CA Spectrum®

Certification User Guide

Release 9.4

CA Technologies Product References

This document references CA Spectrum® (CA Spectrum).

Contact CA Technologies

Contact CA Support

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

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

Providing Feedback About Product Documentation

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

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

Contents

Chapter 1: Out-of-the-Box Certification Support 9

Overview 9
About Generic Certification 10
Device Modeling 11
    How CA Spectrum Identifies the Device Type 11
    How CA Spectrum Identifies the Model Class 12
Support for Chassis Devices 13
    Identification of Chassis Devices 14
    Chassis Views 15
    The Locater Search 19
    Chassis Alarms 20
Reconfigure Existing Models with New Certification Support 21
Interface Modeling 21
Application Modeling 22
Traps, Events, and Alarms 23

Chapter 2: Certification Locater 25

Access the Device Certification Database 25

Chapter 3: Self-Certification 29

About Self-Certification 29
Adding Trap Support 29
Watches to Monitor and Manage Model Conditions 31
Lock the Device Type Setting for a Device Model 31
Lock Device Model Settings 32

Chapter 4: Customizing Identification with Device Certification 33

Device Certification in OneClick 33
Open the Device Certification Dialog 34
About the Device Certification Dialog 34
Device Certification Table 35
Search Device Certification Mappings Using Filters 36
Device Certification Changes 37
Device Mappings 37
    Custom Device Type Mappings 37

Modify Device Certification Entries 38
View Default Values Masked by Custom Device Certifications 40
Copy Device Certification Mappings to Create New Mappings 42
Map Unregistered Devices 43
OneClick Views 44
Delete Custom Device Certification Mappings 44
Distributed SpectroSERVER Support 46
Resolve Device Certification Mapping Conflicts 46
Device Certification Changes are Not Saved 47
Change the Model Type for a Single Device Type 48
Device Certification and Fault-Tolerant Environments 49

Chapter 5: Managing MIBs and Traps With MIB Tools 51

The MIB Tools Utility 51
How a MIB Is Organized 52
MIB Tools Database 52
OneClick MIB Tools Overview 53
Start MIB Tools 53
MIB Tools User Interface 55
MIB Tree Hierarchy Table 56
Attribute Support Table 58
Trap Support Table 59
Import and Export MIBs 61
Import Individual MIBs 61
Delete Individual MIBs 63
Editing MIBs 63
Import Multiple MIBs 63
Create Attribute Support 66
Modifying MIB Objects in the MIB Tools Database 66
Query (GET_NEXT), GET, and SET Requests 67
Query a Subtree of Objects 67
Query an Object 68
Set an Object 68
Device Query and SET Results 70
Export Query Results To Support Troubleshooting 71
Custom Vendor Folders 72
Create Custom Vendor Folders 72
Edit Custom Vendor Folders 72
Delete Custom Vendor Folders 73
Move MIBs to Custom Vendor Folders 73
Contact a Device Using MIB Tools 74

Search for a MIB 75
Trap Support 76
Custom Trap Support File Details 76
Create Trap Support 77
Review Custom Trap Mapping Information 78
Remove Custom Trap Mappings 79
Remove Partial Mappings From Traps 79
Show Advanced Options for Mapping Traps 80
MIB Tools Support for Multiple SpectroSERVERs 82
MIB Tools Synchronization in a DSS Environment 82
Attribute Conflicts 82
Create Consistent Support Across a DSS Environment 83
Synchronize and Update MIB Databases and Support Files on Multiple OneClick Servers 84
Trap Disposition Conflicts 85
Resolve Trap Disposition Conflicts: Remap the Trap 85
Resolve Trap Disposition Conflicts: Edit the AlertMap and EventDisp Files 86

Chapter 6: Developing a New Certification 87

New Certification Management 87
Additional MIB Support 88
Unique Trap Mapping 88
Unique Watches 88
Interface Model Creation 89
New Device Model Type 89
New Device Model Type Design 90
New Device Model Type Creation 91
New Device Model Type Configuration 91
Creating a New Application Model Type 100
Derivation Points and Model Fragments 100
Derivation Point 102
Board and Port Considerations 102
Port-Oriented Devices 103
Chassis Devices 103
GnChassisDerPt 103
GnRelayDerPt 104
Application Model Types 106
Modeling Ports and Boards 109
Port and Board Model Information 110
How to Add Support for Additional Traps 111
Distributing a New Certification 112

Index 115

Chapter 1: Out-of-the-Box Certification Support

This section contains the following topics:

Overview (see page 9)
About Generic Certification (see page 10)
Device Modeling (see page 11)
Support for Chassis Devices (see page 13)
Reconfigure Existing Models with New Certification Support (see page 21)
Interface Modeling (see page 21)
Application Modeling (see page 22)
Traps, Events, and Alarms (see page 23)

Overview

Support for monitoring many devices is provided out-of-the-box in CA Spectrum. Basic monitoring support is supplied through simple or enhanced certifications:

- **Simple Support -** The device is modeled using the CA Spectrum generic certification. This level of certification provides core CA Spectrum capabilities, including discovery, identification, standard MIB and trap support, and standard views. Simple support also includes interface modeling and participation in fault isolation and root cause analysis.
- **Enhanced Support** - The device is modeled using one of the CA Spectrum enhanced certifications, which extend simple certification support. At a minimum, enhanced certification support indicates that support for this device has been extended with proprietary MIB and trap support. Typical extensions include proprietary OneClick views, CPU and memory device thresholding, and serial number support.

More information:

Device Certification Table (see page 35)

About Generic Certification

CA Spectrum provides a generic certification to represent an SNMP-compliant network device that lacks a corresponding CA Spectrum enhanced certification. Management Information Bases (MIBs) support SNMP-compliant devices. MIBs are SNMP structures that describe particular devices. MIBs are imported into the CA Spectrum database and made available through device, application, and interface model types.

Note: For more information about simple and enhanced certification support, see the *Standards-Based Protocol Reference Guide.* For more information about enhanced certification support, see the *Device Management Reference Guide*, *Cisco Device Management Guide*, and *Host System Resources Management User Guide.*

The generic model type, GnSNMPDev, can represent a broad range of devices by creating the following models:

- A model to represent the device.
- Application models to represent each of the standard (IETF) MIBs that the device supports.
- Interface models to represent device ports.

GnSNMPDev lets CA Spectrum dynamically create models to manage devices when a specific management module is unavailable.

GnSNMPDev rapidly queries the device to determine its characteristics and capabilities and then creates a model to represent the device. GnSNMPDev also creates the following models:

- Submodels, referred to as application models, to represent each of the standard MIBs that the device supports.
- Interface models to represent each device port that is defined in the standard MIB-II interface table.

The application and interface models are associated with the GnSNMPDev device model. Together, they provide management capabilities for the device.

Devices that are modeled with the GnSNMPDev model type can be used with all CA Spectrum management tools. GnSNMPDev models participate fully in CA Spectrum root cause analysis, fault isolation, and downstream alarm suppression algorithms. As a result, they can alert users to network and device problems.

More information:

Application Modeling (see page 22)

Device Modeling

When modeling a device using Discovery or the Model by IP Address icon, CA Spectrum automatically chooses the GnSNMPDev model type when an enhanced certification for the device is not available. You can also model a device using the GnSNMPDev model type when you use the Model by Type feature.

You can map the connectivity of interface models automatically using Discovery, or you can map connectivity manually.

The GnSNMPDev model type supports the Cisco Proprietary Discovery Protocol (CDP). A CiscoCDPApp application model is created for Cisco devices that are modeled with GnSNMPDev and that support CDP. This application model lets CA Spectrum use the Proprietary Discovery tables for the device when discovering device connectivity information.

Note: For more information, see the *Modeling and Managing Your IT Infrastructure Administrator Guide*.

More information:

Application Modeling (see page 22)

How CA Spectrum Identifies the Device Type

When modeling a device, CA Spectrum assigns a descriptive identifier or device type. The device icon shape and label reflect device functionality in OneClick.

The DeviceType attribute (0x23000e) is a text string that identifies the type of device being modeled. In OneClick, this string is displayed below the device icon. CA Spectrum lets you search, filter, and report on device models using the DeviceType attribute.

The following process describes how CA Spectrum determines the device type to assign to a device model:

1. If the device type setting is locked, CA Spectrum does not reevaluate the device model. The device type that is set for the device model remains.
2. If the device type setting is not locked, CA Spectrum runs custom intelligence for some models to set the device type name.
3. CA Spectrum checks the System Object Identifier-to-Device Type mapping list. If a device type name (for example, “Cisco 2621”) is found for the device System Object Identifier, it becomes the model device type. If no match is found, CA Spectrum extracts the device enterprise ID from the System Object Identifier. CA Spectrum uses the enterprise ID to identify the manufacturer.

4. CA Spectrum then checks device capabilities and appends an abbreviation (for example, Rtr or Bdg) to the manufacturer name. This entire string becomes the device type name in OneClick (for example, “Cisco Rtr”).
5. If CA Spectrum cannot determine an appropriate device type, the default value “SNMP DV” is assigned.

More information:

About the Device Certification Dialog (see page 34)

How CA Spectrum Identifies the Model Class

CA Spectrum evaluates the model class when a device is modeled for the first time and when you reconfigure a device model.

The following process describes how CA Spectrum determines the model class to assign to a device model:

1. If the model class setting is locked, CA Spectrum does not reevaluate the device model. The model class that is set for the device model remains as is.
2. If the model class setting is not locked, CA Spectrum checks device model support for a specific MIB object. If CA Spectrum detects that a device model supports a certain MIB object, the model class for that device model is set to a specified value.
3. If the search for a supported MIB object fails, CA Spectrum attempts to determine the model class for the device model. CA Spectrum uses the mappings in the Device Certification utility in this search. This utility provides a mapping from System Object ID (which is more general than a MIB object) to Model class.
4. If Device Certification does not contain any model class mappings for a device, CA Spectrum defaults to setting the model class based on whether the device appears to be routing (“Router”), switching (“Switch”), both switching and routing ("Switch-Router”) or simply repeating (“Repeater”).

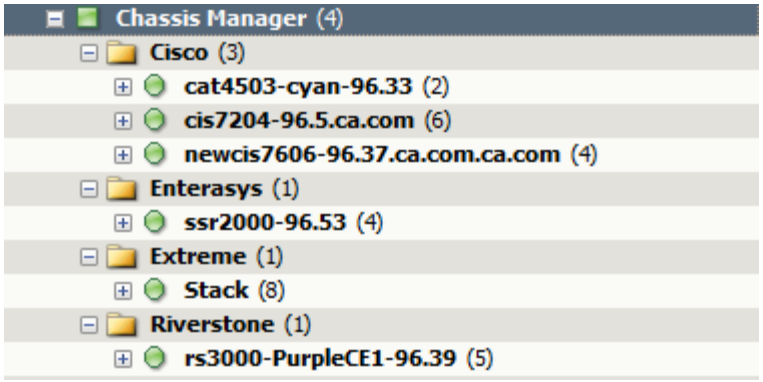
When assigning the icon and label for a device model, CA Spectrum uses the icon for the model class that is assigned as described here. This icon appears throughout OneClick.

Support for Chassis Devices

When a device model uses a CA Spectrum certified proprietary chassis MIB or the Entity MIB, CA Spectrum identifies that device model as a chassis device.CA Spectrum models and arranges all the identified chassis devices with their components or modules under the Chassis Manager node in the navigation pane of OneClick. This arrangement is based on the vendor names of your chassis devices.

For example, devices of "Cisco" that are identified as chassis devices are arranged in a folder with the name "Cisco". Similarly, devices of "Enterasys" identified as chassis devices are arranged in the "Enterasys" folder.

The following image shows how chassis devices are modeled and arranged under the Chassis Manager node. Each vendor folder contains its chassis devices:



As a result, you can monitor the health of your chassis devices and their modules at one place in OneClick, the Chassis Manager node. This node provides a consolidated location to view and manage chassis devices that are modeled in the Universe topology. After selecting a chassis device from this node and then accessing chassis views from the Component Detail pane, you can view the status of all interfaces, and can assess the health of each module. For more information about the chassis views that give detailed information about each module of your chassis device, see Chassis Views (see page 15).

Identification of Chassis Devices

CA Spectrum identifies your device as a chassis device based on the following two types of MIBs:

Proprietary MIB

When a device supports a CA Spectrum certified proprietary chassis MIB, it is identified as a chassis device. For example, when a "Cisco" device supports the "CISCO-STACK-MIB", it is identified as a chassis device based on that MIB.

Note: CA Spectrum always prefers the CA Spectrum certified proprietary chassis MIB to the Entity MIB of a device model for its identification as a chassis device. Only when the proprietary chassis MIB is absent, the device model is identified as a chassis device using the Entity MIB.

Entity MIB

When a device model supports the Entity MIB and the value of the "EnableEntityModuleModeling" attribute is "Yes" for that device model, it is identified as a chassis device. By default, the value of this attribute is "Yes" on a case-by-case basis for the following reasons:

- Some vendors do not implement this MIB indexing scheme correctly.
- Some vendors support the Entity MIB even for non-chassis devices.

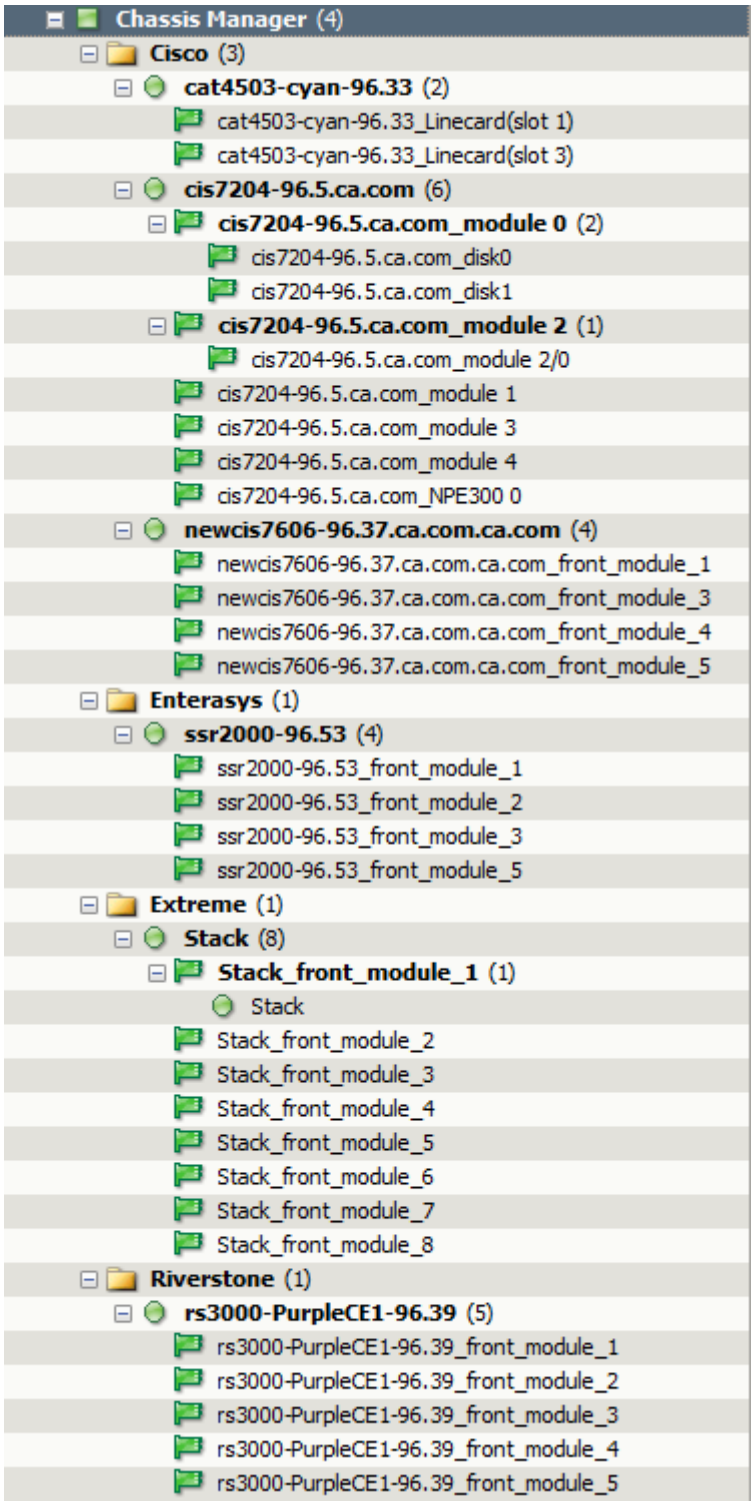
Note: If you do not want CA Spectrum to identify a device model as a chassis device, set the value of this attribute to "No" and reconfigure the model.

Chassis Views

To view the details of your chassis devices, CA Spectrum displays the following three types of chassis views in OneClick:

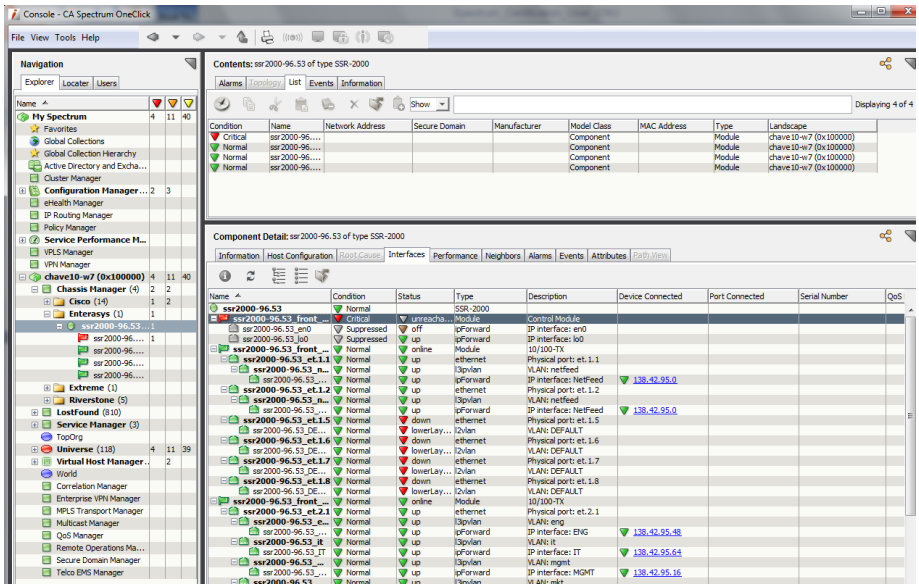
The Basic Module-Level View

This view of your chassis device can be viewed at the Chassis Manager node. The following image shows how chassis devices of Cisco, Enterasys, Extreme, and Riverstone are arranged under the Chassis Manager Node with their modules:


Chassis Manager (4)
Cisco (3)
cat4503-cyan-96.33 (2)
cat4503-cyan-96.33_Linecard(slot 1)
cat4503-cyan-96.33_Linecard(slot 3)
cis7204-96.5.ca.com (6)
cis7204-96.5.ca.com_module 0 (2)
cis7204-96.5.ca.com_disk0
cis7204-96.5.ca.com_disk1
cis7204-96.5.ca.com_module 2 (1)
cis7204-96.5.ca.com_module 2/0
cis7204-96.5.ca.com_module 1
cis7204-96.5.ca.com_module 3
cis7204-96.5.ca.com_module 4
cis7204-96.5.ca.com_NPE300 0
newcis7606-96.37.ca.com.ca.com (4)
newcis7606-96.37.ca.com.ca.com_front_module_1
newcis7606-96.37.ca.com.ca.com_front_module_3
newcis7606-96.37.ca.com.ca.com_front_module_4
newcis7606-96.37.ca.com.ca.com_front_module_5
Enterasys (1)
ssr2000-96.53 (4)
ssr2000-96.53_front_module_1
ssr2000-96.53_front_module_2
ssr2000-96.53_front_module_3
ssr2000-96.53_front_module_5
Extreme (1)
Stack (8)
Stack_front_module_1 (1)
Stack
Stack_front_module_2
Stack_front_module_3
Stack_front_module_4
Stack_front_module_5
Stack_front_module_6
Stack_front_module_7
Stack_front_module_8
Riverstone (1)
rs3000-PurpleCE1-96.39 (5)
rs3000-PurpleCE1-96.39_front_module_1
rs3000-PurpleCE1-96.39_front_module_2
rs3000-PurpleCE1-96.39_front_module_3
rs3000-PurpleCE1-96.39_front_module_4
rs3000-PurpleCE1-96.39_front_module_5

The Interfaces View

This view is an elaborated view of all interfaces present in each module of your chassis device. For a selected chassis device, this view shows all of its interface modules, interfaces within each module, status of modules and its interfaces, and other information under the Interfaces tab of the Component Detail pane. The following image shows how an "Enterasys" chassis device is populated under the Interfaces tab:



The Entity View

CA Spectrum populates this view only for those chassis devices which support the Entity MIB. The Entity View is populated when you expand the "Entity View" under the Information tab of the Component Detail pane. This view has the following two sections:

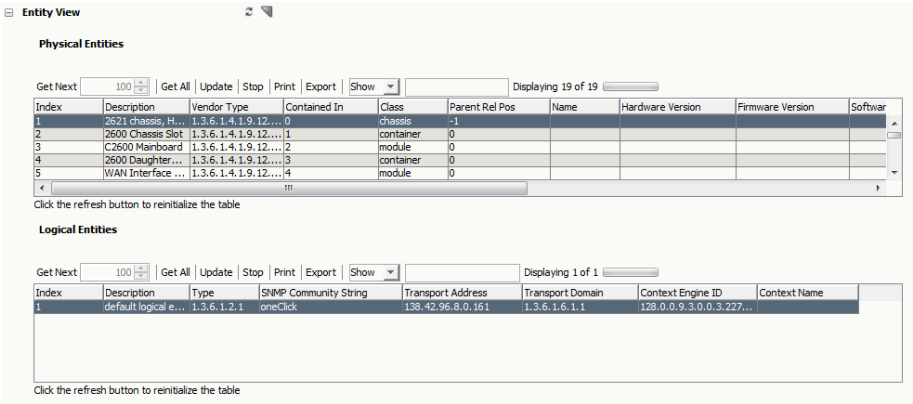
Physical Entities

This section populates the information about each module that exists in your chassis device.

Logical Entities

This section populates the information about the logical entities that exist in each module of your chassis device.

The following image shows how the Entity View is populated for a selected "Cisco" chassis device view under the Chassis Manager node:



The Locater Search

CA Spectrum allows you to find all your modeled chassis devices and their modules in the Chassis node of the Locater tab in the Navigation pane. To find all your modeled chassis devices and their modules, use the following five search criteria available under the Chassis node:

All Chassis

This search criteria finds all your chassis devices that are modeled and arranged under the Chassis Manager node of your landscape. The result of this search lists all of your modeled chassis devices.

All Chassis Managed Devices

This search criteria finds each SNMP capable device model existing on all your modeled chassis devices. The result of this search lists each SNMP capable device model with name of its chassis device.

All Modules

This search criteria finds all your existing modules that are modeled and arranged under each modeled chassis device. The result of this search lists all modeled modules existing in each modeled chassis device.

Managed Devices By Chassis Name

This search criteria finds each SNMP capable device model by the name of its chassis device. The result of this search lists all SNMP capable devices that are mounted on the chassis device you specify.

Modules by Chassis Name

This search criteria finds each modeled module by the name of its chassis device. The result of this search lists all the modeled modules of the chassis device you specify.

Chassis Alarms

When CA Spectrum identifies a device model as a chassis device, the Chassis Fault Domain is associated with that chassis device. This condition correlation domain correlates various alarms on a chassis device and its modules to raise different root cause alarms. For more information about condition correlation domains, see *Condition Correlation User Guide*.

The following alarms are the Chassis Fault Domain alarms:

Chassis Down (0x00010f69)

This alarm is raised when the contact with a chassis device is lost. This alarm is the root cause alarm that suppresses the following alarms that are raised on a chassis device:

- ContactLost_Red (0x00010d35)
- Blade Status Unkown (0x00010f71)
- InferConnectorContactLost_red (0x00010d90)
- Linkdown (0x00010d11)

Blade Status Unkown (0x00010f71)

This alarm is raised when CA Spectrum is not able to contact the chassis on-board agent. This alarm is the root cause alarm that suppresses the following alarms that are raised on a module of a chassis device:

- Catalyst Dev Module Failed (0x011c0488)
- Dev Module Failed (0x00010f70)
- Dev Module Offline (0x00010f86)
- Dev Module Pulled (0x00010f6b)
- Module Offline (0x00010f87)
- Module Pulled (0x00010f6d)

Module Offline (0x00010f87)

This alarm is raised when the state of a Module is reported as "Offline". This alarm is the root cause alarm that suppresses the following alarms that are raised on a module of a chassis device:

- ContactLost_Grey (0x00010d36)
- ContactLost_red (0x000103d5)
- Physical Host Down (0x056e000c)

Module Pulled (0x00010f6d)

This alarm is raised when a Module is pulled out from the chassis. This alarm is the root cause alarm that suppresses the following alarms that are raised on a module of a chassis device:

- ContactLost_Grey (0x00010d36)
- ContactLost_red (0x000103d5)
- Physical Host Down (0x056e000c)

Reconfigure Existing Models with New Certification Support

Existing models do not reevaluate their model class and device type at server startup. Therefore, new mappings that are available in a patch or with an upgrade are not applied to existing models. To pick up the new mappings, reconfigure your existing models.

Follow these steps:

1. Select the device models to update in any OneClick view.
2. Right-click the selected models and select Reconfiguration, Reconfigure Model.

   The selected models are reevaluated. If more current certifications for the models exist, the models are reconfigured.

Interface Modeling

GnSNMPDev creates an interface model for every instance in the MIB-II Interface table. Interface models are instantiated and associated with the device during CA Spectrum modeling. They represent the physical or logical connections on a device.

The device model Interfaces tab in the Component Details panel shows all of the interfaces that CA Spectrum has discovered on a device. The view shows interface status (UP or DOWN) and other information.

Connections between devices can be mapped to the port level, which lets CA Spectrum isolate faults with more granularity. For example, if a port on a device goes down, an alarm is generated on the individual interface model rather than at the device level. Interface model statistics can be polled and logged, letting you monitor and manage device performance with detailed data.

Potential interface model types include the following types:

- Gen_If_Port
- Serial_If_Port
- VLAN_IF
- FrameRelayPort

If Frame Relay Manager is installed and the device supports either of the Frame Relay standard MIBs (RFC1315 or RFC2115), the DLCI circuits are modeled using the DLCI_port model type.

Note: For more information, see the *Standards-Based Protocol Reference Guide*.

If ATM Circuit Manager is installed and the device supports the ATM MIB RFC1695, the ATM logical connections are modeled using the ATMVclLink or ATMVplLink model types.

Note: For more information, see the *ATM Circuit Manager User Guide*.

Application Modeling

When a device is modeled with GnSNMPDev, CA Spectrum creates application models to represent each of the standard (IETF) MIBs that the device supports. Application models are instantiated and are associated with the device during CA Spectrum modeling.

For example, GnSNMPDev intelligence detects that a modeled device performs routing functions (a routing MIB is present). A Routing Application model is created and associated with the device model. Non-routing devices are not burdened with the functionality and attributes that are required to manage routers; each device model carries only the required functionality.

Additional support for standard or proprietary MIBs can be added to the GnSNMPDev model type by customizing the GnSNMPDev management module.

The Locater tab in OneClick lets you search for and access the application models that are associated with a given device model. Several predefined searches are available for application models, but you can also perform a search using custom criteria.

Note: For information about standard MIB applications and accessing their views in OneClick, see the *Standards-Based Protocol Reference Guide* and the *Host System Resources Management User Guide*. For information about creating a search, see the *Administrator Guide*.

Traps, Events, and Alarms

The following table summarizes the trap support that is available with the GnSNMPDev management module for the six generic traps:

Trap Name	OID	Variable Binding	Event Generated	Alarm Generated	Alarm Severity
coldStart	0.0	N/A	0x10306	N/A	N/A
warmStart	1.0	N/A	0x10307	N/A	N/A
linkDown	3.0	1.3.6.1.2.1.2.2.1.1 1.3.6.1.2.1.2.2.1.2 1.3.6.1.2.1.2.2.1.3 1.3.6.1.2.1.2.2.1.7 1.3.6.1.2.1.2.2.1.8	0x220002	0x220001	Yellow alarm on the device (can be configured per port); red alarm on the port
linkUp	2.0	1.3.6.1.2.1.2.2.1.1 1.3.6.1.2.1.2.2.1.2 1.3.6.1.2.1.2.2.1.3 1.3.6.1.2.1.2.2.1.7 1.3.6.1.2.1.2.2.1.8	0x220001	N/A	N/A
authenticationFailure	4.0	N/A	0x1030a	0x1030a	Yellow
egpNeighborLoss	5.0	1.3.6.1.2.1.8.5.1.2	0x1030b	0x1030b	Yellow

In addition, the GnSNMPDev model type supports various RFC and IEEE standard applications traps. This model type also supports any traps that are defined at the global level. You can enhance this support to include other traps and event processing.

Note: For more information about global traps, see the *Event Configuration User Guide*.

More information:

Adding Trap Support (see page 29)
Create Trap Support (see page 77)

Chapter 2: Certification Locater

This section contains the following topics:

Access the Device Certification Database (see page 25)

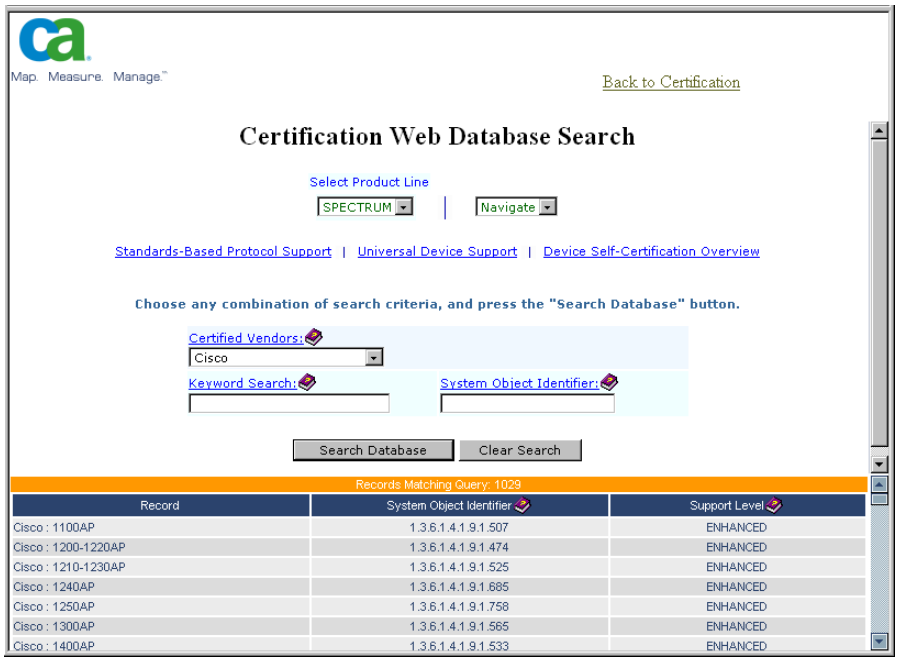
Access the Device Certification Database

An application on the CA Technical Support website lets you search on all CA Spectrum certified devices. You can determine whether CA Spectrum supports a specific device model and filter by firmware version and release. You can also determine whether a device is supported with a Simple certification or an Enhanced certification.

Follow these steps:

1. Navigate to the CA Support Online website.
2. Access the CA Spectrum product page.
3. Click the 'Recommended Reading' link.
4. Click the 'Device and Technology Certification' link.
5. On that page, click the 'Search engine' link.

   The Certification Web Database Search application appears.
6. Select the Spectrum product from the Product Line drop-down list.



7. Complete the following search criteria fields as needed to locate your device:

   Certified Vendors

   Corporations or organizations that manufacture one or more devices that CA Spectrum has certified. A vendor filter limits your search to all devices owned or acquired by the selected vendor.

   Keyword Search

   Searches in the Device Type Name field of each device. A keyword search limits your search to all devices that contain the specific keyword in the Device Type Name field.

   System Object Identifier

   Searches for a System Object Identifier, or a portion of the System Object Identifier. All devices containing the sequence you enter are returned.

   For example, 1.3.6.1.4.1.9.1.685 identifies the Cisco 1240AP device.

   Note: Not all devices have a unique System Object Identifier. In addition, some devices lack a System Object Identifier.

   Support Level

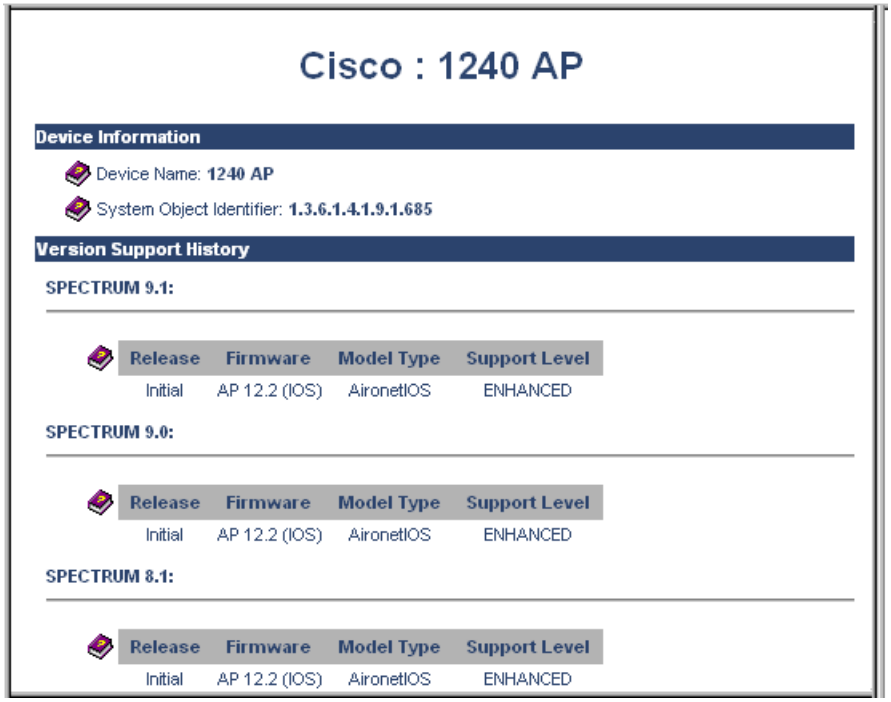
   Indicates the current level of CA Spectrum certification support. Two levels of certification support are available. For more information, see the Overview (see page 9) topic.

8. Click the Search Database button to initiate a search based on your search criteria.

   Results are displayed, one line per device. Details at this level include the device name and model, System Object Identifier and Support Level.

9. Click a specific entry in the results table.

   Detailed information about the selected device appears, as shown:



Chapter 3: Self-Certification

This section contains the following topics:

About Self-Certification (see page 29)
Adding Trap Support (see page 29)
Watches to Monitor and Manage Model Conditions (see page 31)
Lock the Device Type Setting for a Device Model (see page 31)
Lock Device Model Settings (see page 32)

About Self-Certification

You can extend and customize the CA Spectrum simple or enhanced certification support. The following options are available:

- modifying attribute settings
- customizing identification with Device Certification
- managing MIBs and traps with MIB Tools
- developing new certifications

This guide discusses customizing identification with Device Certification, managing MIBs and traps with MIB Tools, and developing new certifications.

Note: For more information about customizing certification support, see the *Event Configuration User Guide*, the *Watches User Guide*, and the *OneClick Customization Guide*.

Adding Trap Support

CA Spectrum uses traps, events, and alarms to notify you about significant occurrences in your infrastructure. These terms apply to specific CA Spectrum entities, as described in the following list:

- Traps are alerts that are sent from SNMP-compliant devices. CA Spectrum receives traps and converts them into events for further processing.
- An alert is an unsolicited message that a managed node on a network sends. The management protocol affects the specific implementation of alerts. In general, CA Spectrum uses SNMP as the management protocol to communicate with devices on a network.

- An event indicates that something significant has occurred. Events are generated for observed behavior within CA Spectrum itself or in the managed environment. CA Spectrum events always occur in relation to a model. When a managed element on the network generates an alert, it is mapped to a CA Spectrum event in the appropriate AlertMap file. The event is then generated with the event code specified in the AlertMap.
- An alarm indicates that a user-actionable, abnormal condition exists on a model. A model usually detects an abnormal condition when an event occurs and the EventDisp file states that an alarm is generated.

By default, the GnSNMPDev model type supports various traps, events, and alarms.

You can also add support for additional traps using the MIB Tools application and the Event Configuration application in OneClick. The high-level process is as follows:

1. Identify the MIB that contains the desired trap definitions.
2. In MIB Tools, import the MIB into the MIB Tools database.
3. Map the traps to events using MIB Tools. Specify the events that generate alarms and the alarm severity.

   MIB Tools automatically creates and installs the appropriate event and alarm support files.
4. Launch Event Configuration directly from MIB Tools:
   a. In the Trap Support table, select the mapped traps whose events and alarms you want to configure.
   b. Edit traps for selected items in the trap support table.
5. Complete the configuration of the events and alarms in Event Configuration.

   For example, specify the symptoms, probable causes, and recommended actions for each alarm. The corresponding messages are displayed in OneClick when the alarms are generated.

   You can also add optional event processing for one or more events. For example, set up logging and create event rules that determine whether the event clears an alarm or generates another event.

   In addition, you can customize the default event message that is displayed in OneClick when the events are generated.

Note: For more information, see the *Event Configuration User Guide*.

More information:

Traps, Events, and Alarms (see page 23)

Watches to Monitor and Manage Model Conditions

You can create one or more watches for a particular model. A *watch* is a mechanism for adding thresholds for model attributes. Watches let you monitor network elements, such as routers, with a high level of detail. They also provide current data that can be used with other CA Spectrum tools in network analysis.

Set up a watch to monitor and analyze the changing internal and external attribute values of a model. Watches can include expressions that incorporate one or more attribute values. These attribute values, or an expression that is derived from these values, can then be measured against a defined threshold value. CA Spectrum evaluates the attribute values defined in a watch by polling the attributes when they are updated or when the watch value is read. Results can be used to generate events and alarms. Results can be logged for historical tracking and report information or sent to script files.

Keep in mind that watches can have an impact on network traffic and system resources. Delete watches that are no longer useful.

Note: For more information, see the *Watches User Guide*.

Lock the Device Type Setting for a Device Model

You can lock the device type setting for a model so that the type is not reevaluated when you reconfigure a device. By default, the device type setting is not locked.

Follow these steps:

1. Locate the device model in the Topology tab of the Contents panel.
2. Select the device model whose device type setting you want to lock.
3. Click the Information tab in the Component Detail panel.
4. Expand the CA Spectrum Modeling Information subview.
5. Click 'set' in the Lock Device Type field, and select Yes.

   The device type setting is locked for the selected device model.

More information:

Modify Device Certification Entries (see page 38)
Custom Device Type Mappings (see page 37)

Lock Device Model Settings

You can lock settings for a device model. Locked settings are not reevaluated when you reconfigure a device. Both the device type setting and the model class setting for a device can be locked. By default, neither setting is locked.

Follow these steps:

1. Locate device models in the Topology tab of the Contents panel.
2. Select the device model whose device type or model class setting you want to lock.
3. Click the Information tab in the Component Detail panel.
4. Expand the CA Spectrum Modeling Information subview.
5. Take one or both of the following steps:
   - Click 'set' in the Lock Device Type field, and select Yes.
   - Click 'set' in the Lock Model Class field, and select Yes.

   The setting is locked for the selected device model.

More information:

Modify Device Certification Entries (see page 38)
Custom Device Type Mappings (see page 37)

Chapter 4: Customizing Identification with Device Certification

This section contains the following topics:

Device Certification in OneClick (see page 33)
Open the Device Certification Dialog (see page 34)
About the Device Certification Dialog (see page 34)
Device Certification Table (see page 35)
Search Device Certification Mappings Using Filters (see page 36)
Device Certification Changes (see page 37)
Device Mappings (see page 37)
Change the Model Type for a Single Device Type (see page 48)
Device Certification and Fault-Tolerant Environments (see page 49)

Device Certification in OneClick

The Device Certification component of OneClick lets you view, create, and edit Device Certification entries. CA Spectrum maps the System Object ID to the device type, model class, and model type. Device certification mappings appear in the Device Certification dialog.

Device certification entries let CA Spectrum initialize the device type, model type, and model class attributes on models during Discovery, Modeling, and device creation. You can create Device Certification entries for devices not directly supported in OneClick.

You can search, filter, and report on device models using the following attributes from the Device Certification list:

- Device Type attribute (0x23000e)
- Model Class attribute (0x11ee8)
- Model Type attributes (0x10000 for Modeltype_Name and 0x10001 for Modeltype_Handle)

These attributes provide a fine level of granularity when managing your network infrastructure.

Device Certification supports a distributed SpectroSERVER environment. Consistent device model identification occurs across a distributed deployment.

Note: To access and edit the device certification entries, log in as an administrator with read and write permissions.

Open the Device Certification Dialog

You can open the Device Certification dialog by taking one of the following steps:

- Open the Device Certification dialog from OneClick by selecting Tools, Utilities, Device Certification.

  The Device Certification dialog opens, displaying device type mappings for all modeled devices.

- Open the Device Certification dialog within the context of a device model. Select the model in the Explorer tab or the List tab of the Navigation panel, or in the Topology tab of the Contents panel. Right-click the selected device and select Utilities, Device Certification.

  The Device Certification dialog opens. The entry for the selected device type is highlighted.

More information:

Resolve Device Certification Mapping Conflicts (see page 46)

About the Device Certification Dialog

The Device Certification dialog lets you maintain a custom list of system object identifiers and their corresponding device type name, model type, and model class. When you create or modify one of these entries, the corresponding attribute for all device models with the given system object ID is set to your customized value. This setting is applied to both existing and future device models. Used with the GnSNMPDev model type, this feature lets you model and monitor any SNMP-compliant device in the network. Devices that lack a specific CA Spectrum management module can also be modeled.

The Identification list in the Device Certification dialog also contains unregistered devices. An "unregistered" device has been modeled using Discovery or Model by IP. Such a device has system object IDs but lacks a device type name, model type, or model class.

You can use the unregistered devices in the Identification list to set up entries for all devices that are modeled with GnSNMPDev. Instead of trying to determine which devices use the GnSNMPDev model type, first model the devices. Once the devices are modeled, their system object IDs are added to the Identification list. You can then filter and sort the list and specify device type names for the unregistered devices that are modeled with GnSNMPDev.

The mappings in the Device Certification dialog are preserved during upgrades and database migrations.

You can open the Device Certification dialog from the OneClick Tools menu or from a device model context:

- Select Tools, Utilities, Device Certification.

  The Device Certification dialog displays device type mappings for all modeled devices.

- Right-click a device model in OneClick. Find the model in the Explorer tab or the List tab of the Navigation panel, or in the Topology tab of the Contents panel. Select Utilities, Device Certification.

  The Device Certification dialog highlights the entry for the selected device type.

More information:

Map Unregistered Devices (see page 43)
Overview (see page 9)

Device Certification Table

The Device Certification dialog lists the mappings of device type name to System Object Identifier (sysObjectID or system OID), model classes, and model types. This list includes all standard CA Spectrum predefined mappings, all user-defined mappings, and any unregistered mappings. The list *does not* include mappings that have specialized device type name handling.

The Device Certification table displays the following information about each mapping:

Vendor Name

Displays the name of the company that manufactures the device, such as Cisco Systems.

System Object ID

Displays the MIB II sysObjectID entry that was retrieved from the device.

Device Type Name

Displays the Device Type value that is mapped to the associated system OID.

Model Type

Identifies the name of the specific model type. If the system OID is supported with a simple certification (the system OID is *not* associated with a specific model type), displays “GnSNMPDev”. By default, this column is not visible.

Model Class

Identifies the model class (such as Router, Switch-Router, or Port) for the device model. If no model class is mapped to the system OID, "Auto" is displayed.

Modification

Identifies the mappings that have been modified in the current DC session.

Support Level

Identifies whether the device for the system OID has a simple or enhanced certification (MM). If the system OID is modeled with the GnSNMPDev model type, the support level is "Simple." If the system OID has a specific certification, the support level is "Enhanced."

Author

Identifies the user who created the mapping. This column is hidden by default.

You can modify the table display using the standard OneClick table preferences and column sorting methods. You can export the data in the Device Certification table to a file in either a comma-separated (.CSV), tab-delimited (.txt), or web page (.HTML) format.

Note: For more information, see the *Operator Guide*.

Search Device Certification Mappings Using Filters

You can search for specific text or numeric strings in the Device Certification table by typing them in the Filter field. As you type in the Filter field, only the mappings that contain matching character strings appear in the table.

Note: The Filter field searches only the *visible* columns.

Use this feature to search for the following attributes:

- **Vendor Name:** Type the name of a specific vendor to display all supported system OIDs for that vendor. This information can help you determine whether CA Spectrum supports a specific model.
- **Custom mappings:** Enter "custom" in the Filter field to view only the mappings that you or someone else has modified using Device Certification.
- **Unregistered devices:** Enter "unregistered" in the Filter field to view only mappings of unregistered devices in the table.

More information:

Map Unregistered Devices (see page 43)

Device Certification Changes

Once you have created or modified a mapping and have applied the changes, all device models in your distributed SpectroSERVER (DSS) environment are updated if they have Device Certification mapping changes. The updated device type names, model classes, and model types appear in the Topology views, the Navigation panel, and the List views for all landscapes.

The CA Spectrum modeling catalog is also updated so that all future device models with this system OID are assigned the corresponding device type, model class, and model type values.

More information:

Device Certification Changes are Not Saved (see page 47)

Device Mappings

Device Certification categorizes device mappings into the following types:

- **CA mappings:** The predefined mappings that are included with CA Spectrum.
- **Custom mappings:** The mappings that are created or customized using the Device Certification utility.
- **Unregistered mappings:** Entries that do not have a mapping.

Note: The mapping type appears in the Author column in the Device Certification dialog.

Custom Device Type Mappings

You can customize any mapping, or you can create new mappings to improve any of the following administrative settings:

- Modify device type names to be more descriptive.
- Modify model classes to more accurately reflect the type of a device.
- Modify the model type to acquire functionality that is associated with a more appropriate model type.
- Add new mappings to accommodate devices.
- Assign device names to unregistered devices that CA Spectrum has identified on your network.

More information:

Lock Device Model Settings (see page 32)
Lock the Device Type Setting for a Device Model (see page 31)

Modify Device Certification Entries

You can modify the device type name of an existing Device Certification mapping. You can also map the model type name and model class of a device.

If you modify the device type name of an existing Device Certification mapping, or if you map the device OID to the appropriate model type name or model class, you create a custom mapping. Custom mappings override original mappings.

However, the original mapping remains intact on the system. If you delete a custom mapping with a modified device type name, model type name, or model class value, the original mapping reappears in the table.

Follow these steps:

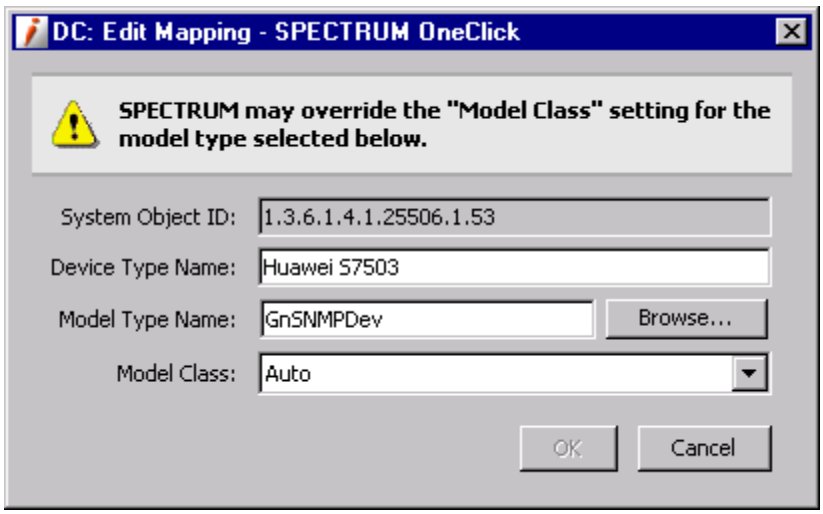
1. Open the Device Certification dialog (see page 34).

   The Device Certification dialog displays the device certification mappings on the Identification tab.

2. Select an entry to modify, and click  (Edit button).

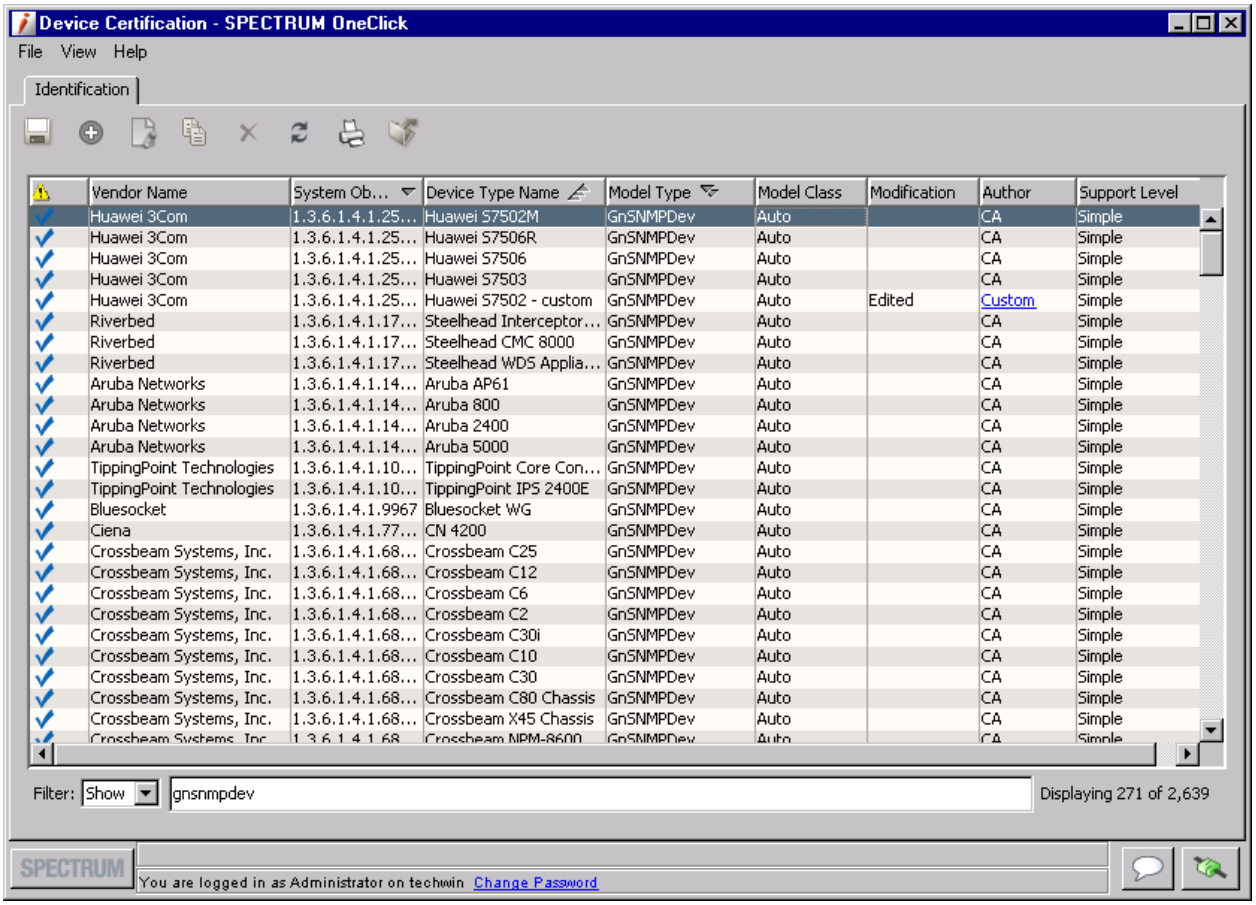
   The DC: Edit Mapping dialog opens.

   Note: For some models, CA Spectrum overrides your custom settings. In these cases, a warning message appears.



3. Edit the following fields, and click OK.
   - Device Type Name
   - Model Type Name
   - Model Class

   Your changes to the selected model appear in the table on the Identification tab.



4. Click Save.

   Your changes are saved and are immediately applied to all device models. When complete, the DC: Operation Results dialog displays the results, either successful or unsuccessful, for each landscape in a DSS environment.

5. Click Close.

   Your selected Device Certification mapping is updated.

More information:

Lock Device Model Settings (see page 32)
Lock the Device Type Setting for a Device Model (see page 31)
View Default Values Masked by Custom Device Certifications (see page 40)

View Default Values Masked by Custom Device Certifications

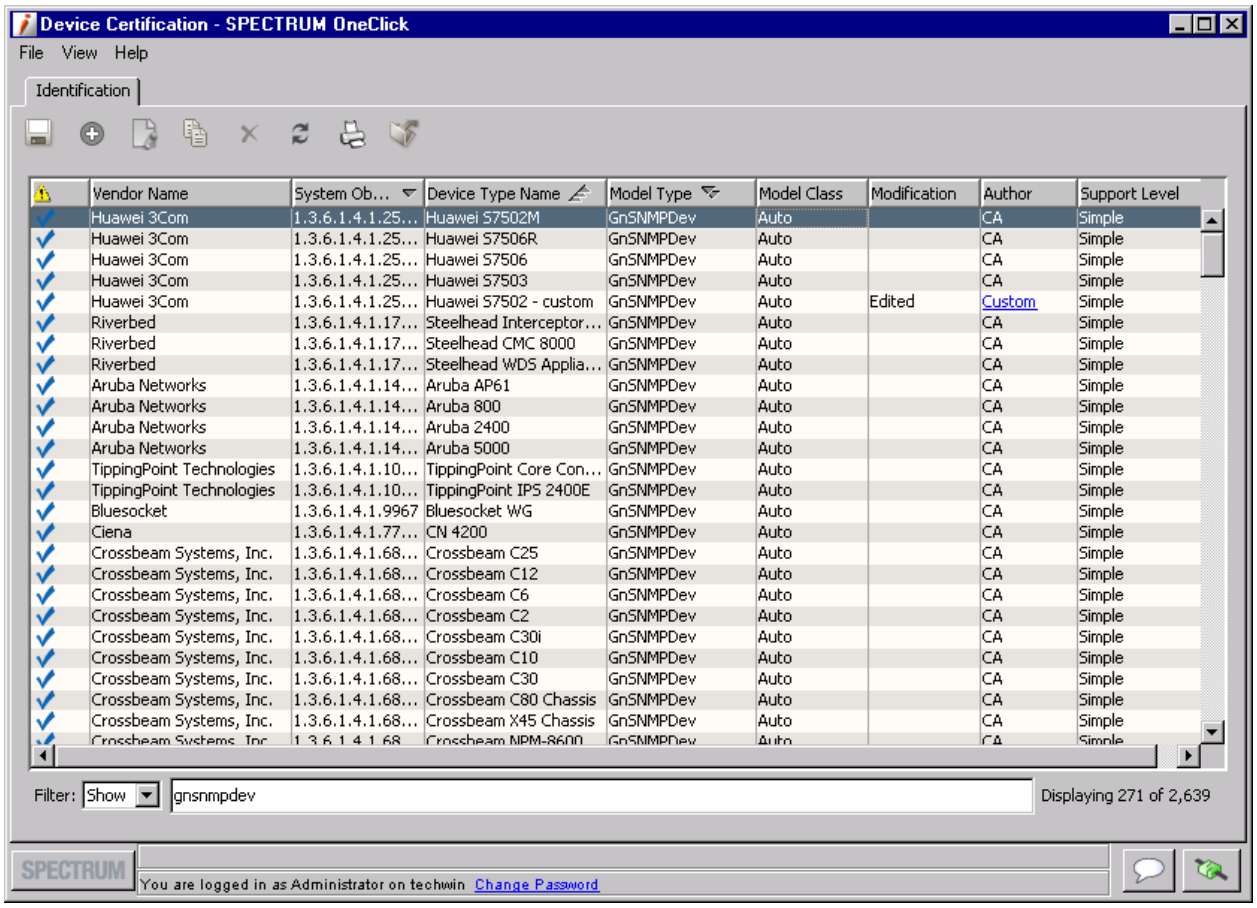
Default device certifications are set up during initial CA Spectrum modeling. When you customize a device certification mapping, your values mask the default values. After customizing your mapping values, you can view the default values that your custom mapping is obscuring. Viewing this information can be useful when determining whether your custom values are still required.

Follow these steps:

1. Open the Device Certification dialog (see page 34).

   The Device Certification dialog displays the device certification mappings on the Identification tab.

2. Locate the custom device certification, and click the 'Custom' link.



The DC: Custom Mapping Details dialog opens. This dialog displays the custom values and the default values for each modified device certification mapping.

More information:

Modify Device Certification Entries (see page 38)

Copy Device Certification Mappings to Create New Mappings

You can create a new device certification mapping by copying an existing device certification mapping. Using the existing mapping can be more efficient when your new mapping is very similar, because the System Object ID, device type, model class, and model type values are prefilled.

Follow these steps:

1. Open the Device Certification dialog (see page 34).

   The Device Certification dialog displays the device certification mappings on the Identification tab.

2. Select the entry to copy and click  (Copy button).

   The DC: Copy Mapping dialog opens.

   Note: For some models, CA Spectrum overrides your custom settings. In these cases, a warning message appears.

3. Edit the following fields, as needed, and click OK.

   - System Object ID
   - Device Type Name
   - Model Type Name
   - Model Class

   Your new device certification mapping appears in the table on the Identification tab. The Modification column specifies "New," and the Author column specifies "Custom."

4. Click Save.

   Your changes are saved and are immediately applied to all device models. When complete, the DC: Operation Results dialog displays the results, either successful or unsuccessful, for each landscape in a DSS environment.

5. Click Close.

   Your new device certification mapping is created and is added to the Device Certification table.

More information:

About the Device Certification Dialog (see page 34)

Map Unregistered Devices

Unregistered devices lack a matching device type, model class, or model type entry. Any unregistered devices appear in bold in the Device Certification dialog, and "Unregistered" appears in the Author column.

To map unregistered devices, know the system OID of the unregistered devices on your network.

Follow these steps:

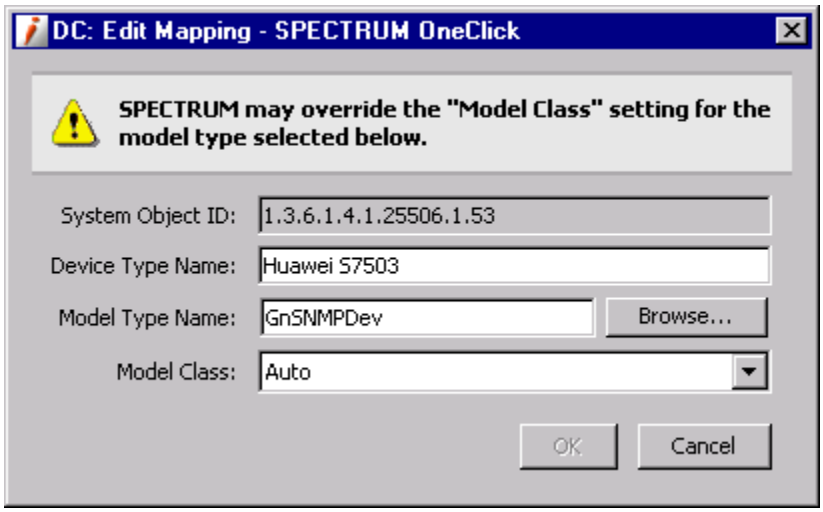
1. Open the Device Certification dialog (see page 34).

   The Device Certification dialog displays the device certification mappings on the Identification tab.

2. Select the unregistered device entry to be mapped, and click  (Edit button).

   The DC: Edit Mapping dialog opens.

   Note: For some models, CA Spectrum overrides your custom settings. In these cases, a warning message appears.



3. Edit the following fields, as needed, and click OK.
   - Device Type Name
   - Model Type Name
   - Model Class

   Your changes to the selected model appear in the table on the Identification tab. "Custom" appears in the Author column, and "P" appears in the Modification column.

4. Click Save.

   Your changes are saved and are immediately applied to all device models. When complete, the DC: Operation Results dialog displays the results, either successful or unsuccessful, for each landscape in a DSS environment.

5. Click Close.

   Your unregistered device is now mapped to the correct device certification values.

More information:

Search Device Certification Mappings Using Filters (see page 36)

OneClick Views

A device that is modeled with the GnSNMPDev model type offers access to all OneClick views, such as Information, Performance, and Alarms.

Note: For more information about OneClick views, see the *Operator Guide.*

Delete Custom Device Certification Mappings

You can delete custom mappings from the Device Certification table. However, default CA Technologies mappings cannot be deleted. When you delete a custom mapping that overrides a default mapping, the default mapping displays after the delete operation completes.

Follow these steps:

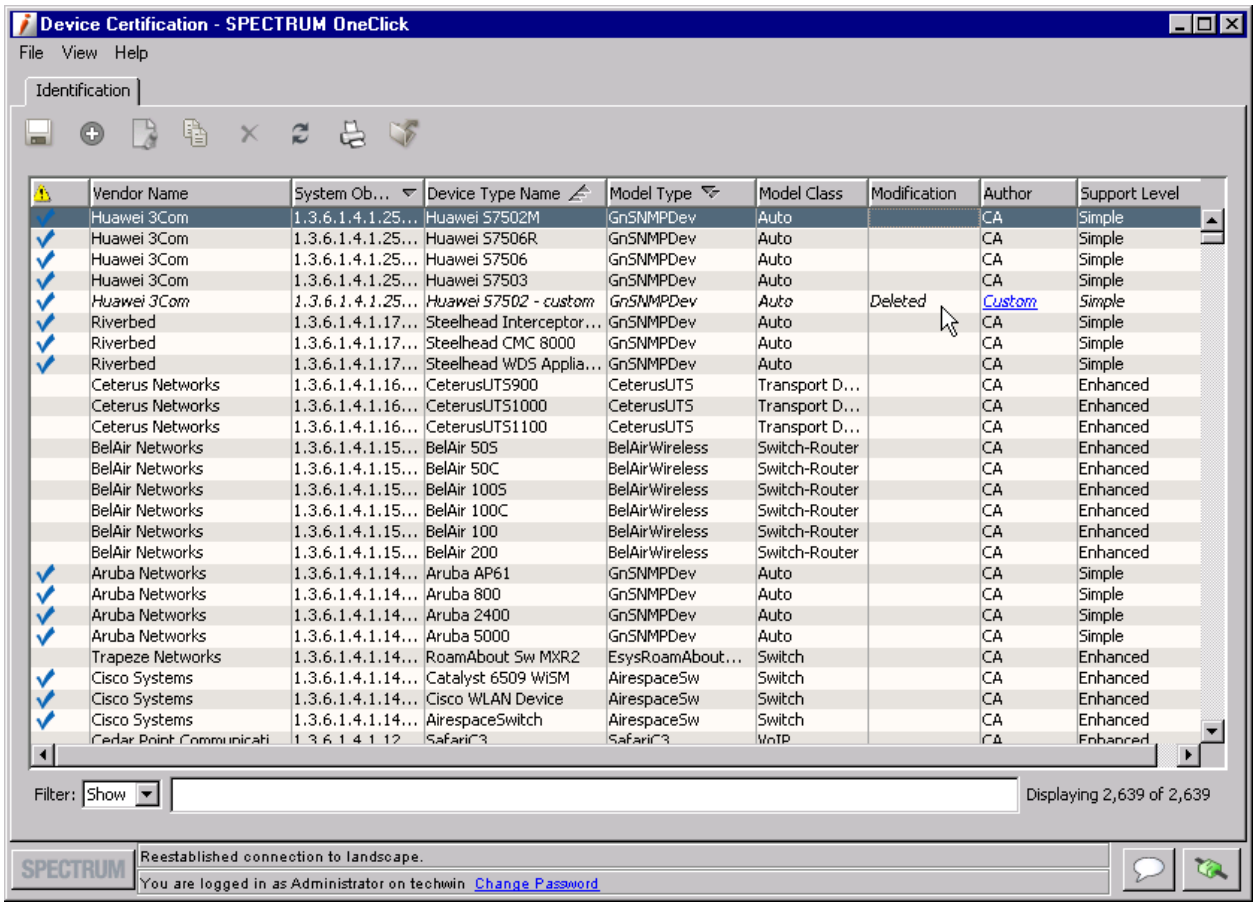
1. Open the Device Certification dialog (see page 34).

   The Device Certification dialog displays the device certification mappings on the Identification tab.

2. Select the entry to delete and click  (Delete).

   Note: The Delete button is not available if the selected entry cannot be removed. For example, a default CA Technologies Device Certification cannot be deleted.

The entry is flagged for removal from the list.



3. Click Save.

   Your changes are saved and are immediately applied to all device models. When complete, the DC: Operation Results dialog displays the results, either successful or unsuccessful, for each landscape in a DSS environment.

4. Click Close.

   Your selected Device Certification mapping is deleted.

More information:

Custom Device Type Mappings (see page 37)

Distributed SpectroSERVER Support

The Device Certification utility supports a distributed SpectroSERVER (DSS) environment. Device Certification detects multiple primary SpectroSERVERs in a CA Spectrum environment and alerts you when it cannot communicate with a SpectroSERVER. Device Certification queries all SpectroSERVERs for Device Certification table entries. Conflicts among SpectroSERVERs for user-specified mappings are detected.

Note: When you open the Device Certification utility in a DSS environment, a warning message appears if any of the primary SpectroSERVERs are down. Any unavailable SpectroSERVERs do not receive the Device Certification mapping changes that you make. Custom Device Certification mapping conflicts occur after all SpectroSERVERs are back online.

Resolve Device Certification Mapping Conflicts

Conflicts in Device Certification mappings can occur in a environment when a sysObjectID has more than one device type, model class, or model name defined. Conflicts usually happen when:

- **User customizations are mismatched across SpectroSERVERs**—This situation can occur when one or more SpectroSERVERs are down prior to starting Device Certification. This situation can also occur when one or more SpectroSERVERs go down after starting Device Certification but before applying changes to custom mappings.
- **Predefined mappings are mismatched across SpectroSERVERs**—This situation occurs when the device certification mappings provided by CA Spectrum are updated on some SpectroSERVERs but not others. For example, bringing a new SpectroSERVER online that has a more recent release of CA Spectrum can lead to differences in the predefined mappings between SpectroSERVERs.

For mismatched predefined mappings, verify the software installation on all SpectroSERVERs so that they use the same device certification table. You can resolve conflicts with customized mappings using the Device Certification dialog.

Follow these steps:

1. Open the Device Certification dialog (see page 34).

   If conflicting Device Certification mappings are detected when you start the Device Certification utility, a Conflicts Encountered dialog opens.

   Important! Resolve the conflicts so that Device Certification can open.

2. Click the Resolve Conflicts button.

   The Resolve Conflicts dialog opens.

3. Select the appropriate name from the Device Type Name drop-down list for each System Object ID, then click OK.

   The Device Certification mapping conflicts are resolved. The Device Certification dialog displays the device certification mappings on the Identification tab.

Note: If the Device Certification utility identifies a Device Certification entry that is present on some, but not all, of the servers in the DSS environment, the entry is automatically applied to all servers that lack it. This condition is different from a conflicting entry condition.

More information:

Open the Device Certification Dialog (see page 34)

Device Certification Changes are Not Saved

Symptom:

I updated my Device Certification mappings, but some of my device models were not updated when I clicked Save. Now the Save button is disabled. Why were some devices not updated, and how can I apply my changes?

Solution:

CA Spectrum is unable to successfully save the updated Device Certification mappings on the first save attempt in the following cases:

- **A server in a distributed SpectroSERVER (DSS) environment cannot be contacted and does not receive the mapping update.**

  In this case, CA Spectrum warns you that one or more servers were down before the mappings were applied, or that some of the mappings were not applied on one or more servers.

  To resolve this problem, take the following steps:

  1. Close the Device Certification dialog.
  2. Resolve the communication issue with the server.
  3. Open the Device Certification dialog (see page 34).

     The Device Certification dialog notifies you that conflicts exist and you must resolve them.
  4. Resolve all conflicts, then click Apply.

     The server synchronizes with your Device Certification mappings and reapplies them to all affected models.

- **Updated mappings are applied to all SpectroSERVERs, but some device models did not reevaluate their Device Certifications.**

  This situation can occur when the model classes or device types are locked. Network failure, stopping a server, or losing contact with the Device Certification client during an update can also cause this scenario. In this case, the SpectroSERVERs are properly updated—the *device models* did not update—so the Device Certification dialog does not notify you about conflicts.

  To force individual device models to reevaluate their Device Certification mappings, take the following steps:

  1. Use a Locater search to find all device models that did not update.

     Note: You can check the Device Type Name or Model Class columns for device models that did not update.

  2. Verify that the device types and model classes are not locked (see page 32, see page 31).

  3. Select all affected device models and reconfigure the models (see page 21).

More information:

Device Certification Changes (see page 37)

Change the Model Type for a Single Device Type

You can use the NewMM.pl post-installation script to automatically change the model type for a single device type. Many key attributes, relationships, and other elements are preserved.

This procedure changes the model type for all models with the specified system Object ID *and* the specified starting model type.

Important! Do not perform this procedure until you modify the model type mapping for the device type in the Device Certification utility. Otherwise, your changes are not communicated to the SpectroSERVER database and you see unexpected alarms.

Follow these steps:

1. Verify that the SpectroSERVER is running.
2. Run the following command from the <*$SPECROOT*>/Install-Tools/PostInstall/ directory:

   ```
   NewMM.pl -m
   ```

Note: On Windows, run all necessary scripts from a bash shell. They do not run as expected from a DOS command prompt.

3. Enter the host name or IP address of the VNM and press Enter.
4. Enter the SpectroSERVER landscape handle when prompted.
5. Enter the system Object ID for the model when prompted.
6. Enter the current model type of the model when prompted.
7. Enter the new model type when prompted.

   The model type is changed.

   The log file, NewMM_Log_<*DATE*>, is created in the <*$SPECROOT*>/Install-Tools/PostInstall/ directory.

   Note: Verify that the model type converted successfully by checking the log file, NewMM_Log_<*DATE*>.

Device Certification and Fault-Tolerant Environments

If you are working in a fault-tolerant environment, Device Certification differentiates between a primary and a backup server. For Device Certification to operate, it must be able to connect to the primary SpectroSERVER. The application cannot connect to the backup server.

The backup SpectroSERVER obtains the Device Identifier List update and device model updates from the primary SpectroSERVER during the Online Backup procedure.

Note: For more information, see the *Distributed SpectroSERVER Administrator Guide*.

Chapter 5: Managing MIBs and Traps With MIB Tools

This section contains the following topics:

The MIB Tools Utility (see page 51)
OneClick MIB Tools Overview (see page 53)
MIB Tools User Interface (see page 55)
Import and Export MIBs (see page 61)
Query (GET_NEXT), GET, and SET Requests (see page 67)
Custom Vendor Folders (see page 72)
Contact a Device Using MIB Tools (see page 74)
Search for a MIB (see page 75)
Trap Support (see page 76)
MIB Tools Support for Multiple SpectroSERVERs (see page 82)
Trap Disposition Conflicts (see page 85)

The MIB Tools Utility

The MIB Tools utility lets you compile, import, and browse Management Information Bases (MIBs). In addition, this utility can issue SNMP requests to network elements and can customize the mapping of MIB objects and traps in CA Spectrum. Use the MIB Tools to create, customize, and troubleshoot network element management in CA Spectrum.

SNMP and MIBs form the structure of network element management in CA Spectrum. CA Spectrum communicates with modeled network entities using SNMP. A MIB is a type of network device database that represents a device as a hierarchical collection of objects. A MIB object represents an individual unit of information in a MIB, such as device uptime. MIBs themselves are text files with special syntax.

MIB Tools include two self-certification tools. First, MIB Tools lets you map MIB objects to attributes in the CA Spectrum database. You can create OneClick views to display the values of those MIB objects, create Watches on the attributes, and set thresholds to send alarms. Second, the MIB Tools utility lets you add support for traps that your devices send.

CA Spectrum complies with the following RFCs regarding MIBs and SNMP:

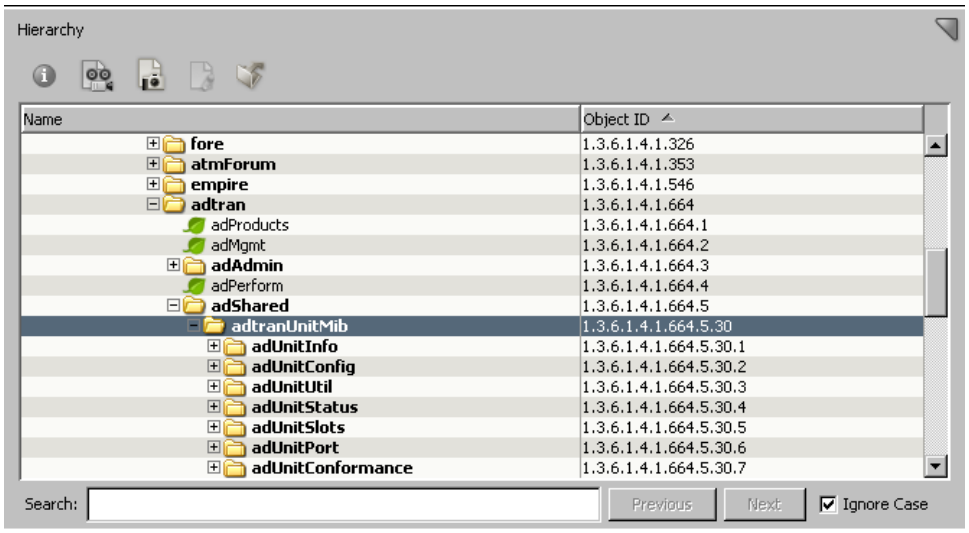
- RFC 1155: Structure and Identification of Management Information for TCP/IP-based Internets
- RFC 1157: A Simple Network Management Protocol
- RFC 1213: Management Information Base for Network Management of TCP/IP-based Internets

How a MIB Is Organized

The International Standards Organization (ISO) supplies a standard tree format for the organization of network device management information. This tree structure branches out into subtrees that are organized into branches (groups of related information) and leaves (the individual pieces of information, or objects).

Each layer of this tree is numerically encoded. A unique number, an Object Identifier (OID), identifies each group and object. This identifier lets an SNMP agent locate the object in a device MIB. An ASCII name is also assigned to each branch or OID to identify management objects.

The following image illustrates MIB objects in the MIB Tools Hierarchy with the name and OID. MIB Tools uses folders, acorns, and leaf icons to show branches, traps, and objects within a MIB. Each folder in the display indicates that more objects are contained in that level of the tree structure.



MIB Tools Database

MIB Tools maintains a MIB database on the OneClick web server. The default database consists of standard and proprietary MIBs. You can add MIBs to this database by importing them. CA Spectrum does not use the MIB Tools database.

OneClick MIB Tools Overview

MIB Tools is a multifunctional MIB utility. Use it to browse MIBs, issue SNMP requests to network elements, import MIBs, and add MIB object and trap mapping support to CA Spectrum. You can use MIB Tools to retrieve supported information directly from a given device to aid in troubleshooting and managing that device. You can customize your CA Spectrum network management environment by using MIB Tools to import the MIBs of network elements that are not yet supported in CA Spectrum.

The MIB Tools utility lets you complete the following tasks:

- Compile and import MIBs into the MIB Tools database.
- Browse MIBs for details of MIB objects and traps.
- Directly query and set values for MIB objects of network elements.
- Export MIB query result values for use in troubleshooting and creating simulations.

  You can export the data that is displayed in MIB Tools into several different file formats. You can export data from the Results, Attribute Support, and Trap Support tables.
- Create custom network element support in CA Spectrum by mapping new traps and MIB objects.
- Delete custom MIBs from MIB Tools database. Standard or proprietary MIBs cannot be deleted.

Start MIB Tools

You can start the MIB Tools utility from the Tools menu. Or you can start it within the context of a selected device model.

To open the MIB Tools utility without the context of a specific device model, click Tools, Utilities, MIB Tools without selecting a device model. The MIB Tools dialog shows the progress of retrieving and loading the MIB Tools database.

You can also start MIB Tools in the context of a specific device model. This method lets you communicate with the device and perform SNMP queries on the model, whose contact criteria are automatically displayed.

Follow these steps:

1. In the OneClick Console, locate the device model you want to investigate with MIB Tools.

   Locate a model in either the Explorer tab of the Navigation Panel or the Topology tab of the Contents panel.

2. Right-click the device model and click Utilities, MIB Tools, or click Tools, Utilities, MIB Tools from the main menu.

   The MIB Tools utility opens.

   The Contact Criteria display the SNMP contact information for the selected device.

   MIB Tools attempts to contact the device.

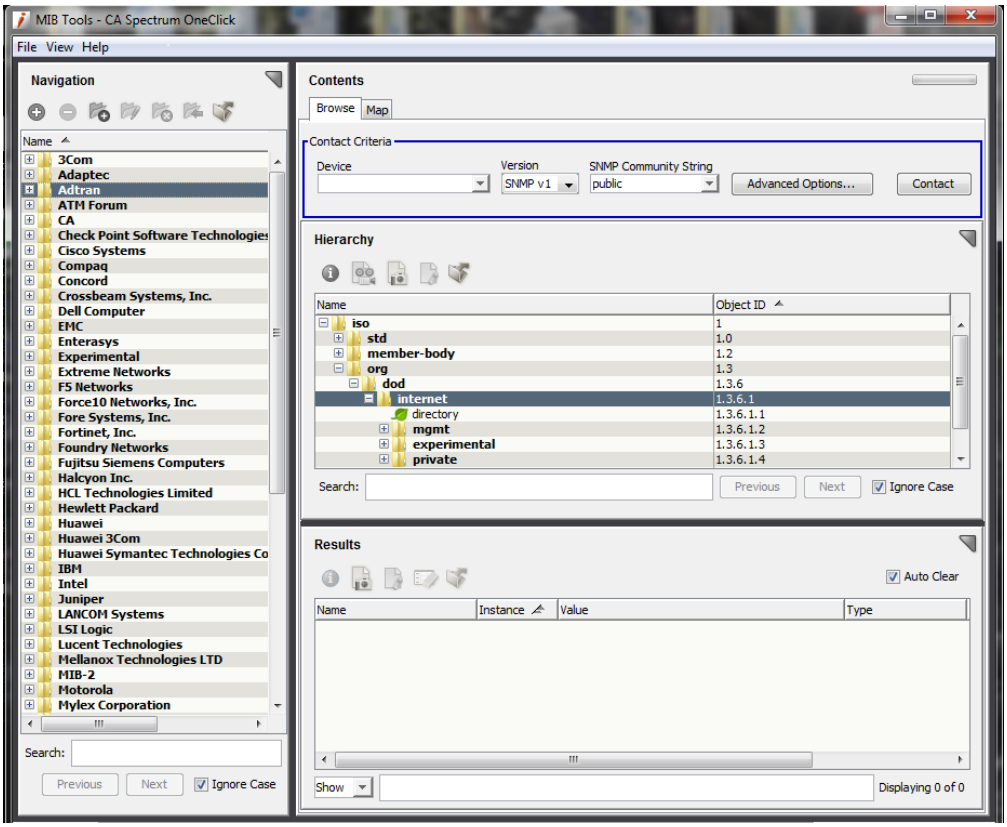
   If the attempt fails, an error message appears.

   If the attempt succeeds, the Contact Status indicator turns green.

   A status dialog shows the progress of retrieving and loading the MIB Tools database.

MIB Tools User Interface

The MIB Tools user interface has two panes that let you find MIBs and view MIB information: the Navigation pane and the Contents pane.



Locate and select MIBs in the Navigation pane. You can view a list of the compiled MIBs in the MIB Tools database. By default, MIBs are organized by vendor and are displayed alphabetically. Sorting is supported on table columns. You can also delete custom MIBs that you select in the Navigation pane.

The Contents panel has two tabs:

Browse tab

Lets you browse MIBs in the MIB Tools database and query devices on your network to obtain or set MIB object values. The Browse tab has three main sections: Contact Criteria, Hierarchy, and Results.

Perform the following tasks from the Browse tab:

- Contact a device.
- View the name, object ID, or access type of a MIB object in the Hierarchy section.
- View the results of a query in the Results section.

Map tab

Identifies the objects and traps for a selected MIB that CA Spectrum supports. You can see the CA Spectrum Attribute ID for an object and the Event code for a trap. Unsupported objects lack an Attribute ID or Event code.

Perform the following tasks from the Map tab:

- Create attributes in the CA Spectrum database from MIB objects.
- Map traps in CA Spectrum.
- Identify the MIB objects that have been previously mapped to attributes in the CA Spectrum database using MIB Tools.
- Identify all trap support in CA Spectrum.

MIB Tree Hierarchy Table

When you select a MIB in the Navigation panel, its details appear in the Hierarchy tree table in the Browse tab of the Contents panel.

The default view of the Hierarchy tree table displays the entire MIB Tools database starting at the iso branch.

Browse MIBs

You can browse the MIB Tools database and view detailed information for each group, object, and trap. For trap objects, you can view variable binding details. To browse an individual MIB, select the MIB in the Navigation panel and use the Hierarchy tree table to navigate the MIB folders, groups, objects, and traps.

Search MIBs

You can search the Hierarchy tree table for a text string by entering it into the Search field. The Next and Previous buttons let you scroll through each successive instance of the search string.

The Hierarchy tree table displays the following information:

Name

Displays the name of the MIB object.

Object ID

Displays the Object ID of the MIB object.

Access

(Hidden by default) Displays the access type for the object. The type is read-only, read/write, read-create, or not-accessible.

Note: For more information, see the *Operator Guide*.

The Hierarchy toolbar provides the following functionality:

Button	Description
	**Information button:** Opens a dialog with details about a selected item. This information is taken directly from the MIB.
	**Query button:** Retrieves a subtree of management values using SNMP GET_NEXT requests. All of the objects returned have an OID that is prefixed by the OID of the branch that you select for the query.
	**GET button:** Performs an SNMP GET of a selected scalar object. If a scalar object is not selected, lets you build an SNMP GET or SNMP GET_NEXT request.
	**SET button:** Sets the value of the selected MIB object on a specific device.
	**Export button:** Exports the table contents to an external file.

More information:

Query an Object (see page 68)
Query a Subtree of Objects (see page 67)
Set an Object (see page 68)

Attribute Support Table

When you select a MIB in the Navigation panel, its attribute support details appear in the Attribute Support table. Find this table in the Map tab of the Contents panel. The Attribute Support table displays information for MIB objects and CA Spectrum attribute support.

Note: The table does not reflect the CA Spectrum attribute support from the CA Spectrum model type catalog. Only the CA Spectrum attribute support that is created using MIB Tools is included.

The Attribute Support table displays the following information:

Name

Is the name of the object in the MIB.

Object ID

Is the object ID in the MIB.

Attribute ID

Specifies the attribute ID value. If the object is not supported in CA Spectrum, this field is blank. The word 'Conflict' can appear to indicate a conflict with the assigned attribute ID for the object in a DSS environment.

Landscapes

Indicates whether the attribute is supported on some, all, or none of your landscapes.

Needs Update

Indicates whether the attribute requires an update. For example, if the enumerations in the MIB do not match entries in the CA Spectrum database, the entry needs an update. A checkmark indicates that an update is needed.

The toolbar in the Attribute Support table includes the following functionality:

Button	Description
	**Information button:** Opens a dialog with details about a selected item.

Button	Description
	**Create Attributes button:** Creates CA Spectrum Attribute IDs for the objects that are selected and that lack support in CA Spectrum. If you do not select any objects in the Attribute Support table, and no attributes are currently mapped, creates attributes for *all* objects in the table. This button is disabled if the selected entry is already mapped, or if no items are selected and any of the entries are already mapped.
	**Export button:** Exports the table contents to an external file.

Trap Support Table

When you select a MIB in the Navigation panel, its trap support details appear in the Trap Support table in the Map tab of the Contents panel.

The Trap Support table lets you view the traps defined in the MIB selected in the Navigation panel and all default and custom CA Spectrum event codes mapped to them.

The Trap Support table displays the following trap and CA Spectrum event information:

Name

Specifies the name of the trap in the MIB.

Object ID

Specifies the trap object ID in the MIB.

Event Code

Specifies the event code. If the trap is available for only select CA Spectrum models, the event code appears as ‘Partial.’ If the trap has different event codes mapped on different SpectroSERVERs in a DSS environment, the event code appears as ‘Conflict.’

Landscapes

Indicates if the attribute is supported on some, all, or none of your landscapes.

Event Type

Indicates whether the trap has a default mapping, a custom mapping, or both. Default mappings show ‘CA’ in this column; custom mappings show ‘Custom.’ However, if a custom trap mapping is obscuring, or shadowing, a default trap mapping, this column shows a ‘Custom’ link. Click 'Custom' to see the disposition details of the custom mappings.

The toolbar in the Trap Support table includes the following functionality:

Button	Description
	**Information button:** Opens a dialog with details about a selected item.
	**Map Traps button:** Maps traps for the selected item or items. If no objects in the Trap Support table are selected and no traps are mapped, creates traps for *all* objects in the table. Does not apply if the selected entry is already mapped, or if no items are selected that lack mappings.
	**Remove Traps button:** Removes the custom trap mappings from the selected item or items in the Trap Support table. When you unmap a trap from an event code, the corresponding "EvFormat" and "Pcause" files are deleted, and that event code is deleted from the database.
	**Edit Traps button:** Opens the Event Configuration application, which lets you edit the mapped trap selected in the Trap Support table. Does not apply to unmapped traps, whose entries lack event codes. For more information, see the *Event Configuration User Guide*.
	**Export button:** Exports the table contents to an external file.
Remap All Conflicts	**Remap All Conflicts check box:** Remaps traps that are partially supported in CA Spectrum, or that have inconsistent, conflicting support across SpectroSERVERs in a DSS environment. Remapping traps makes the traps available to all model types on all SpectroSERVERs.

Import and Export MIBs

You can add MIBs to the MIB Tools database individually using the MIB Import feature. You can also import multiple MIB files using a script at the command line. Import MIBs into the MIB Tools database for the following reasons:

- To view new MIB objects in a MIB that is not imported.
- To retrieve MIB objects from a device whose MIB is not imported.
- To build OneClick views or create Watches based on MIB objects that are not already supported in CA Spectrum.

Note: Importing MIBs is only the first step to using them in OneClick. The second step involves mapping the MIB objects.

You can also export data that is displayed in MIB Tools for use outside of MIB Tools and OneClick. You can export data that is displayed in the Results, Attribute Support, and Trap Support tables into several different file formats.

More information:

Import Multiple MIBs (see page 63)

Import Individual MIBs

You can import individual MIBs into the MIB Tools database. The MIBs that you import are stored in the following directory on the OneClick web server:

<$SPECROOT>/MibDatabase/userContrib

When you import a new MIB using the MIB Tools, CA Spectrum stores it as a custom MIB.

To import a MIB into MIB Tools, the MIB file must be on a file system that is accessible from the OneClick Console. You can only compile text-formatted MIB files. Files in Microsoft Word or rich text formats (containing control characters) cannot be compiled and are ignored.

Note: If you have many MIBs to import, you can import multiple files in bulk using the BulkMibImport command (see page 63). The MIB files that are referenced by MIB Tools can only contain a single MIB. This restriction includes the MIB that is being compiled and any dependent or imported MIBs that are in the same directory. If a referenced file contains multiple MIBs, break each MIB into a separate file.

When you import a MIB file, MIB Tools recursively checks the MIB file for any IMPORTS statements that reference other MIBs. MIB Tools identifies any dependent or imported MIBs in files that are in the same directory as the MIB that is being compiled. As long as they are in the same directory, the MIB files that resolve IMPORTS statements are also compiled and are placed in the following directory on the OneClick web server:

<*$SPECROOT*>/MibDatabase/Dependent

These files are therefore available for subsequent import requests.

After you locate and compile a referenced MIB, the MIB Tools utility no longer has to locate and compile it each time another MIB file references it.

Follow these steps:

1. Click Utilities, MIB Tools from the main menu.

   MIB Tools opens.

2. Click  (Add MIB) in the Navigation panel.

   The 'MIB Tools: Add MIB' dialog opens.

3. Click Browse to locate and select the file that contains the MIB that you want to import. Or manually enter the complete path and filename in the MIB File Name field

4. Click Open.

5. Click Compile.

   A message appears in the Compiler section of the dialog, relating to the status of the compile request.

   On success, a message states that the file was successfully compiled.

   Otherwise, errors or warnings that were generated during the compilation appear. You cannot import a MIB that contains errors. Click Show Editor to edit the MIB file and correct any errors.

6. Once the MIB compiles successfully, click Add to add the MIB and keep the 'MIB Tools: Add MIB' dialog open so that you can add additional MIBs, or click Add & Close to add the MIB and close the dialog.

   The MIB is added to the MIB Tools database, the 'MIB Tools: Add MIB' dialog closes, and the MIB is added to the list in the Navigation panel. If the MIB references a new vendor, a new folder appears for that vendor.

Note: If you import an updated copy of an existing standard MIB, the MIB Tools adds that MIB as a unique custom MIB under the same vendor folder by appending MIB-MODULE-NAME<n>. For example, when you update the ADTRAN-AOS MIB and import it, MIB Tools adds it as ADTRAN-AOS1 after the existing ADTRAN-AOS MIB. You can delete the ADTRAN-AOS1 MIB as it is a custom MIB.

Delete Individual MIBs

You can delete previously imported custom MIBs from the MIB Tools database. When you delete a MIB all the corresponding mapped attributes and events that are associated to traps remain mapped.

Follow these steps:

1. Click Utilities, MIB Tools from the main menu.

   The MIB Tools opens.

2. Expand the required vendor folder, and select the required custom MIB.

3. Click  on the Navigation tool bar.

   The " Delete Custom MIBs" dialog opens.

4. Click OK.

   The MIB Tools deletes the MIB from its database.

Editing MIBs

MIB Tools includes an editor that lets you locate and correct errors that are identified during the compilation of a MIB file. To troubleshoot compiler errors, click Show Editor in the 'MIB Tools: Add MIB' dialog. You can then view the MIB file in the editor and make changes to remove errors. You can search the file for alphanumeric strings, locate specific lines in the file, and save the changes that you make.

Import Multiple MIBs

Use the BulkMibCompile command to import large numbers of MIB files into the MIB Tools database. This command lets you migrate existing MIBs without using the MIB Tools interface. BulkMibCompile is located in the default CA Spectrum installation directory on the OneClick server:

`<$SPECROOT>/MibDatabase/scripts/BulkMibCompile.sh`

This command uses the following format:

```
BulkMibCompile [-u <MYSQL USER>] [-p <MYSQL PASS>] -d <MIB DIRECTORY> [-f <FILE MASK>]
[-skip_search] [-standard_mibs]
```

-u

Specifies the MYSQL username. If you do not specify a username, BulkMibCompile uses the default MYSQL username 'root'. This parameter is not required if the default username is correct.

-p

Specifies the MYSQL password. If you do not specify a password, BulkMibCompile uses the default MYSQL password, 'root'. This parameter is not required if the default password is correct.

-d

Specifies the directory containing the MIBs to import.

- If you are running a Windows Cygwin bash shell, use the following format for specifying a directory:

  ```
  <$SPECROOT>/MibDatabase/scripts/BulkMibCompile.sh -d c:\\MibDirectory
  ```

  Note: The double backslash (\\) is required. A single backslash (\) is an escape character in this environment.

- Otherwise, use the following format:

  ```
  <$SPECROOT>/MibDatabase/scripts/BulkMibCompile.sh -d /usr/MibDirectory
  ```

-f

Specifies the file mask. Use a file mask that includes all of the MIB files that you want to import in a directory. Examples of file masks include:

```
RFC*
*RFC*
*.mib
```

-skip_search

Speeds up the import process by instructing the compiler to resolve IMPORTS statements. Use this option if the MIBs that are referenced in IMPORTS statements in the MIB files that you are importing are in the MIB directory that you specified and are named using their MODULE-NAME.

If you do not use -skip_search, the BulkMibCompile code searches each MIB for IMPORTS statements and attempts to resolve them. This process is repeated during the compilation of each MIB.

-init

Reinitializes the MIB Tools database, clearing the database of all MIBs.

-standard_mibs

Specifies that all the MIBs being imported into the <MIB DIRECTORY> will be added as standard MIBs.

Note: If you run the "BulkMibCompile" command without this option at the CLI, all the imported MIBs will be added as custom MIBs. Custom MIBs can be deleted.

Example: Import All MIBs in a Directory

To import all MIBs in a directory, use the following syntax:

```
<$SPECROOT>/MibDatabase/scripts/BulkMibCompile.sh -d /usr/MibsToCompile
```

A MIB that has been successfully compiled is added to the database. The imported MIB overwrites any existing MIBs with the same MIB MODULE-NAME. Any compilation errors are displayed, and compilation continues with the next MIB that is imported.

Example: Reinitialize a MIB Tools Database

To reinitialize a MIB Tools database, use the following command:

```
<$SPECROOT>/MibDatabase/scripts/BulkMibCompile.sh [-u <MYSQL USER> ] [-p <MYSQL PASS> ] -init
```

Example: Populate the MIB Tools Database of Another OneClick Server from the Primary OneClick Server

To populate the MIB Tools database of another OneClick server from the primary OneClick server, take the following steps:

1. Copy the contents of *<$SPECROOT>*/MibDatabase and *<$SPECROOT>*/MibDatabase/Dependent on the primary OneClick server to the same directories on the secondary OneClick server.
2. On the secondary OneClick server, run BulkMibCompile to import the MIBs that you copied into the *<$SPECROOT>*/MibDatabase:

   ```
   <$SPECROOT>/MibDatabase/scripts/BulkMibCompile.sh -d <$SPECROOT>/MibDatabase -skip_search
   ```

   After the script has completed, the databases on the original and destination OneClick servers are identical.

More information:

Synchronize and Update MIB Databases and Support Files on Multiple OneClick Servers (see page 84)

Create Attribute Support

You can add support for MIB objects that CA Spectrum does not currently support. Create attribute support for a MIB that you have imported into the MIB Tools database.

To create support for MIB objects in the CA Spectrum database, create an attribute identifier (ID) for the MIB objects. Attribute IDs are used to create mappings between the CA Spectrum database and MIB objects. You can use the attribute information to develop custom views, Watches, or leverage other features to manage network devices.

Follow these steps:

1. In the Navigation panel of MIB Tools, select the MIB for which you want to create attribute support.

   The list of MIB objects appears in the Attribute Support table.

2. Check the columns for MIB objects that lack corresponding Attribute IDs.

3. Select a MIB object, and click  (Create Attributes).

4. Click OK to continue creating the attributes on all SpectroSERVERs.

   The 'MIB Tools: Attribute Creation Results' dialog shows the status, including the number of attributes created for each landscape.

5. Click Close.

   The Attribute Support table now shows the Attribute ID that was created for each object. The new attributes are available for use by all SNMP-capable models in CA Spectrum.

Modifying MIB Objects in the MIB Tools Database

After you have imported a MIB object into the MIB Tools database and have mapped the object by configuring attribute support, the object definition is locked. The only part of the object definition that you can change is the enumerations.

Important! Do not edit a MIB object that has been mapped. Modifications to any parameters other than enumerations (such as name or data type) are not supported. In addition, you cannot use MIB Tools to delete attributes that have already been mapped.

The only way to modify a MIB object that has been mapped is to reimport the the MIB into the MIB Tools database. For more information, see Import Individual MIBs.

Query (GET_NEXT), GET, and SET Requests

You can run queries (GET_NEXT) and perform GET and SET operations for MIB objects to retrieve and set information about network devices.

You can query a device whose MIB has not been imported into the MIB Tools database. However, you must then know the Object IDs of the objects to query. In addition, you can query a device that is not modeled in CA Spectrum.

Query a Subtree of Objects

You can query a MIB object in the MIB database on a network device. You can perform a query to repeatedly perform a request to find multiple instances of a MIB object as long as the objects returned have an OID that is prefixed by the OID of the branch you selected to do the query on.

Follow these steps:

1. Establish contact with the device (see page 74) to query.
2. Navigate to the MIB object you want to query.

   Note: Select an inaccessible object (folder) or a table leaf object to perform the SNMP GET_NEXT query. Performing an SNMP GET_NEXT query on a scalar leaf object yields an empty result set.

3. Click  (Query/GET_NEXT) in the Hierarchy tree table toolbar.

   Note: The Query/GET_NEXT button is available when a readable object is selected in the Hierarchy tree table.

   The query results appear in the Results table.

   Note: If you select a group object (contains other groups and individual objects) or a table object (contains multiple instances of the same object) in the Hierarchy tree table and click  (GET), the SNMP GET dialog opens. Change the request type to a GET_NEXT in the Request Type drop-down menu before proceeding. Performing a GET request on a group object fails.

More information:

MIB Tree Hierarchy Table (see page 56)

Query an Object

You can query a MIB object in the MIB database. You can perform a query to search for a single instance of an accessible MIB object. You can also perform a query to repeatedly request multiple instances of a MIB object. But the objects that are returned must have an OID that is prefixed by the OID of the branch where you run the query.

Follow these steps:

1. Establish contact with the device (see page 74) to query.
2. Navigate to the group object or leaf that you want to query.
3. Click  (GET) in the Hierarchy tree table toolbar.

   Note: The GET button is available after you select a readable object in the Hierarchy tree table.

   One of the following things occurs:

   - If a scalar leaf object is selected, the GET request is performed.
   - If a table leaf object is selected, the SNMP GET dialog opens.
4. To retrieve a particular instance of the table object, specify an instance identifier and click OK.

   Note: If the device contains multiple instances of the MIB object, enter the value of the instance to query in the Instance field.

   If the query is successful, the result appears in the Results table.

   If the query fails, an error message appears, indicating the reason for the failure.

   If no results appear in the Results table, the object is not supported by the device, the object is not accessible, or the device is unreachable.

More information:

MIB Tree Hierarchy Table (see page 56)
Export Query Results To Support Troubleshooting (see page 71)

Set an Object

You can set the value of a MIB object on a network device. The MIB objects that you modify must have write access, such as read/write. View the access value for MIB objects in the Hierarchy tree table (see page 56) Access column.

Follow these steps:

1. Establish contact with the device (see page 74) on which you want to set the value of the MIB object.
2. Navigate to the MIB object that you want to modify.
3. Click  (SNMP SET) in the Hierarchy tree table toolbar.

   The 'MIB Tools: SNMP Set' dialog opens.
4. Take the following steps:
   - Verify that the correct MIB object is selected in the dialog. Or for a MIB object that is not in the MIB Tools database, enter the OID for the object that you want to modify.
   - Enter the Instance of the object that you want to modify.
   - Enter the new Value for the object instance.

   Note: Depending on the object type, you can often select a value from a list in the 'MIB Tools: SNMP Set' dialog.
5. Click OK.

   A dialog opens, indicating whether the SET action was successful. If the action was unsuccessful, a reason is provided.

If the SET action fails, use the following checklist for troubleshooting:

1. Verify that the device is reachable. Try to contact the device from the Contact Criteria section of the Browse tab.
2. If you can contact the device, verify that you can perform a GET action on the device for the MIB object.
3. Verify that the MIB object has read/write access by checking the value for the Access parameter in the Hierarchy tree table.

   Note: If the MIB is not part of the MIB Tools database, use the MIB itself to verify that the MIB object has read/write access.
4. Finally, verify that the SNMP community string in the Contact Criteria is the correct one for writing to the device. You can verify the community string value in the CA Spectrum Modeling Information subview in the OneClick Component Detail panel.

More information:

MIB Tree Hierarchy Table (see page 56)

Device Query and SET Results

The Results section of the Browse tab displays the results of SNMP GET_NEXT, GET, and SET requests in a table.

The available columns include the following:

Name

Displays the name of the MIB object queried.

Instance

Displays the instance of the object queried.

Type

Displays the object type, such as Integer, Counter, IP Address, Octet String, Gauge, Time Ticks, and so on.

Value

Displays the value of the MIB object read from the device.

Object ID

Displays the Object ID of the object.

Access

(Hidden by default) Displays the access type for the object. The access type can be read-only, read/write, read-create, or not-accessible.

Note: For more information about setting table preferences, see the *Operator Guide*.

The Results toolbar provides the following functionality:

Button	Description
	**Information button:** Opens a dialog with details about a selected item.
	**SNMP GET button:** Retrieves the value of the selected MIB object on a specific device. Take this step after a SET action because the values in the Results table are not automatically updated.
	**SNMP SET button:** Sets the value of the selected MIB object on a specific device. Available only for objects that have read/write access. Take this step after a SET action because the values in the Results table are not automatically updated.
	**Clear button:** Clears the contents of the Results table.

Button	Description
	**Export button:** Exports the table contents to an external file.
Auto Clear	**Auto Clear check box:** Clears the contents of the Results table each time you query a device. To view sequential SNMP queries, clear this check box.
Filter	**Filter text box:** Filters the Results table. Only results that contain the text string that you supplied as a filter appear in the table.

Export Query Results To Support Troubleshooting

To aid in troubleshooting a problem, CA Technical Support sometimes asks you to provide a .sim file. Exported from MIB Tools, this file contains information to build a simulation of your device. You can use a query to obtain a detailed SNMP snapshot of your device to provide to CA Technical Support.

Follow these steps:

1. Perform a query (see page 68) on the device for which you require support (typically from the 'internet' branch).
2. Click  (Export) in the Results table toolbar.

   The 'Export table data to file' dialog opens.
3. Select Simulation (*.sim) from the 'Save as type' list.

   Note: We recommend using the name of the device in the filename for easier recognition.
4. Save the file to your local file system.

   The query results are exported.

More information:

MIB Tree Hierarchy Table (see page 56)

Custom Vendor Folders

Create Custom Vendor Folders

You can create custom vendor folders to organize your compiled MIBs. You can edit these folders to change their names and can also delete these folders.

Note: A star icon indicates custom vendor folders. You can only modify or delete custom vendor folders, not the predefined vendor folders.

Follow these steps:

1. In MIB Tools, click  (Creates a new folder) in the Navigation panel.

   The 'MIB Tools: Create Vendor' dialog opens.

2. Enter a name for the vendor folder you want to create and click OK.

   The new folder appears in alphabetical order in the Navigation panel.

Edit Custom Vendor Folders

You can change the name of custom vendor folders.

Note: A star icon indicates custom vendor folders. You can only modify or delete custom vendor folders, not the predefined vendor folders.

To edit a custom vendor folder

1. In MIB Tools, select a custom vendor folder in the Navigation panel and click  (Edit Folder).

   The 'MIB Tools: Edit Vendor' dialog opens.

2. Edit the name of the folder and click OK.

   The name of the folder is changed.

Delete Custom Vendor Folders

You can delete the custom vendor folders that you created.

Note: A star icon indicates custom vendor folders. You can only modify or delete custom vendor folders, not the predefined vendor folders.

Follow these steps:

1. In the MIB Tools utility, select a custom vendor folder in the Navigation panel and click  (Delete Folder).

   A confirmation dialog opens.

2. Click Yes to confirm the deletion.

   The custom vendor folder is deleted.

   Any MIBs that were in the deleted folder are automatically moved back to their original predefined vendor folders.

Move MIBs to Custom Vendor Folders

You can move a MIB from a predefined vendor folder to a custom vendor folder. You can also move a MIB from a custom vendor folder back to its default folder.

Follow these steps:

1. In MIB Tools, expand the folder where the MIB you want to move exists, select the MIB, and then click  (Move MIB).

   The 'MIB Tools: Move Selected MIB' dialog opens.

2. Select the custom vendor folder where you want to move the MIB, and click OK.

   The MIB appears in the custom vendor folder in the Navigation panel.

3. (Optional) Move the MIB back to its predefined folder.

4. Expand the custom folder, select the MIB, and click Move MIB.

   The 'MIB Tools: Move Selected MIB' dialog opens.

5. Select the predefined vendor folder where the MIB was originally stored and click OK.

   If you select multiple MIBs to move, an [ORIGINAL VENDOR FOLDER] option appears. Click this option to move multiple MIBs to their respective default vendor folders.

   The MIB reappears in the predefined vendor folder in the Navigation panel.

Contact a Device Using MIB Tools

You can contact a device using MIB Tools. The utility lets you specify parameters for the SNMP query that is sent. SNMP security information information is required.

Follow these steps:

1. Open MIB Tools and click the Browse tab.
2. Complete the following fields in the Contact Criteria section:

   Device

   Specifies the IPv4 or IPv6 address or the hostname for the device.

   Note: Check the Device list to see the last ten devices that were successfully contacted.

   Version

   Specifies the version of SNMP you want to use.

   SNMP v3 Profile

   If you are using SNMPv3, select the profile required to contact the device from the SNMP v3 Profile drop-down. If you want to create a profile, click the Profiles button.

   SNMP Community String

   Specifies the SNMP community string used to access the device.
3. (Optional) If you are using Secure Domain Connector, or if you want to change any of the default values for SNMP to use in contacting the device, click Advanced Options and complete the following fields:

   Landscape

   Specifies the landscape from which the SNMP request should be initiated, if you have a Distributed SpectroSERVER (DSS) environment.

   Secure Domain

   Specifies the secure domain to which the SNMP request should be forwarded, if you have the Secure Domain Manager add-on application installed with secure domains configured.

   Port

   Specifies the value of the UDP port to contact the device on.

   Default: 161

Retry Count

Specifies the number of times to retry contacting the device before MIB Tools stops attempting to contact the device.

Default: 3

Timeout (ms)

Specifies the amount of time, in milliseconds, to wait before trying to contact the device again.

Default: 3000

4. Click Contact to initiate contact with the device.

   The color of the line surrounding the Contact Criteria section indicates the status of contact with the device. The following defines the contact status indicator colors.

   - **Blue:** Contact with the device has not been initiated.
   - **Yellow:** Contact with the device is in progress.
   - **Green:** Contact with the device was successful.
   - **Red:** Contact with the device was unsuccessful.

More information:

Query an Object (see page 68)
Query a Subtree of Objects (see page 67)
Set an Object (see page 68)

Search for a MIB

You can search for MIBs in the Navigation panel of the MIB Tools utility.

Follow these steps:

1. Enter a text string in the Search field in the Navigation panel.

   Note: Wildcards are not supported.

2. (Optional) Select the Ignore Case check box to make the search case-insensitive.
3. Press Enter to search.

   The Next and Previous buttons let you scroll through each successive instance of the search string.

   When a match is found, it is highlighted in the Navigation panel. The Contents panel is populated with the information about the selected MIB.

Trap Support

The MIB Tools utility lets you create custom traps and modify traps. You can create trap support for MIB objects that are not currently supported by CA Spectrum. The first step is importing the MIBs.

You can also create support for traps that are defined in the MIBs that you have imported. And you can customize traps for the MIBs that you have imported into the MIB Tools database. Finally, you can add custom trap support for new devices that are not yet supported in CA Spectrum.

Custom Trap Support File Details

When you map traps using MIB Tools, entries are generated in the following files on all SpectroSERVERs in your DSS environment:

<$SPECR00T>/custom/Events/EventDisp
<$SPECR00T>/custom/Events/AlertMap

The mapping information for a trap in these files overrides any previous mapping information for that trap in other SpectroSERVER files or directories. And when you map traps, files are generated in the following directories on the OneClick server to which you are connected:

- *<$SPECROOT>*/custom/Events/CsEvFormat—An Event Format file applies to each CA Spectrum event. These files define the event text that appears in the OneClick Alarm and Event lists.
- *<$SPECROOT>*/custom/Events/CsPCause—A Probable Cause file defines the text for each CA Spectrum alarm that appears in the OneClick Alarm Details view. Select an Alarm Severity level when you map a trap to enable Probable Cause.
- *<$SPECROOT>*/custom/Events/CsEvFormat/EventTable—An Event Table file determines each varbind sent with a trap that includes enumerated definitions in the MIB.

Manually copy these directories from your primary OneClick server to other OneClick servers in a multiple OneClick server environment.

If you use the CLI commands 'show alarms' or 'show events', or if you deploy CA Spectrum Alarm Notification Manager (SANM), copy the contents of these directories to all SpectroSERVERs in *<$SPECROOT>*/SG-Support.

Note: For more information, see the *Event Configuration User Guide*. The *Concepts Guide* and the *Event Configuration User Guide* provide information about CA Spectrum alarm and event support files.

More information:

Synchronize and Update MIB Databases and Support Files on Multiple OneClick Servers (see page 84)

Create Trap Support

You can add support for MIB objects that CA Spectrum does not currently support by creating trap support. The MIBs must have already been imported into the MIB Tools database.

You can create support in CA Spectrum for traps that are defined in MIBs and are imported into the MIB Tools database. You can add custom trap support for new devices that are not supported in CA Spectrum. Or you can change the way that a trap is supported.

In a fault-tolerant SpectroSERVER environment, MIB Tools does not map traps on a secondary SpectroSERVER when the primary SpectroSERVER is down. If you have multiple primary SpectroSERVERs, then at least one of your primary SpectroSERVERs can be available to create trap support.

Follow these steps:

1. Select Tools, Utilities, and MIB Tools from the OneClick main menu.
2. Select MIB in the Navigation panel.

   A list of traps for each MIB appears in the Trap Support table. The  button (Map Traps) is only enabled if some traps lack CA Spectrum support.

   Note: Use multiselect to select specific traps. Support is created only for the traps you select.

   Important! Any traps with partial support appear in the Trap Support table with Partial in the Event Code column. To create global trap support for these traps, remap these traps before continuing. For more information, see the Modeling Your IT Infrastructure Administrator Guide.

3. Click  (Map Traps).

   The MIB Tools: Assign Trap Alarms dialog opens.

4. Click set in the Alarm Severity column to select the severity for the alarm that CA Spectrum generates when it receives the trap.

5. Click OK to map the traps on all SpectroSERVERs.

   The MIB Tools: Trap Support Results dialog displays the status of the Map Traps action. When the action completes, the results list the number of traps that were created for each landscape.

   You can use Show Advanced Options when mapping traps to CA Spectrum events. For more information, see Show Advanced Options for Mapping Traps (see page 80).

6. Click Close.

   The Trap Support table shows the event code for each trap. The new events are now available for use by all model types in the CA Spectrum modeling catalog.

   CA Spectrum processes the trap accordingly.

Review Custom Trap Mapping Information

You can use MIB Tools to determine whether a trap has a default mapping, a custom mapping, or both.

Follow these steps:

1. In MIB Tools, select the MIB in the Navigation panel whose trap support you want to verify. Click the Map tab.

   The list of traps for the MIB appears in the Trap Support table.

2. Right-click the Trap Support table column header.

   The Table Preferences dialog opens.

3. In the Columns tab, select the Event Type check box, and click OK.

   The Event Type column is added to the Trap Support table.

   Default mappings have 'CA' in the Event Type column.

   Custom mappings have 'Custom' in the Event Type column. If a custom trap mapping is obscuring, or shadowing, a default trap mapping, the Event Type column displays a ‘Custom’ hyperlink.

4. Click the ‘Custom’ link.

   The MIB Tools: Custom Trap Mapping Details dialog opens.

5. Review the Custom Disposition Details and the CA Disposition Details for the trap to determine whether to remove the custom trap mapping (see page 79).

6. Click Close.

   The dialog closes.

Remove Custom Trap Mappings

You can remove custom mappings from traps in MIB Tools.

Follow these steps:

1. In MIB Tools, select the MIB in the Navigation panel whose custom trap support you want to remove. Click the Map tab.

   The list of traps for the MIB appears in the Trap Support table.

   The Remove Traps button is available if at least one item with a custom mapping is selected.

2. Click  (Remove Traps).

   The MIB Tools: Delete Trap Mappings dialog opens.

3. Click OK to confirm the removal of this custom mapping.

   The MIB Tools: Trap Support Results dialog shows the status of the Remove Traps action.

   The results show the number of traps that were removed for each landscape.

4. Click Close.

   If no default mapping exists, the Trap Support table shows a blank entry in the Event Code column. However, if the custom mapping was shadowing a default mapping, the event code of the default mapping is displayed.

Remove Partial Mappings From Traps

You can identify when a trap is only supported on select model types in the CA Spectrum modeling catalog. You can also change trap support to include all model types.

When a trap is globally supported in CA Spectrum, it is available for use by all model types. The event code value for a globally supported trap appears in the Event Code column in the Trap Support table.

When a trap is partially supported in CA Spectrum, it is available for use by certain model types only. The event code value for a partially supported trap appears in the Event Code column in the Trap Support table as 'Partial.'

We recommend not modifying partially supported traps. However, you can remap a trap to make it globally supported in CA Spectrum.

Important! If you remap a trap, you can disable useful CA Spectrum functionality. Proceed with caution.

To change partial trap support to global trap support, remove the partial mapping and then remap the trap.

Follow these steps:

1. In MIB Tools, click the Map tab.
2. Click 'Partial' in the Event Code column of the Trap Support table for the trap you want to remap.

   The 'MIB Tools: Partial Trap Support' dialog opens, displaying the trap event code, the landscapes it is available on, the MIB object name, and the object ID.
3. Click Remap.

   The 'MIB Tools: Confirm Remap' dialog opens.
4. Click Yes to continue remapping the trap.

   The remapped trap appears in the Trap Support table with the previous partial mapping removed; there is no event code value and 'Partial' no longer appears in the Event Code column.
5. (Optional) Remap these traps globally (see page 77).

   The partial mapping is removed.

To remove a partial mapping from all traps in a MIB, select the Remap All Conflicts check box.

Follow these steps:

1. In MIB Tools, select the MIB that contains the traps whose partial support you want to remove in the Navigation panel. Click the Map tab.

   The traps with partial support appear in the Trap Support table.
2. Select the Remap All Conflicts check box.

   All partial trap support is removed for the traps.
3. Verify that the traps lack an event code and that 'Partial' no longer appears in the Event Code column.

   The previous partial mapping is now removed.
4. (Optional) Remap these traps globally (see page 77).

   The partial mapping is removed.

Show Advanced Options for Mapping Traps

You can use Show Advanced Options when mapping traps to CA Spectrum events. This option lets you select the event codes to assign to new trap mappings. You can also export the events and alarms to a specific directory.

Follow these steps:

1. Select Tools, Utilities, and MIB Tools from the OneClick main menu.
2. Select MIB in the Navigation panel.

   A list of traps for each MIB appears in the Trap Support table. The  button (Map Traps) is only enabled if some traps lack CA Spectrum support.

   Note: Use multiselect to select specific traps. Support is created only for the traps you select.

   Important! Any traps with partial support appear in the Trap Support table with Partial in the Event Code column. To create global trap support for these traps, remap these traps before continuing. For more information, see the Modeling Your IT Infrastructure Administrator Guide.

3. Click  (Map Traps).

   The MIB Tools: Assign Trap Alarms dialog opens.

4. Click Show Advanced Options.

   The following advanced options are available in MIB Tools:

   Starting Event Code

   Lets you specify the first code to use for events.

   By default, CA Spectrum automatically calculates and assigns event codes for new trap mappings. A read-only event code appears in the Starting Event Code field and in the selected Use Default check box.

   However, you can specify a starting event code to assign to new trap mappings. Clear the Use Default check box to enable the Starting Event Code text box. CA Spectrum then automatically calculates and assigns the event codes that are based on the new starting event code.

   Note: Select a unique starting event code.

   Install Trap Support

   Installs the event and alarm support files in the CA Spectrum installation area, providing immediate support for the new trap mappings.

   Export Trap Support

   Lets you specify the directory where the event and alarm support files are exported. Selecting this option does *not* provide support for the new trap mappings in CA Spectrum.

MIB Tools Support for Multiple SpectroSERVERs

The MIB Tools utility offers the following features to supports a Distributed SpectroSERVER (DSS) environment:

- **Offline SpectroSERVERs:** MIB Tools can recognize that a SpectroSERVER is offline while attempting to create Event Codes and Attribute IDs for MIB objects. You receive a notification.
- **Incomplete Trap and Attribute Support:** MIB Tools can identify situations where MIB objects are supported on some, but not all, SpectroSERVERs.
- **Resolve Incomplete Support:** MIB Tools can resolve incomplete trap support.
- **Conflicting Support:** MIB Tools can identify MIB objects that have conflicting Event Code or Attribute ID mappings on multiple SpectroSERVERs.
- **Resolve Trap Conflicts:** MIB Tools can resolve trap disposition conflicts.

More information:

Resolve Trap Disposition Conflicts: Remap the Trap (see page 85)
Create Consistent Support Across a DSS Environment (see page 83)

MIB Tools Synchronization in a DSS Environment

Use the following guidelines to maintain synchronization among SpectroSERVERs in a DSS setup:

- Do not create attribute or trap support if any of the SpectroSERVERs are down. A warning dialog appears in this situation.
- Always create consistent support for attributes and traps on all SpectroSERVERs.
- Always resolve trap disposition conflicts (see page 85) when they appear.

Attribute Conflicts

When inconsistent support for an attribute occurs in a DSS environment, an attribute conflict condition exists. Usually, this results when an attribute is mapped to different attribute IDs on one or more SpectroSERVERs.

The Attribute ID value for the attribute in the Attribute Support table indicates that a conflict has been detected. To see the landscape and the attribute ID for the attribute, click the 'Conflict' link in the table. At least one SpectroSERVER has a different attribute ID, or no attribute ID, for the attribute in conflict situations.

To resolve attribute conflicts, synchronize the modeling catalogs on your SpectroSERVERs.

Note: For more information, see the *Distributed SpectroSERVER Administrator Guide.*

Create Consistent Support Across a DSS Environment

When a trap or attribute is supported on some, but not all, SpectroSERVERs in a DSS environment, the value 'Some' in the Landscapes column. Click the 'Some' link to see the mapping of the attribute ID or event code to a landscape for a MIB object. Landscapes where the selected MIB object is not mapped lack a value.

Create consistent support across your DSS environment for traps and attributes that are supported on some of your SpectroSERVERs. Remap the traps and recreate the attributes once all SpectroSERVERs are running.

You can create consistent attribute support and consistent trap support across a DSS environment.

Follow these steps:

1. Select the MIB containing the attributes for which you want to create consistent support in the Navigation panel of MIB Tools.
2. Click the Map tab, and verify that attributes appear in the Attribute Support table with the value 'Some' in the Landscapes column.
3. Click  (Create Attributes), and click OK to confirm.

   The MIB Tools: Attribute Creation Results dialog displays the status and results of the Create Attributes action.
4. Click OK.

   The value ‘All’ appears in the Landscapes column for all the attributes.

Follow these steps:

1. In MIB Tools, select the MIB containing the traps for which you want to create consistent support in the Navigation panel.
2. Click the Map tab and verify that traps appear in the Attribute Support table with the value 'Some' in the Landscapes column.
3. Click  (Map Traps).

   The MIB Tools: Assign Trap Alarms dialog opens.

4. (Optional) Click set in the Alarm Severity column to select the alarm severity for the alarm that CA Spectrum generates when it receives the trap. Or select None if the trap does not generate an alarm.
5. Click OK to map the traps on all SpectroSERVERs.

   The MIB Tools: Trap Support Results dialog shows the status of the Map Traps action. The results list the number of traps that were created for each landscape.
6. Click Close.

   The Trap Support table shows the event code for each trap.

More information:

MIB Tools Support for Multiple SpectroSERVERs (see page 82)

Synchronize and Update MIB Databases and Support Files on Multiple OneClick Servers

When you import MIBs into the MIB Tools database and create trap and attribute support, information is written to each SpectroSERVER in your DSS environment. However, information required by OneClick to support the traps and attributes is written only on the OneClick server to which you are connected.

If you have multiple OneClick servers in your environment, maintain synchronicity among the MIB Tools databases and support files for new attribute and trap support.

You can synchronize and update MIB databases and support files on multiple OneClick servers.

Follow these steps:

1. Designate one of your OneClick servers as the primary server that contains the primary MIB Tools database.
2. Import all MIBs and create attribute and trap support on this primary OneClick server.
3. Distribute the MIB Tools database to other OneClick servers (see page 63) from the primary server.
4. Distribute OneClick support files (see page 76) created for events and alarms from the primary OneClick server to the other OneClick servers.

More information:

Import Multiple MIBs (see page 63)

Trap Disposition Conflicts

When inconsistent support for a trap occurs in a DSS environment, a trap disposition conflict condition exists. Inconsistent trap support includes the following situations:

- A trap is mapped to different event codes on one or more SpectroSERVERs.
- A trap is disposed differently on one or more SpectroSERVERs.
  - For example, a trap is disposed to a minor alarm on one SpectroSERVER, and a major alarm on another.
  - A trap is using an event rule for complex processing, but another instance of the trap is not using that rule.

When a trap disposition conflict exists, the Event Code value for that trap in the Trap Support table reads ‘Conflict.

More information:

MIB Tools Synchronization in a DSS Environment (see page 82)

Resolve Trap Disposition Conflicts: Remap the Trap

You can resolve a trap disposition conflict by remapping the trap on all SpectroSERVERs in your DSS environment. Resolving trap disposition conflicts creates consistent support for the trap on all SpectroSERVERs. Each SpectroSERVER in your DSS environment must be running to resolve a trap disposition conflict.

Follow these steps:

1. In MIB Tools, click the Map tab and locate the trap in conflict in the Trap Support table.
2. Take one of the following steps:
   - For multiple trap disposition conflicts, select Remap All Conflicts.
   - For a single trap, click Conflict in the Event Code column.

     The Trap Disposition Conflict dialog lists each landscape and the event code for the trap.

     Click Remap, and then click Yes to confirm.

The trap now has a value of 'Some' or 'None' in the Landscape column of the Trap Support table.

3. Click  (Map Traps) to create an event code for the trap on all SpectroSERVERs in the DSS environment.

   The traps are remapped.

Resolve Trap Disposition Conflicts: Edit the AlertMap and EventDisp Files

You can resolve a trap disposition conflict by editing the AlertMap and EventDisp files for the trap. Such conflicts occur in a DSS environment when, for example, a trap is mapped to different event codes on one or more SpectroSERVERs.

Follow these steps:

1. In MIB Tools, click the Map tab and locate the trap in conflict in the Trap Support table.
2. Click Conflict in the Event Code column.

   The Trap Disposition Conflict dialog lists each landscape and the event code for the trap.
3. Locate the desired mapping by reviewing the Event details.
4. Synchronize the conflicting events on the other SpectroSERVERs in your DSS environment by editing the appropriate EventDisp file and AlertMap file with a text editor.

   Note: The AlertMap files and EventDisp files are located in the <*$SPECROOT*>/SS/CsVendor directory.
5. Issue a command on each SpectroSERVER to reload its events and alerts.

   For more information, see the *Event Configuration User Guide*.

Chapter 6: Developing a New Certification

This section contains the following topics:

New Certification Management (see page 87)
New Device Model Type (see page 89)
Creating a New Application Model Type (see page 100)
How to Add Support for Additional Traps (see page 111)
Distributing a New Certification (see page 112)

New Certification Management

To model a device, CA Spectrum uses a device model type and its associated interface and application model types. You can add device model types or you can enhance the functionality of the GnSNMPDev device model type. To enhance device management with CA Spectrum, a solid understanding of the functional components of a device is required.

The GnSNMPDev device model type and the interface and application models that are known to the GnSNMPDev model type support many device functions. Supported functions comprise both proprietary and standard MIBs. Identify the functionality of the device that GnSNMPDev already supports. For example, if device interfaces map one-to-one with physical ports on a single board, GnSNMPDev supports this device without enhancement. GnSNMPDev includes native support for MIB-II interfaces in the Snmp2_Agent application model.

To test GnSNMPDev device support, use an IP address to model the device in OneClick. CA Spectrum automatically finds the model type most appropriate for the device. If a specific model type is lacking, CA Spectrum selects the GnSNMPDev model type and instantiates a GnSNMPDev model to represent the device. You can then evaluate the type of support that CA Spectrum can provide for your device by default.

Once you have established the default support, consider the required customizations. You can then more easily decide if further customization is necessary to manage your device properly. The following sections outline some scenarios in which expanded support is required.

Additional MIB Support

If device management in your environment requires access to additional MIBs, MIB objects can be made available to a device model. The following methods let you increase access to MIBs:

- (Recommended) Import the new MIB directly into the SpectroSERVER using the import mechanism provided in MIB Tools.
- Create a device model type to represent your device and include the necessary MIBs in the device model type.
- Create an application model type that provides access to the new MIB.

More information:

Creating a New Application Model Type (see page 100)
New Device Model Type (see page 89)

Unique Trap Mapping

Create a device model type if the device that you are modeling requires unique trap processing in response to a common trap. For example, assume that you want core routers to generate a major alarm in response to an authentication failure. However, all other devices generate a minor alarm in response to the same failure.

By default, CA Spectrum generates a minor alarm in response to an authenticationFailure trap. You can create a device model type, and you can configure support for the trap in the event and alert configuration files for this device model type. This support overrides CA Spectrum default processing for the authenticationFailure trap for this model type only.

Unique Watches

You can generate events and alarms that are based on the results of a watch. The GnSNMPDev model type provides a number of predefined watches that can be enabled for individual models.

You can customize the watch implementation on the models that represent your device for each applicable GnSNMPDev model. However, you can avoid repeating this customization on each model by creating your own device model type to implement the customized watch.

All of the information that makes up a watch is stored as attributes in the model type specified in the watch. The only exception is the probable cause information that is created for an alarm that results from the watch. This information is stored in the ProbCause model type .

Because you have not created the ProbCause model type with your Developer ID, you lack permission to export and distribute it with your management module. As a result, the probable cause information for the watches that you have created is not distributed. To solve this problem, derive a new model type from the ProbCause model type. The probable cause information for any watches for any of your management modules is automatically stored in this derived model type. Because you have created this derived model type, you can distribute it with your management module.

Note: For more information, see the *Watches User Guide*.

More information:

New Device Model Type (see page 89)

Interface Model Creation

If your device does not advertise interface (port) information in the MIB-II standard interface table but instead uses information from a proprietary MIB, CA Spectrum cannot model the associated interfaces.

Without interface models, you cannot resolve connections to the interface level, nor can you monitor the status of each interface. To work around this problem, create a new application model type that includes support for the proprietary MIB with the interface information.

More information:

Creating a New Application Model Type (see page 100)

New Device Model Type

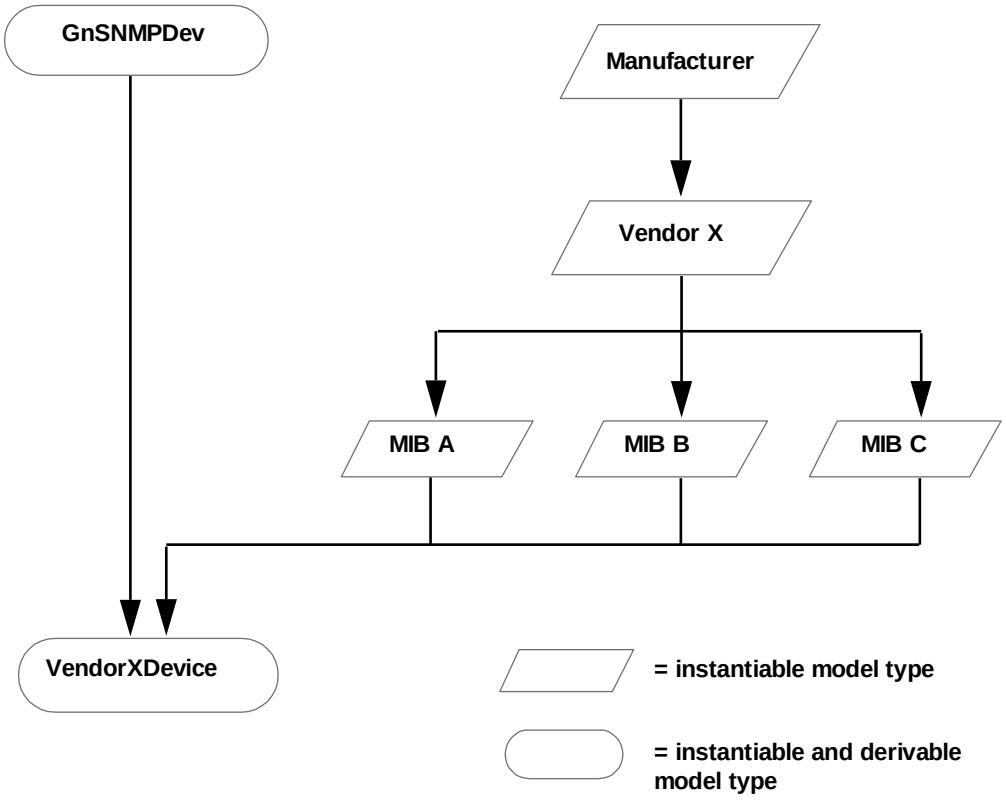
CA Spectrum offers multiple options for creating device model types. The topics in this section describe some of the factors to consider. A new device model type requires some or all of the following tasks:

- Understanding the database derivation and MIB requirements
- Creating the model type and setting required attributes
- Selecting discovery and identification mechanisms

- Making desired customizations
- Making the new model type distributable to other CA Spectrum hosts

New Device Model Type Design

The device model type database architecture for developing new device model types organizes all of the proprietary MIBs that you import into separate MIB model types. These MIB model types can then be derived directly into the device model type, as shown in the following diagram:



This scheme has the following advantages:

- A single MIB can be derived into multiple model types. For example, you can use the same MIB to create multiple device model types or a device and an application model type while maintaining the attribute IDs.
- Vendor MIBs can be organized in a single collection.
- Convenient access to MIB information is available directly from the device model. For example, you can access OneClick views, watches, and logging and polling information about proprietary vendor attributes from the device model.

If the new device model requires access to additional MIBs from CA Spectrum, the simplest method is the MIB import mechanism that is available in MIB Tools. This mechanism creates attributes from these objects and makes them available to all models that represent a device in the SpectroSERVER database.

Note: For more information, see the *Device Management User Guide*. The MIB import mechanism distributes the new MIB across all SpectroSERVERs in a distributed environment.

New Device Model Type Creation

After selecting a database scheme, use the Model Type Editor to create your model types. Derive all device model types from the GnSNMPDev model type.

Note: For more information, see the *Model Type Editor User Guide*.

When you are using the Model Type Editor to create a device model type, remember to set the model type flags correctly for the new model type.

More information:

Model Type Flags Setting (see page 91)

New Device Model Type Configuration

A few steps are involved in configuring a new device model type. Complete the following tasks:

- Set model type flags
- Set attribute values
- Map a device or a device family to the new device model type
- Configure serial number handling

The following sections provide high-level information about these configuration steps.

For more information, see the *Model Type Editor User Guide*.

Model Type Flags Setting

Set the values of model type flags so that models of the new model type behave as expected. Each flag represents a Boolean value and can either be selected (set to TRUE) or not (set to FALSE).

In most cases, you set the Visible, Instantiable, and Derivable flags to TRUE.

Visible flag

Makes the model type visible to all Model Type Editor users. If set to FALSE, the model type is only visible to a user with the same Developer ID as the one used to create the model type.

Instantiable flag

Lets you instantiate a model of this model type in OneClick.

Derivable flag

Lets this model type be used as a base for other model types.

In most cases you should set the No Destroy, Unique, and Required flags to FALSE.

No Destroy flag

If set to TRUE, prevents users from destroying a model of this type in OneClick.

Unique flag

If set to TRUE, only lets one model of this model type be instantiated in OneClick.

Required flag

If set to TRUE, a model of this model type must always exist in the SpectroSERVER database.

Attribute Values Setting

Once you have created your device model type, use the Model Type Editor to set the default value of several attributes. Some of these settings are used to configure built-in capabilities that are inherited by deriving from the GnSNMPDev model type. The following section describes the attributes and settings.

CompanyName

Is the name of the company that developed the management module.

Description

Is an attribute that exists in the MMDeveloper group and in the CommonInfo group. The Description attribute in the MMDeveloper group has a default value of Generic SNMP Device Management Module. We recommend resetting this default value to a description of your management module. The Description attribute in the CommonInfo group can be similarly reset or left empty.

Desc_Key_Word

Enables resolution of multiple device model types. If the System_Desc_Verify or Vendor_Object_ID discovery mechanisms identify multiple device model types, a search of sysDescr looks for a substring match for the value of this attribute.

DeviceSerialAttr

Is the device serial number. Set the value to the attribute ID of the external attribute that contains the serial number. When the model is created, it reads this external attribute and writes it into Serial_Number.

DeviceType

Identifies the device. A default value is required for this description attribute when the DeviceNameList mechanism is not used for identification. Setting the default value guarantees that a value is present for displaying, sorting, and filtering.

DeviceTypeDiscEnable

Enables or disables DeviceType naming intelligence. The default value (true) is appropriate for most device model types. However, set this value to false under either of the following conditions:

- A new device model type has been derived from a base model type other than GnSNMPDev. The new type has specialized DeviceType naming intelligence that is inappropriate for the derived device model type.
- A more appropriate DeviceType name can be set in the catalog for the derived model type.

Disposable_Precedence

Is evaluated during device model type discovery when multiple model types are identified as candidates. The higher value is the chosen model type.

This value is also evaluated when a model is created with the same MAC address as a previously existing model. In this case, the disposable_precedence attributes of both models are evaluated. The model with the higher value replaces the existing model by appropriating its CONNECTs associations.

Enable_IH_Enterprise_Disc

Enables or disables the automated setting of the Manufacturer and App_Manufacturer attributes based on the enterprise ID term of the device sysObjectID.

The default value, true, is appropriate for GnSNMPDev because it is used to model devices from various manufacturers. However, for a new device model type that is derived from GnSNMPDev with a known device manufacturer, we recommend setting the value of Enable_IH_Enterprise_Disc to false. With that attribute set to false, set the default values of the Manufacturer and App_Manufacturer attributes to the appropriate names.

Manufacturer

Is the name of the vendor that manufactures the device.

MMName

Is the name of the management module.

MMPartNumber

Is the part number that you plan to assign to the management module.

System_Desc_Verify

Provides a device model type discovery mechanism that parses the sysDescr for firmware version information. Clear this default value if you are not using this discovery mechanism. It interferes with the other discovery methods if enabled.

System_Oid_Verify

Is a legacy attribute. Refer to SysOIDVerifyList.

SysOIDVerifyList

Used in conjunction with DeviceNameList. Populating this list attribute with sysObjectID values enables device model type discovery intelligence to match the list against the device sysObjectID value. If a match is found, this model type is selected as a possible candidate for modeling.

Vendor_Name

Is the name of the company developing the management module.

Vendor_Object_ID

Provides a device model type discovery mechanism by which a partial sysObjectID match identifies a device model type.

VendorIDVerifyList

Maps devices to device model types based on whether the device supports specific MIB objects. Used during Discovery with VendorOIDVerifyList.

Specify the list of enterprise IDs to compare against the device to model. If a match is found, the corresponding MIB object specified in VendorOIDVerifyList is read from the device.

VendorOIDVerifyList

Maps devices to device model types based on whether the device supports specific MIB objects. Used during Discovery with VendorIDVerifyList.

Specify the list of attribute IDs for the MIB objects to read from the device.

Verify_Mismatch_Model

Causes CA Spectrum to perform checks for a device model type match with the device that is modeled. Set this attribute to true.

More information:

Map Using a MIB Object (see page 97)
Device Mapping (see page 95)
Map Using Unique sysObjectID Value (see page 95)
Map Using sysObjectID and Strings in sysDesc (see page 96)
Map Using Firmware Version Strings in sysDesc (see page 97)

Device Mapping

Each device on the network requires a unique identifier. Most commonly the MIB-II object sysObjectID provides this unique identifier. Vendors typically assign a unique sysObjectID value for a device, creating a one-to-one mapping. Vendors often advertise this information in a product MIB, where you can find the mapping of sysObjectID to device model type.

You can use the Model Type Editor to identify a device in several ways:

- If your device provides a unique sysObjectID value, use the process described in Map Using Unique sysObjectID Values (see page 95).
- If your device does not provide a unique sysObjectID but does provide a unique substring within sysDescr, use the process described in Map Using sysObjectID and Strings in sysDesc (see page 96).
- If your device does not provide a unique sysObjectID but does provide a firmware version text string in sysDescr, use the process described in Map Using Firmware Version Strings in sysDesc (see page 97).
- If your device does not provide any of the previously described information, you can map the device to a device model type based on whether the device supports specific MIB objects in a proprietary MIB. Refer to Map Using a MIB Object (see page 97).

More information:

Map Using a MIB Object (see page 97)
Map Using Unique sysObjectID Value (see page 95)
Map Using sysObjectID and Strings in sysDesc (see page 96)
Map Using Firmware Version Strings in sysDesc (see page 97)

Map Using Unique sysObjectID Value

If your device has a unique sysObjectID value, you must relate your new device model type to the sysObjectID to help ensure that CA Spectrum selects the new device model type to represent the device. To do this, add the sysObjectID value to the SysOIDVerifyList model type attribute. If the new device model type represents a family of devices, then add each sysObjectID value.

If another model type contains the same sysObjectID value in its SysOIDVerifyList attribute, it is possible that CA Spectrum will choose the other model type to represent a device with this sysObjectID. If this occurs, you should change the disposable_precedence attribute value on your device model type to a higher value than that of the other model type. For example, if the other model type has a disposable_precedence value of 10, change the disposable_precedence value on your model type to 11.

To provide identification to your model, configure the model type to display a different device name for each of the devices that the model type is designed to support. For example, assume your device model type represents the 8480 series of switches made by MySwitch, Inc. Instead of seeing the device name MySwitch_8480XX for all of the switches in the 8480 family, you want to display the model number of the switch, as appropriate. If CA Spectrum is modeling an 8480-09 switch, the model should display the device name MySwitch_8480-09. If CA Spectrum is modeling an 06 switch, the model should display the device name MySwitch_8480-06.

Follow these steps:

1. Set the SysOIDVerifyList attribute equal to the sysObjectID(s) of the devices that the model type represents.
2. Set the DeviceNameList attribute equal to the device names that apply to each sysObjectID listed in the SysOIDVerifyList attribute.
3. Specify the same number of names in the DeviceNameList attribute as there are sysObjectIDs listed in the SysOIDVerifyList attribute. List the names in the same order as their corresponding sysObjectIDs.
4. Clear the System_Desc_Verify default value.

   The DeviceNameList attribute only works for device model types that use the SysOIDVerifyList attribute model type discovery mechanism. Verify that both lists have the same number of entries. Otherwise, the DeviceType attribute is not set correctly.

Map Using sysObjectID and Strings in sysDesc

If your device does not provide a unique sysObjectID, a partial or complete match of the sysObjectID in combination with a sysDescr substring can provide unique identification.

You can set up the relevant attributes to enable this functionality on the model type.

Follow these steps:

1. Set the Vendor_Object_ID attribute equal to the partial or complete sysObjectID value returned by your device.

   Only the first seven terms up to the enterprise ID are used for comparison.
2. Set the Desc_Key_Word attribute equal to the unique, partial sysDescr value returned by your device.

3. Set the DeviceType attribute to be equal to the desired identification string.
4. Clear the System_Desc_Verify default value.

Map Using Firmware Version Strings in sysDesc

If your device does not provide a unique sysObjectID or a unique sub-string within sysDescr, check whether sysDescr provides a unique firmware version. This discovery mechanism searches the sysDescr value for either “Version” or “Revi.” If one of these strings is found, the value of System_Desc_Verify is compared against the text that follows these key words. If a match is found, the device model type is selected. In the case where multiple model types have the same System_Desc_Verify value, a substring in sysDescr can be compared by setting the Desc_Key_Word.

You can set up the relevant attributes to enable this functionality on the model type.

Follow these steps:

1. Set the System_Desc_Verify attribute to be equal to the contents of sysDescr that follow the key text noted above.
2. Set the Desc_Key_Word attribute to be equal to the unique, partial sysDescr value returned by your device.
3. Set the DeviceType attribute to be equal to the desired identification string.

Map Using a MIB Object

If your device does not provide a unique sysObjectID or a unique sub-string or firmware version within sysDescr, check if it supports a proprietary MIB. You can map the device to a device model type based on whether the device supports specific MIB objects.

This discovery mechanism compares the enterprise ID of the device against each enterprise ID specified in the VendorIDVerifyList attribute. If a match is found, the MIB object specified in the same instance of the VendorOIDVerifyList attribute is read from the device. If the SNMP read succeeds, this model type is added to the list of model type candidates (from which the model type with the highest disposable_precedence attribute value is ultimately selected). The enterprise ID match mechanism is implemented for performance reasons: the SNMP read is only initiated for targeted devices.

You can set up the relevant attributes to enable this functionality on the model type.

Follow these steps:

1. Add the enterprise ID of the device to model with this model type to the VendorIDVerifyList attribute.
2. Add the attribute ID of the MIB object to read from the device as the corresponding instance in the VendorOIDVerifyList attribute.

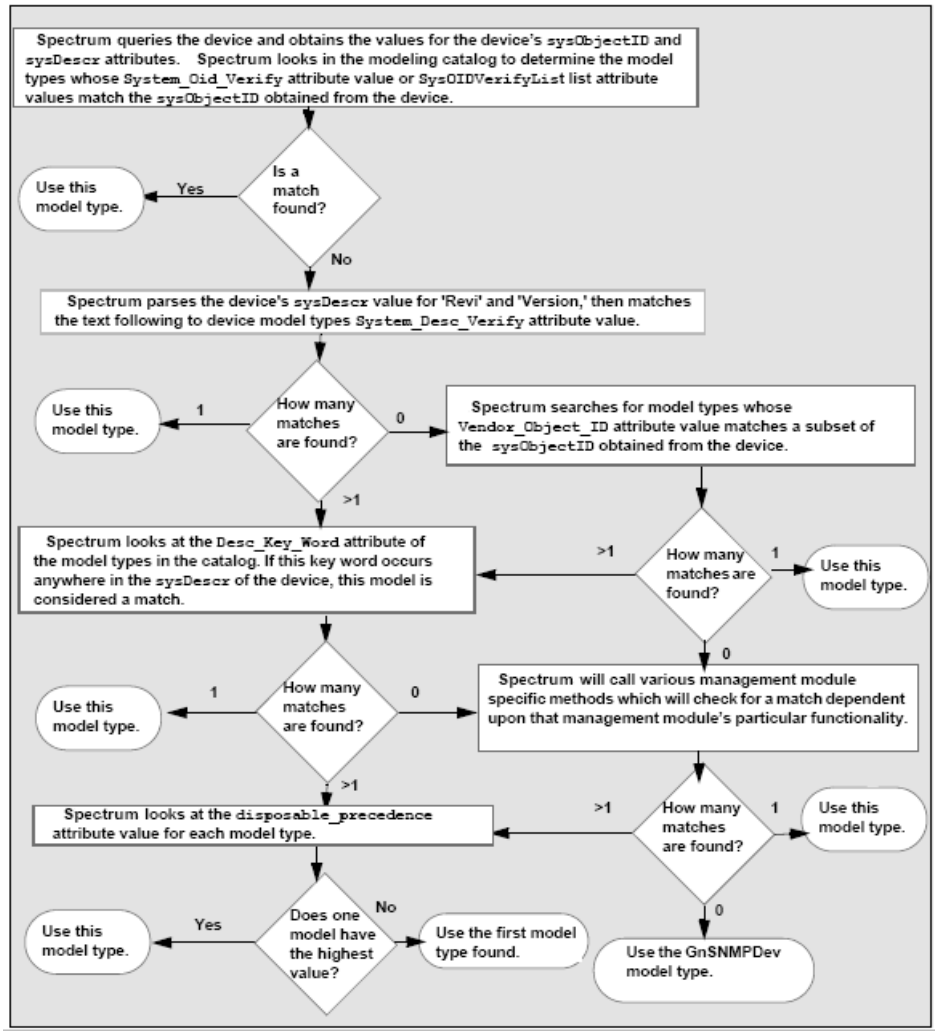
3. Repeat the preceding steps for each enterprise ID/attribute ID pair to evaluate in conjunction with one another.
4. Set the DeviceType attribute equal to the desired identification string.

Discovery and Identification Flowchart

This flowchart identifies the steps that CA Spectrum takes to determine the device model type to represent a device. The flowchart includes discovery and identification mechanisms:

- Using unique values in sysObjectID
- Using strings in sysObjectID and sysDesc
- Using firmware version strings in sysDesc

You can also map a device (or a device family) to a new device model type based on whether the device supports specific MIB objects in a proprietary MIB.



More information:

Map Using a MIB Object (see page 97)

Configure Serial Number Handling

Device model types contain an attribute for setting and displaying the serial number of the modeled device: Serial_Number (0x10030). Enter the appropriate serial number value in any of the views where the attribute is displayed. Or, if the serial number is available as an external device attribute, you can configure the model type to retrieve this value and set the Serial_Number attribute.

Follow these steps:

1. Verify that the external attribute that contains the serial number is not a list attribute and is of type TEXT_STRING or OCTET_STRING.
2. Set the value of the DeviceSerialAttr (0x3d0063) attribute for this device model type in the Model Type Editor. Make this value equal to the ID of the external attribute that contains the serial number.

   When a model of this model type is instantiated, CA Spectrum sets the Serial_Number attribute so that the value is equal to the value of the external attribute.

Creating a New Application Model Type

This section describes how to expand support for a device using application model types. All application model types are derived from a series of standard model types, called *derivation points*. An Application often corresponds to the functionality of a MIB.

Derivation Points and Model Fragments

Select derivation points and use them as base model types for new application model types. Derivation points have the functionality to support different types of applications. When you derive a new model type from one or more of these derivation points, the model type inherits the derivation point functionality.

Some derivation points require the use of model fragments. The available model fragments are model types with attached inference handlers. These inference handlers provide the model fragments with certain behaviors and intelligence, such as the ability to create port or board models. To use the functionality from these inference handlers, map attribute IDs from the model type that represents the MIB to specific model fragment attribute values.

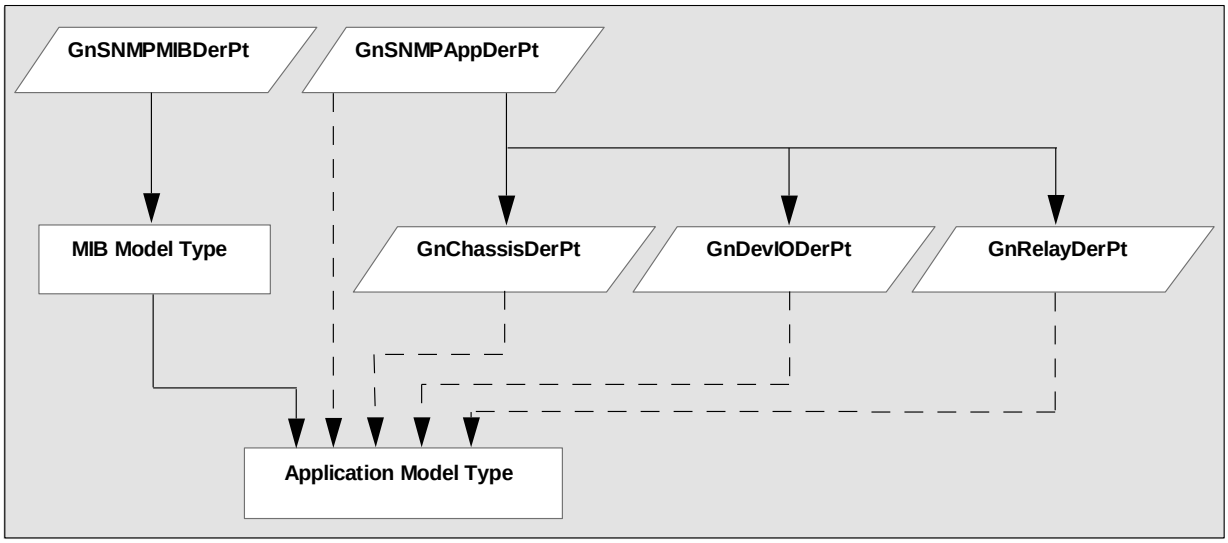
Model fragments are typically included as base model types for the GnSNMPDev derivation points that require them. However, it can be necessary to add a model fragment as a base model type to a new model type to gain the capabilities of the inference handler that is attached to the model fragment.

The following model types can be used as application derivation points:

- GnSNMPMibDerPt
- GnSNMPAppDerPt
- GnChassisDerPt
- GnDevIODerPt
- GnRelayDerPt

The following figure shows the application derivation point hierarchy and sample derived model types. The lines connecting the model types denote the inheritance structure.

Note: Select only one of the dotted line paths for the derivation hierarchy of your new application model type.



The derivation points for application model types are all designed to provide specific functionality. The GnChassisDerPt, GnDevIODerPt, and GnRelayDerPt derivation points have model fragments that enhance this functionality. The following table shows each derivation point, the application model type that it creates, and its associated model fragments.

Derivation Point	Model Type	Associated Model Fragment
GnSNMPMibDerPt	MIB Model Type	N/A
GnSNMPAppDerPt	Application model type with no requirement to manage ports or boards.	N/A

Derivation Point	Model Type	Associated Model Fragment
GnSNMPMibDerPt	MIB Model Type	N/A
GnChassisDerPt	Application model type to model chassis functionality. Provides management for ports and boards.	GnChassis_MF
GnDevIODerPt	Application model type for devices that require port management but not board management (such as switches or terminal servers).	GnDeviceIO_MF
GnRelayDerPt	Application model type for repeater functionality	GnDataRelay_MF

Derivation Point

GnSNMPAppDerPt includes the functionality that is required for an application model type. GnChassisDerPt, GnDevIODerPT, and GnRelayDerPt are derived from GnSNMPAppDerPt and therefore inherit this functionality. Each also includes some specialized functionality for managing ports and boards.

If your device does not manage ports and boards and you are only interested in expanding support, use GnSNMPAppDerPt to derive your application model type.

If your device uses other MIBs to extend the functionality of MIB-II to manage ports and boards, use GnChassisDerPt, GnDevIODerPt, or GnRelayDerPt. Each of these derivation points uses model fragments that contain the attributes and intelligence to create port models. The topic titled Board and Port Considerations (see page 102) explains how to select an appropriate derivation point for your port or board model type.

More information:

Application Model Types (see page 106)
Board and Port Considerations (see page 102)

Board and Port Considerations

If you are modeling a chassis (a device with multiple modules or boards that can be inserted and removed), create the application model type from the GnChassisDerPt and GnRelayDerPt derivation points. These derivation points are used to model both the boards and ports in the device. The intelligence of these derivation points creates both board and port models.

If the device you are modeling is not a chassis, build your application model type from the GnDevIODerPt derivation point. The intelligence of this derivation point creates only port models (no boards) that are associated with the device model.

The structure and content of the relevant MIBs is important to consider. Chassis and data relay MIBs generally have a standard structure. A chassis MIB usually has a slot and board table. The index of the table represents the slot in the chassis where the board is plugged in.

A data relay MIB usually has two tables: a board table and a port table. The board table is indexed by the slot where the board is plugged in. The port table typically has two indexes: a board index and a port number. And vendors have devised several variations to the standard structures.

Port-Oriented Devices

Use the GnDevIODerPt to model port-oriented, non-chassis devices. Most MIBs for these port-oriented devices conform to the structural requirements to use GnDevIODerPt. The MIB must contain a port table, with at least one index, the port number. The derivation point executes a read_next (which is analogous to the get_next SNMP call) on this attribute. For each successful read of the index attribute, a port model with the appropriate instance ID is instantiated.

Chassis Devices

The structure of the MIBs that are associated with chassis devices is highly varied. We recommend reviewing the requirements of the GnChassisDerPt and the GnRelayDerPt derivation points. You can thus examine the variations and how they affect the modeling of the device.

GnChassisDerPt

The GnChassisDerPt is used to create an application model type that becomes the chassis manager application. This application is responsible for the creation and management of board models in the SpectroSERVER database. This chassis manager relies on three attributes (usually list attributes) for the information it needs:

- slot index
- board type
- board label

A single chassis manager application can be instantiated or managed by the main device model. The chassis manager intelligence expects the MIB to have a slot or board table that is indexed by an integer value. This value represents the slot into which a particular board is plugged. The intelligence performs a read_next on this slot index attribute. For each successful read, the intelligence creates a model in the database to represent that board. Because the intelligence can only reference one index value, all boards in the chassis require an entry in this single table of the chassis MIB.

In addition to finding the slot where a board is plugged in, the manager intelligence must determine the board type and label the board correctly. The board type and board label attributes determine this information. These attributes do not have to exist in the same table as the slot index attribute. The attributes must only exist in a table that uses the same indexing scheme as the table used to discover the boards.

The MIB can have all the board information in non-list attributes rather than in a table. In this case, the information that is supplied within the MIB applies to a single board. The slot index value is not an index into a table, but simply an integer attribute that returns the slot where the board is located. The chassis manager intelligence tests the slot index attribute. For a non-list attribute, a read is used instead of a read_next to get the board number. If the slot index attribute is not a list attribute, the board type and board label attributes are not list attributes.

GnRelayDerPt

The GnRelayDerPt derivation point is used to model the ports on a chassis. You can use this derivation point with GnChassisDerPt to create one application model. Or you can use it on its own to create an application model that is separate from the chassis manager.

The term *chassis support application* describes an application that was built using GnRelayDerPt. This derivation point provides support to the chassis manager application (such as modeling the ports for each board). Unlike the chassis manager application, multiple chassis support applications can be instantiated under the main device model. This ability lets you model a chassis whose boards support different protocols.

Although all the boards can show up in the slot table of the chassis, a MIB that corresponds to the appropriate protocol can manage the data relay component of each board. Each of these protocol-dependent MIBs must be modeled as separate application models (built from the GnRelayDerPt derivation point). The ports on each board can then be discovered and modeled.

The typical structure of a data relay MIB has two tables: a board table and a port table. Do not confuse the board table with the slot table that is used with the chassis manager. In some cases, they can be the same table. However, the board table in the data relay MIB has an entry for each board that the MIB supports. Typically the board table is indexed by the position of the board in the chassis. For example, if the data relay MIB is an Ethernet MIB, any board that supports the Ethernet protocol (typically a repeater board) has an entry in the board table of this MIB. If a FDDI board is plugged into the chassis, the board creates an entry in the common slot table. However, this new board does not appear in the board table of the Ethernet MIB. Instead, it appears in the board table of the FDDI MIB.

In addition to the board table, the data relay MIB has a port table. For each port that the MIB supports, this table contains a corresponding entry. The tables often contain the status and statistical information for each port. The port table contains two indices: a board index and a port index. Because the port table contains a board index, the chassis support intelligence can associate the port models with the appropriate board models; the board index supplies the mapping of a port to a board.

GnDataRelay_MF is the model fragment within the GnRelayDerPt derivation point, which contains the attributes and intelligence to model the ports of each board and associate those port models with the appropriate board model. The GnDataRelay_MF model fragment intelligence works with only one board table and one port table. This requirement matches the typical structure of a data relay MIB. If your data relay MIB contains sets of tables–for example, a set of board and port tables for each of the major protocols–you must separate these MIB tables or groups into separate model types. Use each model type as a base for the appropriate application that is built with the GnRelayDerPt.

In some cases, the data relay MIB lacks the typical structure: both a board table and a port table, with the port table indexed to provide the physical mapping of ports to boards. For example, the chassis device uses a MIB with a different indexing scheme for accessing the port information. The FDDI MIB indexes the port table by the SMTIndex and the PortIndex. The SMTIndex is not used to identify the board where the FDDI port is physically located.

This situation can also be created if a vendor reuses a MIB from another device. The original device that the MIB was designed to manage was a port-oriented device (no boards, only ports). The vendor supplies the same functionality in a board that can be plugged into its chassis, and has used the original MIB to manage the ports on that board. The port table does not contain a board index; the device does not identify which board has a given port.

In such a case, implement the DataRelay_MF model fragment functionality as you would with a port-oriented device.

Application Model Types

Complete the following tasks when you create an application model type:

- Import the required MIBs.
- Derive the application model type.
- Set up application model discovery.
- Set the model name.
- Map the model fragments.
- Set the model type flags.

Use the Model Type Editor to accomplish each of these tasks. The following sections explain why these tasks are required.

Note: For more information, see the *Model Type Editor User Guide*.

Required MIBs Import

When you create an application model type, in some cases the MIB model type already exists in CA Spectrum. Or you might need to provide access to the new MIB. To provide access to the new MIB, you have two options:

- Use the Model Type Editor to import the MIB directly into the new application model type.
- Create a MIB model type.

If the MIB will be derived into multiple model types, consider deriving the MIB into a separate model type that can be used as a derivation point. The attribute IDs can then be maintained across the model types. Organizing this MIB model type under a new or existing vendor model type maintains database organization.

To create a MIB model type, derive a new model type from GnSNMPMibDerPt. Import the compiled MIB, and supply the SMI (Structure of Management Information) path.

Note: If the wrong SMI Path is used, the Model Type Editor does not produce an error. However, when you view imported attributes, the OID Prefix value is incorrect.

More information:

New Device Model Type Design (see page 90)

Application Model Type Derivation

To derive an application model type, use the Model Type Editor to set the GnSNMPAppDerPt model type as the current model type. Then create a new derived model type.

After you have created the new application model type, add any MIB model type that you created as a base model type to the new application model type.

The new application model type now contains two base model types:

- GnSNMPAppDerPt model type
- custom MIB model type

Application Model Discovery

When a device model for a specific device is instantiated, CA Spectrum queries the Model Type catalog. Most application model types that are derived from GnSNMPAppDerPt are queried. The query retrieves the value of the default_attr or default_attr_list attribute of each of these model types. CA Spectrum then queries those attributes on the device MIB. When a match is found between an attribute value retrieved from the application model type and the corresponding attribute value retrieved from the MIB, CA Spectrum instantiates a model of this model type.

You can use either the default_attr_list or default_attr to specify attribute IDs from attributes of a MIB model type. CA Spectrum queries the attributes whose attribute ID is contained in the default_attr or default_attr_list. If default_attr_list is used, CA Spectrum goes through the list of attribute IDs. The first supported attribute ID that is found is used to instantiate that application model to represent the MIB functionality.

Set the Default Attribute Values

The default_attr_list attribute lets you specify multiple attribute IDs, and the default_attr attribute lets you specify one attribute ID. Each attribute lets CA Spectrum identify the application model type that represents the MIB functionality.

The default_attr_list attribute is helpful when you have one device that supports a single table in a MIB rather than the entire MIB, and another device that supports other objects in the same MIB, but not in the particular table that the other device supports. In this scenario, use the default_attr_list attribute to specify multiple attribute IDs. This step ensures that the application model type that represents the MIB is instantiated for both devices even though they do not support the same MIB objects.

Set the default_attr or default_attr_list in all application model types. When choosing a value, we recommend using an attribute from the MIB model type that represents a mandatory, non-list, external MIB variable. Using such an attribute is especially important when you create a chassis application.

Specify a value for default_attr.

Follow these steps:

1. Find the MIB attribute for the application model type with which you are working.
2. Use the attribute ID of this attribute to set the value of the default_attr attribute in the application model type.

   Look specifically at the attributes of the model type that represents the MIB. You can find the attribute IDs of the attributes of a model type on the Attributes tab in the Model Type Editor.

Specify values for default_attr_list.

Follow these steps:

1. Find the MIB attributes for the application model type with which you are working.
2. Use the attribute IDs of these attributes to specify values in the default_attr_list attribute in the application model type.

   Note: Find the attribute IDs of the attributes of a model type on the Attributes tab in the Model Type Editor.
3. Set the Model_Group attribute to the decimal value of the model type handle of the application model.
4. Verify that the value of Model_Group is set appropriately.

   If Model_Group is set to 0, CA Spectrum only uses the default_attr attribute to identify the application model type that represents the MIB functionality.

Model Name Setting

Set the Model_Name attribute of the application model type to the appropriate value. By default, this value is used as the model name for any application model of this type.

Model Fragment Mapping

If your new application model type is derived from GnChassisDerPt, GnDevIODerPt, or GnRelayDerPt, use the model fragments that correspond to these model types. This practice ensures correct operation of port and board management. For a model fragment to function properly, use the Model Type Editor to map MIB attribute values from the application model type to model fragment attribute values. The model fragment gains access to information from the MIB that it uses to create and manage ports, boards, and interfaces.

For example, the boardIndex_Attr is one of the required attributes for the GnChassis_MF model fragment, which is used with the GnChassisDerPt derivation point. This attribute lets the model fragment discover the boards that are present in a chassis. The boardIndex_Attr must be set to the index attribute value in the board (group) table of the chassis or repeater MIB. The index attribute usually returns an integer value or a series of values that represents a board number.

Certain derivation points have associated model fragments. The attributes that are associated with that model fragment are available to any model type that was based on these derivation points. To gain the functionality of a model fragment that is not included with one of your base model types, include that model fragment as a base model type.

Model Type Flags Setting

When creating an application model type, set the value of a few different flags to ensure that models of this model type work correctly. These flags are available on the Flags tab of the current model type in the Model Type Editor. Each flag represents a Boolean value and can either be selected (set to TRUE) or deselected (set to FALSE).

In most cases, set the Visible, Instantiable, and Derivable flags to TRUE.

- If the Visible flag is set to TRUE, the model type is visible to all Model Type Editor users. Otherwise, the model type is only visible to a user with the developer ID that was used to create the model type.
- If the Instantiable flag is set to TRUE, you can instantiate a model of this model type in OneClick.
- If the Derivable flag is set to TRUE, this model type can be used as a base for other model types.

The No Destroy, Unique, and Required flags are typically set to FALSE.

- If the No Destroy flag is set to TRUE, users cannot destroy a model of this type in OneClick.
- If the Unique flag is set to TRUE, only one model of this model type can be instantiated in OneClick.
- If the Required flag is set to TRUE, a model of this model type must always exist in the SpectroSERVER database.

Modeling Ports and Boards

When you create application model types from GnChassisDerPt, GnDevIODerPt, and GnRelayDerPt, these applications create the port and board models that are required to represent your device. CA Spectrum generally uses two model types to model these boards and ports: GnModule and GnPort. You can derive new model types from these model types for customization purposes.

In OneClick, you can view the ports for a device on the Interfaces tab in the Component Details panel. To view the boards for a device, use the Locater tab to search for the boards by model type name.

Modeling Boards with GnModule

Typically a board is modeled for one reason: to be a container for the port models that are physically located on it. In the chassis support of GnSNMPDev, the GnModule model type models many different types of boards.

Derive all new board model types from the GnModule model type. Two GnModule attributes help to define the type of board that a particular model represents:

gnType

This attribute provides the board type as read from the chassis slot table. When each GnModule model type is instantiated, the chassis manager intelligence supplies the gnType attribute.

gnName

This attribute is supplied by the chassis manager, which uses information in the chassis slot table when the board is first created.

Modeling Ports with GnPort

Port models are very similar to board models. GnSNMPDev provides one port model type that is sufficient for most modeling needs. The GnPort model type is the default model that is used to model ports using the GnSNMPDev chassis support.

Port and Board Model Information

This following information is not necessary for modeling your ports and boards. We provide this information to help you understand how the information for each board and port is read and displayed in OneClick.

All external attributes that are associated with the boards and ports are read through the application models that support the board and port models. The application models are used because they contain the MIB model types and thus the external attributes that are associated with the boards and ports.

In OneClick, you can view the ports for a device on the Interfaces tab in the Component Details panel. To view the boards for a device, use the Locater tab to search for the boards by model type name.

How to Add Support for Additional Traps

CA Spectrum notifies you about significant occurrences on your network using traps (alerts from SNMP-compliant devices), events, and alarms.

- An *alert* is an unsolicited message sent out by a managed node on a network. A more specific definition of an alert depends on the management protocol that is used to report the alert. In general, CA Spectrum uses SNMP as the management protocol to communicate with network devices. Alerts that an SNMP-compliant device generates are called *traps*. CA Spectrum receives traps and converts them to events for further processing.
- An *event* is an object in CA Spectrum that indicates that something significant has occurred within CA Spectrum itself or within the managed environment. Events always occur in relation to a model. When a managed element on the network generates an alert, this alert is mapped to a CA Spectrum event in the appropriate AlertMap file. The event is then generated and takes on the event code that is specified in the AlertMap file.
- An *alarm* is an indication that a user-actionable abnormal condition exists on a model. A model usually detects an abnormal condition when an event occurs, and the EventDisp file indicates that an alarm is generated.

When you create a model type, typically you add support for additional traps, events, and alarms. You can do this using the MIB Tools application and the Event Configuration application in OneClick. The high-level process is as follows:

1. Enable the OneClick preference that lets you select whether to install or to export the event and alarm support files from MIB Tools:
   a. Click View, Preferences in the OneClick Console.

      The Set Preferences dialog opens.

   b. Expand the MIB Tools folder in the left panel and select Show Advanced Map Options.

   c. Select Yes from the drop-down list.

   Enabling this option lets you use MIB Tools to export the files that support trap, event, and alarm processing to a user-defined directory. You can then package the files in your new management module.
2. Identify the MIB that contains the desired trap definitions.
3. In MIB Tools, import the MIB into the MIB Tools database.
4. Also in MIB Tools, map the traps to events, and specify the events that generate alarms (and the severity of the alarms).

5. While you are still in the Assign Trap Alarms dialog in MIB Tools, take the following steps:

   a. Under Advanced Options, select Export Trap Support.

   b. For Starting Event Code, enter the event code for the first trap that you have mapped.

      The event code is a 4-byte integer that is expressed in hexadecimal format. The first 2 bytes contain the developer ID, and the last 2 bytes identify the event with a unique number. You specify the event code for the first trap. The codes for the remaining traps are assigned sequentially based on the first.

      Note: To identify your custom event codes in OneClick, and to prevent potential conflicts with other CA Spectrum event codes, we recommend using a starting event code that begins with your CA-assigned developer ID.

   c. For Directory, click Browse, navigate to the directory where the event and alarm support files are exported , select the directory, and click Open. For example, browse to C:\win32app*<vendor_name>*.

6. Click OK in the Assign Trap Alarms dialog.

   MIB Tools creates the appropriate event and alarm support files and exports them to the directory you specified.

7. In Event Configuration, complete the configuration of the events and alarms.

   For example, specify the symptoms, probable causes, and recommended actions for the alarms. These messages are displayed in OneClick when the alarms are generated.

   You can also specify event processing for one or more events, such as logging the event, using the event to clear an alarm, or generating another event using event rules.

   In addition, you can customize the default event message that is displayed in OneClick when the events are generated.

Note: For more information, see the *Event Configuration User Guide*.

More information:

Distributing a New Certification (see page 112)

Distributing a New Certification

After you have created and customized model types, use the CA Spectrum Extension Integration (SEI) Toolkit to create a virtual CD (VCD) for distributing the new model types to other CA Spectrum hosts.

The SEI toolkit includes command-line tools for creating required files and for assembling and packaging extensions into a management module that you can distribute. The toolkit helps you create a management module that is compatible with software from CA and other third-party developers. It lets you install a module in your existing CA Spectrum environment with minimal installation or integration issues.

Note: For more information about the CA Spectrum Extension Integration toolkit, see the *CA Spectrum Extension Integration (SEI) Developer Guide*.

Index

A

AdminOIDToModelClassMap • 12
alarm • 29, 111
alert • 29, 111
AlertMap files • 76
and RFCs • 51
Attribute Support table • 83
attributes
    support conflicts • 82
    support for • 66, 83

B

boardIndex_Attr • 108

C

CDP • 11
CiscoCDPApp • 11
Contact Criteria • 74
creating
    custom vendor folders • 72
    trap support • 77

D

deleting
    custom vendor folders • 73
    mappings • 44
Derivable flag • 91, 109
derivation points • 100
Desc_Key_Word • 98
device certification • 12, 34
Device Certification
    about • 33
    entries • 38
    mapping • 36, 37, 42, 44, 46
    starting • 34
    table • 35
Device Type attribute • 34, 95
DSS and MIB Tools
    best practices • 76, 82
    defined • 82
DSS environments • 46, 59, 82, 83, 85

E

editing
    custom vendor folders • 72
event • 29, 111
export
    Attribute Support table • 70

F

fault-tolerant environments • 49

G

GnChassis_MF • 108
GnChassisDerPt • 102
GnDevIODerPt • 102
GnModule • 110
GnPort • 110
GnRelayDerPt • 102
GnSNMPAppDerPt • 102

I

inference handlers • 100
Instantiable flag • 91
InternalOIDToModelClassMap • 12

M

Map tab • 59, 79
MIB support • 12
MIB Tools
    and synchronization • 84
    browser • 56
    contact a device with • 74
    exporting queries • 71
    Hierarchy table • 52, 56
    open • 53
    open with device context • 53
    overview • 53
    Results table • 70
    Results table; Results table, MIB Tools; • 70
    search MIBs • 56
MIBs
    defined • 51, 52
    MIBs, delete • 63
    moving • 73

organization • 52
searching for • 75
model class
lock • 12, 32
model fragments
about • 100
mapping • 108
Model Type Flags • 109
modeling catalog • 37

N

No Destroy flag • 91, 109

R

read_next • 102
Required flag • 91, 109
RFC 1155 • 51
RFC 1213 • 51

S

SNMP GET • 67
SNMP GET_NEXT • 56, 67
SNMP SET • 56, 67, 68
SpectroSERVERs
multiple in MIB Tools • 82
SPECTRUM Modeling Information subview • 32
star icon • 72
sysDescr • 98
sysObjectID • 36, 37, 42, 43, 98
SysOIDVerifyList • 98
System_OID_Verify • 98

T

Trap Support table • 59, 77, 79, 83
traps
advanced mapping option • 80
and partial mappings • 79
custom mappings • 78, 79
disposition conflicts • 85
remove custom mappings • 79
resolving disposition conflicts • 85
support for • 76, 77

U

Unique flag • 91, 109
unregistered devices • 43

V

VCD • 112
vendor folders • 72, 73
VendorIDVerifyList • 92, 95
VendorOIDVerifyList • 92
Virtual CD • 112
Visible flag • 91, 109

W

watches • 31