attributeChangeTemplate
```

```
                        .replaceFirst( "%AttributeName%", "State")
                        .replaceFirst( "%NewValue%", stateMatcher.group(1))
                        .replaceFirst( "%OldValue%", infoObject.State));
               }
               if( provisioningStatusMatcher.find()) {
                  attributeChanges.append( attributeChangeTemplate
                            .replaceFirst( "%AttributeName%", "ProvisioningStatus")
                        .replaceFirst( "%NewValue%",
                              provisioningStatusMatcher.group(1))
                        .replaceFirst( "%OldValue%",
                              infoObject.ProvisioningStatus));

                  // provisioning status changes are final
                  m_activeTransactions.remove(objectId);
               }
               if( reasonForFailureMatcher.find()) {
                  attributeChanges.append( attributeChangeTemplate
                           .replaceFirst( "%AttributeName%", "ReasonForFailure")
                           .replaceFirst( "%NewValue%",
                                reasonForFailureMatcher.group(1))
                           .replaceFirst( "%OldValue%",
                                infoObject.ReasonForFailure));
               }
              String messageText = attributeChangeEventTemplate
                  .replaceFirst("%Id%", infoObject.Id)
                  .replaceFirst("%Name%", infoObject.Name)
                  .replaceFirst("%AttributeChanges%",
                              attributeChanges.toString());

              publishJmsMessage( messageText);
        }
     }

/** Publish a JMS message with the specified message text
*
* @param messageText text of the message
*/
protected void publishJmsMessage( String messageText) {

       System.out.println("Publishing JMS message:");
       System.out.println(messageText);

        try {
                   if( m_jmsSession == null) {
                       initializeJms();
                   }
                   TextMessage message = m_jmsSession.createTextMessage();
```

```
                    message.setText(messageText);
                    m_jmsPublisher.publish(message);

              } catch (Exception e) {
             System.err.println( e.toString());
                     // cause re-initialization of the JMS session
                     m_jmsSession = null;
              }
    }

/** Processes the event string passed as input parameter. Two types of events
* are being processed:
* - create on TransactionEntry and
* - modify on objects for which create has been previously processed
*
* @param eomEvent the event string to be processed
*/
protected void processEomEvent( String eomEvent) {
       // matchers designed to match EOM events on transaction entry objects
       only
       Matcher createEomEventMatcher = createEomEventPattern.matcher(
       eomEvent);
       Matcher modifyEomEventMatcher = modifyEomEventPattern.matcher(
       eomEvent);
              if( createEomEventMatcher.find()) {
                     onCreateEvent( createEomEventMatcher);
              }
              else if( modifyEomEventMatcher.find()) {
                     onModifyEvent( modifyEomEventMatcher, eomEvent);
              }
    }

// Map of active transaction information keyed by transaction IDs private
HashMap<String,TransactionInfo>  m_activeTransactions = new
HashMap<String,TransactionInfo>();

// EOM session to poll for events
       private EomSession m_eomSession = null;

       // JMS session and topic publisher
private TopicSession   m_jmsSession = null;
private TopicPublisher m_jmsPublisher = null;
}
```

Coding examples

The following examples illustrate how the code in the OJDL sample file (shown starting on page 392) can be used to transforms the OSS Integration Manager (OIM) events on TransactionEntry objects into Java Message Service (JMS) events.

In this section, OSS Integration Manager (OIM) events are defined as events generated by the Service Activator OIM. OJDL clients poll for those events by calling EomSession.getEvents.

Implementation of the java.lang.Runnable interface

Implements the run() method of Runnable. The run() method executes an infinite loop that is only interrupted when the application exits. It repeatedly calls

```
getEomEvents()
```

The returned events are then processed:

```
for (int i = 0; i < events.length; i++) {
  System.out.println(events[i]);
  processEomEvent( events[i]);
}
```

The run method pauses before repeating the same cycle:

```
Thread.sleep( IPSA_EVENT_POLLING_INTERVAL);
```

A real transaction publisher will be executing the run method on a separate thread concurrently with other activities.

EOMSession management

An EOM session is used in order to obtain OIM events:

```
events = m_eomSession.getEvents(false);
```

The session is reestablished when exceptions occur (for example due to IP connectivity loss):

```
// reconnection attempt
System.err.println("Re-attempting to establish EOM session to
   host: " + IPSA_HOST + " port: " + IPSA_PORT + ". Attempt
   # " + numAttempts);
openEomSession();
```

Up to [IPSA_NUM_RECONNECT_ATTEMPTS] attempts are made to reestablish a failed session

```
for( int numAttempts = 0; numAttempts <=
IPSA_NUM_RECONNECT_ATTEMPTS;    ++numAttempts) {
...
   System.err.println("Re-attempting to establish EOM
   session to host: " + IPSA_HOST +
       " port: " + IPSA_PORT + ". Attempt # " + numAttempts);
   openEomSession();
...
}
```

Handling of create events

Create events on TransactionEntry objects are recognized using a pattern matcher and processed in the onCreateEvent() method:

```
...
Matcher createEomEventMatcher =
   createEomEventPattern.matcher( eomEvent);
...
if( createEomEventMatcher.find()) {
   onCreateEvent( createEomEventMatcher);
}
```

The onCreateEvent () method stores a record of the transaction being created (referred to as active transaction):

```
  m_activeTransactions.put( infoObject.Id, infoObject);
```

Next, it formats the text of the outgoing JMS message and calls publishJmsMessage().

Handling of modify events

Modify events on active transactions are recognized using a pattern matcher and processed in the onModifyEvent() method:

```
Matcher modifyEomEventMatcher =
   modifyEomEventPattern.matcher( eomEvent);
...
else if( modifyEomEventMatcher.find()) {
   onModifyEvent( modifyEomEventMatcher, eomEvent);
}
```

The onModifyEvent method makes sure that the event is on an active transaction before processing it:

```
if( m_activeTransactions.containsKey( objectId)) {
    ...
}
```

The onModifyEvent method uses four patterns and matchers to match changes on the following attributes: ScheduleTime, State, ReasonForFailure, and ProvisioningStatus.

In addition to the new value, the old value of the attribute is retrieved from the stored info object. Both values are used to format an attribute change clause in the text of the outgoing JMS event. More than one attribute change clause can occur in the same JMS event. This would typically be the case when the event indicates Provisioning Status changed to failed. The same event will contain ReasonForFailure.

All attribute change clauses are concatenated to form the text of the outgoing JMS message passed to publishJmsMessage().

Customizing the format of the JMS message

The sample uses templates for the text of both TransactionCreate:

```
protected final String createEventTemplate =
   “TransactionCreation(\n” +
   “Id: \”%Id%\”,\n” +
   “Name: \”%Name%\”,\n” +
   “NumberOfConcretes: \”%NumberOfConcretes%\”,\n” +
   “ScheduleTime: \”%ScheduleTime%\”,\n” +
   “State: \”%State%\”\n” +
   “)\n”;
```

and AttributeChange events:

```
protected final String attributeChangeEventTemplate =
   “AttributeChangeEvent(\n” +
   “Id: \”%Id%\”,\n” +
   “Name: \”%Name%\”,\n” +
   “%AttributeChanges%)\n”;
```

There is also a template for a single attribute change:

```
protected final String attributeChangeTemplate =
   “%AttributeName%: ( \”%OldValue%\”, \”%NewValue%\”)\n”;
```

These templates contain markers in the format %markername% which are replaced with the actual values. The templates can be changed to any desired format. For example, you can change the templates so that the messages conform to a particular XML schema or DTD.

Publishing the outgoing message

The JMS message is published from within the publishJmsMessage:

```
TextMessage message = m_jmsSession.createTextMessage();
message.setText(messageText);
m_jmsPublisher.publish(message);
```

The sample uses a JMS topic to publish the message. This may be changed to a queue if the receiver is using a queue.

Initializing the JMS entities

All JMS initialization is done in the initializeJms() method. This method may require customization to suit the specific application needs.

Index