CA Spectrum®

Modeling Gateway Toolkit Guide

Release 9.4

CA Technologies Product References

This document references the following CA Technologies products:

- CA Spectrum® (CA Spectrum)
- CA Spectrum® Modeling Gateway Toolkit (Modeling Gateway)
- CA CMDB

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: Modeling Gateway Overview

This section contains the following topics:

Modeling Gateway Prerequisites

Before you use the CA Spectrum Modeling Gateway toolkit, make sure that you:

- Have had significant exposure to CA Spectrum.
- Read the *Concepts Guide.*
- Have a working knowledge of XML.
- Understand the concept of a Document Type Definition (DTD).
- Have a detailed understanding of the network topology you are importing.
- Can use UNIX or Windows to navigate through the file system, copy and delete files, and create and edit text files.

About the Modeling Gateway Toolkit

The CA Spectrum Modeling Gateway toolkit lets integrators import and export network topology data into and out of CA Spectrum. The toolkit includes a Document Type Definition (DTD) that defines XML elements and attributes. The toolkit also includes a resource file that defines CA Spectrum syntax and what information to import or export.

For a topology import, using the DTD elements, you can create an XML file that describes devices, ports, and connections on your network. This XML file can create new topology data in CA Spectrum, update existing data, or can destroy data that is no longer correct. Additionally, the elements and attributes that are used in the XML syntax can be expanded and customized to suit the needs of most integrations.

The toolkit also lets you use comma-delimited ASCII text files to import Frame Relay or ATM connections. You can also import this connection information using the XML functionality that was mentioned previously.

Once the network topology data exists in CA Spectrum, you can manage these devices like any other models that are created manually or by Discovery. You can view the results of the import, as well as any diagnostic information about each import.

The Modeling Gateway toolkit also lets you export topology information and configuration settings from CA Spectrum using an XML file. The information can then be imported into a specified SpectroSERVER through the Modeling Gateway.

Populating CA Spectrum with dynamic network topology information on an ongoing basis was previously a difficult task. Discovery and manual modeling are not suited to the constant updates necessary in a changing environment. Modeling connectivity using Discovery can also be a challenge with various physical infrastructures, such as those found in these environments:

- Cable MSO (Multi-Service Operator)
- ATM (Asynchronous Transfer Mode)
- Frame Relay

CA Spectrum Modeling Gateway is an effective solution for these problems.

Import Architecture

For an import, during the import integration process you take data from the third-party database and create an input file. Depending on the content, this input file can be an XML file or a comma-delimited ASCII file. The XML input file gives you the widest range of import options and is the main focus of this guide. The comma-delimited file lets you create connections for Frame Relay and ATM circuits.

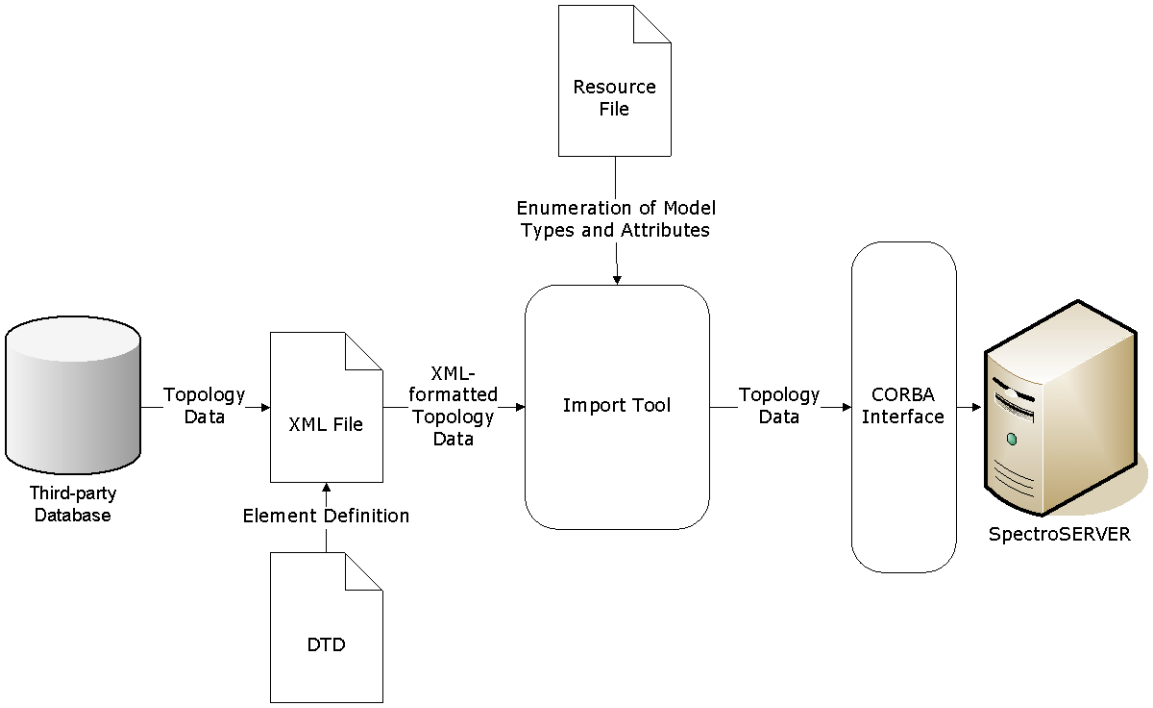
When creating an XML input file, work with the provided Document Type Definition (DTD) file and the .modelinggatewayresource.xml file. The DTD defines the XML elements, attributes, and their associated syntax rules. The .modelinggatewayresource.xml file shows which CA Spectrum model types and attributes are available for use. This file relates the CA Spectrum model type names and attribute names with the unique hexadecimal identifier that CA Spectrum uses for that model type or attribute. The .modelinggatewayresource.xml file can be customized to suit the needs of your specific integration.

Once you create the first input file, it can act as a template for multiple data sets representing the same type of input. For example, you can create an XML file for importing devices and can use this file repeatedly by substituting the device-specific topology data.

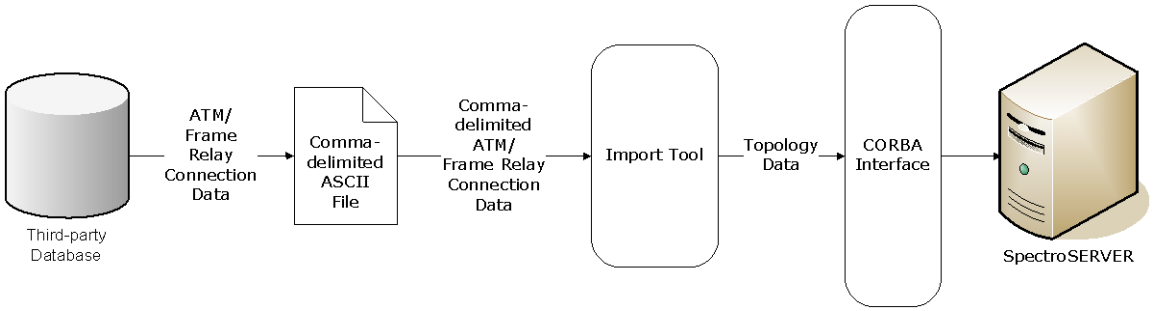
The modelinggateway tool is a command-line utility that reads the network topology information from the input file and sends the data to the SpectroSERVER database. With the data from the XML file, the import tool can create, destroy, and update connections, devices, and container models. The tool can also be used to export data from CA Spectrum.

The CA Spectrum Modeling Gateway also provides mechanisms to verify the safety and accuracy of each database import. For example, it can maintain an audit trail that includes a record of each creation, deletion, association, and update made. You can view information about the import within OneClick. CA Spectrum reports the error by generating an event when any type of critical failure occurs during the import process. All errors and their possible causes are logged in an error log file. You can also turn on a debug log which can help you locate the source of problems or inaccuracies.

The following illustration shows how the Modeling Gateway uses an XML file for importing data. The data flows from the third-party database and is formatted in the XML file using the DTD and .modelinggatewayresource.xml for syntax purposes. The import tool then interprets the XML file and sends data into CA Spectrum through the CA Spectrum CORBA interface.



The following illustration shows how the Modeling Gateway uses a comma-delimited ASCII text file to import Frame Relay and ATM connection information.



Export Architecture

For export, using the SpectroSERVER as a resource, Modeling Gateway can export topology and configuration data into an XML file. This XML file can then be integrated with a third-party tool or reimported into another SpectroSERVER. For example, the file can be reimported for a CA Spectrum partition or landscape handle change.

Chapter 2: Import Topology Data into CA Spectrum

This section contains the following topics:

How to Import With Modeling Gateway

You can import the third-party topology data into CA Spectrum using Modeling Gateway. Initially, extract the topology data from the third-party database and format the data in an input file.

To import third-party topology data into CA Spectrum using Modeling Gateway, perform these tasks:

1. Extract Topology Data.

   Extract the network topology data from the third-party database. Since each database system is different, see the documentation for the database you are working with to complete this step.

2. Format the Data in an Input File (see page 14).

   To format the data for the import tool, create an XML or a comma-delimited input file.

3. (Optional) Move the modelinggateway Tool.

   Note: To run the modelinggateway tool on another server, move the modelinggateway tool and all of its support files to that server. For more information, see the *Distributed SpectroSERVER Administrator Guide*.

4. Run the modelinggateway Tool (see page 31).

   Once the input file is created, use the import tool to send the data into CA Spectrum.

5. (Optional) Import Comma-delimited Files (see page 33).

   You can import Frame Relay and ATM connection data from the OneClick interface without the import tool.

6. View Import Information (see page 33)

   Verify the progress and results of the import in OneClick.

Note: Do not use Modeling Gateway to migrate models from one version of CA Spectrum to another. The methodology that is used to identify and model an entity can differ between CA Spectrum versions. Therefore, do not use Modeling Gateway to import any XML files that are exported from a different version of CA Spectrum.

Format the Data in an Input File

Two types of input files are used to import data using the Modeling Gateway: XML files and comma-delimited files. XML input files can be used to create or destroy models and connections, and update attribute values and connectivity information. The syntax in the DTD provided with the Modeling Gateway defines the elements to be used in the XML input file. Comma-delimited files can only be used to create ATM and Frame Relay connections. The following sections outline how to create each of the types of input files.

XML Input Files

To understand the elements that create an XML input file, be familiar with the process CA Spectrum uses to model a network infrastructure. The following section provides an overview of this process and discusses how it applies to the XML elements used in an XML input file. If you are comfortable with these concepts, you can skip this section and can go to XML Input File Syntax (see page 17).

More information:

XML Input File Syntax (see page 17)

Hierarchical Views

A view in CA Spectrum is a way to organize data so it can be displayed or manipulated. The hierarchical views represent ways to structure your network data. When structuring your network data in the XML file, you choose from elements that represent each of the hierarchical views. The two types of hierarchical views are Topology and Location.

Topology View

The Topology view is really an abstraction of networking components. When working with this view, you represent the physical or logical components of your network and group these components with logical connectivity in mind. You can also choose to represent connections graphically using pipes that show how devices are connected at the port or device level. In OneClick, this view appears as the Universe topology.

Note: For more information about the topology views available in OneClick, see the *Modeling and Managing Your IT Infrastructure Administrator Guide*.

Location View

The Location view organizes your network data by physical location. Using this view, you can depict your network in terms of geography. You can start with your global offices. Then, go right to the wiring closet on each floor of each building in each region where your offices are located. In OneClick, this view appears as the World topology.

Note: For more information about the topology views available in OneClick, see the *Modeling and Managing Your IT Infrastructure Administrator Guide*.

Models and Model Types

Numerous model types are predefined in CA Spectrum. When model types are instantiated in the CA Spectrum interface to represent a specific network entity, they are referred to as models. The two major categories of model types are:

- Intelligent model types
- Container model types

Intelligent model types can be instantiated to represent actual devices that operate on the network. They have IP and MAC addresses, and CA Spectrum can communicate directly with these devices using SNMP. Container model types are instantiated into models that are primarily used to group models together.

Models can be grouped based on the type of hierarchical view being used. For example, you can use the Topology view to create a LAN model that groups certain devices on a segment of your network. Or, you can use the Location view to create a Room model that groups the devices together in one room of your building.

A container model can contain other container models, intelligent models, or both, depending on the specific model type. For example, a network container model could contain an intelligent model to represent a router. The network container model could also contain a LAN container model to represent a range of IP addresses. On the other hand, a Building model can only contain container models; for example, a Floor, a Section, or a Room.

The elements in the DTD let you depict your network topology using any of the hierarchical views and their respective container model types. You can also use any of the instantiable intelligent model types. Intelligent model types are not dependent on the type of hierarchical view used.

Note: You can specify the model handle rather than the model type to identify a model, if desired. When specifying a model handle, Modeling Gateway ignores any other model identifiers, and it uses only the model handle to identify the model.

Not all model types that are defined in the CA Spectrum knowledge base can actually be used to create a model in OneClick. Some are used as base model types from which other model types are derived.

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

Modeling Methods

Two methods are used to model devices in CA Spectrum. The first method is to use the IP address or the DNS name of the device. With this information, CA Spectrum contacts the device and creates a model using the model type that best represents the functionality of the device.

The second method is to provide a model type for the device model creation. You still must provide an IP address or a DNS name so CA Spectrum can communicate with the device. However, your chosen model type or model handle is instantiated regardless of the CA Spectrum assessment of the device functionality.

To create a nondevice model, like a container model, a model type and model name must be provided for the import.

CA Spectrum Attributes

Each model type has a set of associated attributes. Each attribute describes the model type in some way. The attributes in an instantiated model take on values that reflect the device that the model represents and describe the current state of the model. For example, the model type Host_Sun has the attribute IPAddress. If a model of the type Host_Sun is instantiated, the value of this attribute reflects the IP address of the device that the model represents.

XML syntax also uses the term attribute. The XML attributes describe more information about an element. In CA Spectrum Modeling Gateway XML syntax, some XML attributes are used to give value to CA Spectrum attributes.

Note: Attributes that CA Spectrum defines are referred to as CA Spectrum attributes; the generic XML attributes are referred to simply as attributes.

XML Input File Syntax

Use the syntax rules that are defined in the DTD file when you generate the XML file. The following section provides an overview of the functionality of each of the DTD elements.

The following explanations and examples do not cover all the attributes of each element. For a complete reference on each element and its attributes, see Document Type Definition Elements (see page 43).

Root Element

The elements that are defined in the DTD exist in a hierarchical structure that parallels the network representation within CA Spectrum. The root element that must be used with each XML import file is the Import element. XML syntax rules specify that the root element is the outermost element and denotes the beginning and end of the XML file. Therefore, the Import element surrounds the rest of the XML elements that are used in your document.

Model-Oriented Elements

The model-oriented elements define physical or logical components of your network. They are container-type elements that are used to create models which define logical ways of grouping network elements. Grouping is based on the type of CA Spectrum hierarchical view they are in. Each of these container-type elements can exist in one of the specific hierarchical views.

- Topology_Container
- Location_Container
- Device
- Schedule
- Port
- Connection
- GenericView_Container

Topology_Container

The Topology_Container element creates a model that groups other models according to physical or logical topology. The Topology_Container element creates container models, so use the model_type attribute or a model handle to identify the specific container you want to use. An enumeration of possible model_type values is in the DTD. A LAN is an example of a Topology_Container model_type value. You specify the name of the Topology_Container using the name attribute. The name and model_type attributes uniquely identify the created model. If you specify a model handle, the name and model_type attributes are ignored. Topology_Containers can contain other Topology_Container elements, devices, or connections. The Topology_Container models are always placed in the OneClick Topology view.

Note: By default, the name and model_type attributes give values to the CA Spectrum attributes Model_Name and Modeltype_Name. However, you can change which CA Spectrum attribute the name attribute gives value to by editing the .modelinggatewayresource.xml file. Whatever new CA Spectrum attribute is chosen (along with the model type) is used to identify uniquely the container. This change lets two containers have the same model name.

Location_Container

The Location_Container element groups other models according to physical or geographical location. A Building and a Room are both examples of Location_Container element model type values. The Location_Container element creates container models, so use the model_type attribute or a model handle to identify the specific container you want to use. An enumeration of possible model_type values is in the DTD. You specify the name of the Location_Container using the name attribute. The name and model_type attributes uniquely identify the created model. If you specify a model handle, the name and model_type attributes are ignored.

Note: By default, the name and model_type attributes give values to the CA Spectrum attributes Model_Name and Modeltype_Name. However, you can change which CA Spectrum attribute the name attribute gives value to by editing the .modelinggatewayresource.xml file. Whatever new CA Spectrum attribute is chosen (along with the model type) is used to identify uniquely the container. This change lets two containers have the same model name.

Device

The Device element defines a device on the network. This element is used with other elements to create, update, or destroy an instance of a device model in CA Spectrum. When working with an SNMP device, provide a valid and unique IP address or DNS name to identify uniquely the device using the ip_dnsname attribute. This unique identification lets CA Spectrum communicate with the device and select the most appropriate model type, which is based on the device functionality. Set the ip_dnsname to a valid string. If ip_dnsname is invalid or not contactable, the device model creation can fail. If model_type is provided with an invalid or noncontactable ip_dnsname, a device model is still created with the specified model type. However, the device model is not activated to provide any valid network information or status. Possible model_type values for devices are enumerated in the .modelinggatewayresource.xml file.

Schedule

The Schedule element defines when a device model is put into maintenance mode. When a device model is in maintenance mode, management traffic to the device and its components is suspended. Suspending traffic prevents CA Spectrum from generating any events or alarms on the device model while you are performing maintenance on the device.

Port

Ports are automatically created for a device when you create the device model. The Port element lets you modify some CA Spectrum port attribute values, or specify a port-level connection. You can specify different kinds of ports, including a Frame Relay or ATM circuit. To identify the port on the device, provide the values for the identifier_name and identifier_value attributes. The possible values for identifier_name are enumerated in the DTD. The identifier_value is the value of the identifier that the identifier_name attribute chooses. A Port element must always be specified as a child of a Device element.

Connection

The Connection element defines a physical or logical connection between two devices, including WAN link connections, and therefore must contain two Device child elements. If a Port element is specified in the Device element, the connection is resolved on the specified port for that device. If the Device element does not specify a Port element, CA Spectrum Discovery tries to determine the ports in the connection to be resolved.

GenericView_Container

To create a container model in a Generic view, use the GenericView_Container element. Both the GenericView and GenericView_Container elements are used to create a customized view. Therefore, as the integrator, you decide when or how to use this container.

Task-Oriented Elements

The rest of the elements that are defined in the DTD are task-oriented elements. These elements and their attributes help define the type of action the input file generates. Using them, you can create new topology information, update, overwrite, or delete existing topology information. An individual input file can use zero or one of each of these elements, except for the Connection element. You can use as many Connection elements as necessary.

- Topology
- Location
- GenericView
- Connection
- Update
- Destroy

New Topology Data

The task-oriented elements define what action you would like to take with your XML file. Use the Topology and Location elements when you would like to create new network topology data in CA Spectrum. These elements define the hierarchical view where you would like to create the data. You can then use the corresponding model elements as child elements to create models for the network entities. To customize a view for your specific integration needs, use the GenericView element.

Create a Location View

You can create your topology information in the Location view for viewing in the World topology in OneClick. To create this information in the Location view, construct your XML file using the Location element inside the Import root element. The Location element can contain Location_Container elements to create a specific container model, or Device elements to create device models.

Example: Site Container in Location View

The following example creates a Site container in the Location view and a device within that container.

```
<?xml version = "1.0" standalone = "no" ?>
<!DOCTYPE Import SYSTEM ".modelinggateway.dtd">
<Import>
        <Location>
                <Location_Container  model_type = "Site" name = "My_Town" >
                        <Device ip_dnsname= "10.253.9.18"
                                community_string="public"/>
                </Location_Container>
        </Location>
</Import>
```

The Import element is the root element and is always contained in the input file.

The Location element indicates that you are creating models in the Location view.

The Location_Container element creates a container model. This model is a logical component rather than a physical component of the network. Therefore, CA Spectrum cannot contact it and cannot define the model type using an IP address or a DNS name. To indicate the type of container model to create, provide a value for the model_type attribute and name attribute. Possible model_type attribute values are listed in the DTD and in the following section: Location_Container (see page 56). The name attribute is required and must specify a unique name for the model.

Note: When you specify a model handle, the model handle is used to identify the container model. Therefore the name and model_type attributes are ignored, if provided.

The Device element creates a model inside the Site Location_Container model. The ip_dnsname attribute is a required attribute for the Device element. If a device can be contacted, CA Spectrum uses the IP Address or the DNS name to find the device.

For a complete reference of these elements and their possible attributes, see Document Type Definition Elements (see page 43).

More information:

Document Type Definition Elements (see page 43)
Create Connections Using Discovery (see page 24)
Create Connections Using the Connection Element (see page 24)

Create a Topology View

You can import your network data into the Topology view for viewing in the Universe topology in OneClick. To import into the Topology view, construct your XML file using the Topology element inside the Import root element. The Topology element can contain:

- Topology_Container elements to create a specific type of container model.
- Device elements to create a specific type of device model.
- Connection elements to create connections between two devices.

Using the hierarchy and syntax rules that are outlined in the DTD, you can accurately express the physical and logical connectivity of your network.

Example: LAN Container in Topology View

This example creates a LAN container in the Topology view and a device within that container.

```
<?xml version = "1.0" standalone = "no" ?>
<!DOCTYPE Import SYSTEM ".modelinggateway.dtd">
<Import>
        <Topology>
                <Topology_Container  model_type = "Lan"
                        name = "Sample_LAN" Security_String = "public"
                        subnet_address= "10.253.9.0" subnet_mask =
"255.255.255.0">
                        <Device ip_dnsname= "10.253.9.18"
                                community_string="public"/>
                </Topology_Container>
        </Topology>
</Import>
```

The Import element is the root element and is always contained in the input file.

The Topology element indicates that you are going to create models in the Topology view.

The Topology_Container element creates a container model. This model is a logical component rather than a physical component of the network. Therefore, CA Spectrum cannot contact it and cannot define the model type using an IP address or a DNS name. To indicate the type of container model to create, provide a value for the model_type attribute and name attribute. The possible model_type attribute values are listed in the DTD and in the following section: Topology_Container (see page 77). The name attribute is required and must specify a unique name for the model. The other attributes that are specified are optional.

Note: When you specify a model handle, the model handle is used to identify the container model. Therefore the name and model_type attributes is ignored, if provided.

The Device element creates a model inside the LAN Topology_Container model. The ip_dnsname attribute is a required attribute for the Device element. If CA Spectrum can contact the device, the IP Address or the DNS name is used to find the device. When CA Spectrum locates the device, it determines the appropriate model type to use to create the model.

Represent the Same Device in Multiple Views

You can create an XML file that represents a device in multiple views. To create this file, we recommended that each Device element you use to create a model of this device has identical attributes and attribute values. If they are not identical, the import tool attempts to merge the attributes and values of these Device elements to create a set of consistent attributes and values. Sometimes an attribute is specified in each of these Device elements, but different values are used. In this case, the value for the last Device element that is listed in the XML file overrides all previous values for that attribute in the other Device elements that are used to create a model of that device.

Remember that some attributes have default values. For example, the default value of the attribute community_string is public. Therefore, when you specify a Device element attribute and value to represent device A, we recommended that you specify it in any other Device element that represents device A. Doing so _helps ensure that a default value for that attribute is not used to override the previously specified value.

Example: Create a Device in Both the Topology and Location Views

In the following example, device 10.253.9.16 is created in the Topology and Location views. You can see that the attributes and values that are used to describe the device is the same for both views.

```
<?xml version = "1.0" standalone = "no" ?>
<!DOCTYPE Import SYSTEM ".modelinggateway.dtd">
<Import>
<!-- Topology View import -->
        <Topology discover_connections="false" complete_topology="false">
                <Device ip_dnsname="10.253.9.16" community_string="zippo" />
        </Topology>
```

```
<!-- Location View import -->
        <Location complete_topology="true">
                <Location_Container model_type="Site" name="Durham">
                        <Device ip_dnsname="10.253.9.16"
community_string="zippo"/>
                </Location_Container>
        </Location>
</Import>
```

Create Connections Using Discovery

A CA Spectrum connection represents a physical or logical link between two devices. You can create connections two different ways in the XML input file. The first method employs CA Spectrum Discovery, using the discover_connections attribute of the Topology element. Discovery runs on the newly created device models when the discover_connections attribute is set to true. Then, Discovery establishes connectivity for these devices.

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

Example: Using Discovery to Create Connections

```
<?xml version = "1.0" standalone = "no" ?>
<!DOCTYPE Import SYSTEM ".modelinggateway.dtd">
<Import>
        <Topology >
                <Topology_Container  model_type = "Lan" name = "Sample_LAN"
discover_connections= "true"
                        Security_String = "public" subnet_address= "10.253.9.0"
                        subnet_mask = "255.255.255.0" >
                        <Device ip_dnsname= "10.253.9.18"
                                community_string="public"/>
                        <Device ip_dnsname= "10.253.9.20"
                                community string="public"/>
                </Topology_Container>
        </Topology>
</Import>
```

Create Connections Using the Connection Element

The second way to create connections is to use the Connection element, which connects devices that are already created. Connections can be specified between two ports, between a device and a port, or between two devices.

Specifying the connectivity between devices lets CA Spectrum isolate faults to the device, but specifying port-level connections is preferred. The port-level connections are a finer grade of connectivity, allowing CA Spectrum to resolve the connections and analyze faults at the port level. CA Spectrum automatically attempts to determine the ports when you specify a connection between two devices but you do not specify one or both ports that are used in the connection. If this process is successful, CA Spectrum resolves the connection to the port level.

CA Spectrum generates an error indicating a connection failure when both of these conditions apply:

- CA Spectrum is unable to determine both ports that are used in the connection.
- At least one of these devices is a manageable device.

The error is written to the error log file.

If CA Spectrum can determine only one of the ports, then the connection is resolved only to the port level on one side of the connection. The other side remains resolved to the device level.

If both devices are unmanageable devices, CA Spectrum establishes the connection at the device level.

Example: Create a Connection between Existing Ports

The following example creates a connection between two existing ports, each port belonging to a different device. The Connection element identifies both the Port and Device elements to be linked. The connection is resolved at the port level for both devices because a Port element is specified within each Device element.

```
<?xml version = "1.0" standalone = "no" ?>
<!DOCTYPE Import SYSTEM ".modelinggateway.dtd">
<Import>
        <Connection>
                <Device ip_dnsname= "172.19.57.93">
                        <Port identifier_name = "frCircuitTableInstance"
                                identifier_value="4.161"/>
                </Device>
                <Device ip_dnsname = "192.168.125.161">
                        <Port identifier_name= "frCircuitTableInstance"
                                identifier_value= "2.861"/>
                </Device>
        </Connection>
</Import>
```

The previous example specifies a connection between the DLCI ports. Because the value of the identifier_name attribute is frCircuitTableInstance, the port is identified using the OID instance value from the frCircuitTable object in the MIB. The OID instance value is specified using the identifier_value attribute.

The Connection element is contained within a Topology element or a Topology_Container element to indicate the hierarchical placement of the devices. This case does not change the results of the input file.

Important! Modeling Gateway does not report an error when you attempt to import an XML file that contains multiple Connection elements using the same Port element. A single port cannot have multiple connections. If the same port is specified in multiple Connection elements, the last Connection element in the XML file overrides all the previous Connection elements specifying that port.

More information:

Error Log (see page 35)

Create WA_Link Connections

To create WA_Link connections, use the following syntax:

```
<Connection>
    <Device ip_dnsname=10.253.9.18/>
    <Device ip_dnsmame=10.253.9.100 model_type="WA_Link">
</Connection>
```

Modeling Gateway automatically creates a WA_Segment. The link is created between the segment and the device or devices. To specify a connection between a second device and the link, add a second connection tag to the import file.

Synchronize Information Between CA Spectrum and the Third-Party Database

The Topology, Location, Topology_Container, and Location_Container elements have an attribute named complete_topology. Setting the value of this attribute to true indicates that the XML file defines all the models and connections that CA Spectrum must know about. When the XML file is imported into CA Spectrum, any models in that CA Spectrum view that are not represented in the XML file are sent to the Lost and Found. If there are subcontainers in the view, CA Spectrum refers to the value of the complete_topology attribute that is set in the element specifying the subcontainer. If the complete_topology attribute value is not specified in the subcontainer element, the value is inherited from the parent element. Thus, if the parent element has a complete_topology setting of true and the subcontainer element does not specify a setting for complete_topology, the complete_topology value for the subcontainer is also true.

When CA Spectrum imports the XML file, models are sent to the Lost and Found under these conditions:

- The models exist either directly in the view you are importing into or in the subcontainer of that view.
- The models *do not* exist in the XML file.

This behavior is useful when synchronizing the data in your third-party database with the data in CA Spectrum.

Example: Complete_Topology Set to True

In the following example, complete_topology is set to true within the Topology element. Except for models that are specified in this input file, all existing models in the Topology view would be sent to the Lost and Found. With this sample input file, only two models are specified:

- The LAN Topology_Container
- The device at IP address 10.253.9.18

If these models did not exist, they would be created. If they existed, CA Spectrum would update their CA Spectrum attribute values using the attribute values in the input file. Any other model existing in the Topology view (except for the VNM) would be sent to the Lost and Found.

```
<?xml version = "1.0" standalone = "no" ?>
<!DOCTYPE Import SYSTEM ".modelinggateway.dtd">
<Import>
        <Topology complete_topology="true">
                <Topology_Container  model_type = "Lan" name ="Sample_LAN"
                        Security_String = "public" subnet_address= "10.253.9.0"
                        subnet_mask = "255.255.255.0">
                        <Device ip_dnsname= "10.253.9.18"
                                community_string="public"/>
                </Topology_Container>
        </Topology>
</Import>
```

If the complete_topology attribute was used in the Topology_Container element instead of the Topology element, CA Spectrum would remove only unspecified models from that Topology_Container down through the hierarchy.

Update Information

To update CA Spectrum attribute and association information for existing models, use the Update element. The Update element can enclose Container elements, Device elements, and Association elements. The value of Port attributes is updated using the appropriate Device element.

Example: Update Attributes for Two Different Models and Create an Association

The following example shows an Update input file. In this case, two attributes for two separate models are updated and an association is created between the two models.

```
<?xml version = "1.0" standalone = "no" ?>
<!DOCTYPE Import SYSTEM ".modelinggateway.dtd">
<Import>
  <Update>

    <Topology_Container model_type="Lan" name="Sample"
                        model_name = "newLAN"/>
    <Device ip_dnsname= "Test1" poll_interval= "1108"/>

    <Association relation="0x10002">
       <Left_Model> <Topology_Container name="Net"
                                        model_type="Network" /></Left_Model>
       <Right_Model> <Device ip_dnsname="172.24.94.94" /></Right_Model>
    </Association>
  </Update>
</Import>
```

The first updated attribute is the model_name attribute of the LAN container model. The model name is changed from Sample to newLAN. Note the use of the following attributes: name and model_name. Both of these attributes exist to change the CA Spectrum attribute Model_Name. To identify the container model and then specify the new name of the container model using the model_name attribute, use the name attribute with the current name as the value.

Next, the example changes the value of the poll interval for the device from Test1 to 1108. Assigning a new value to the poll_interval attribute overwrites the old value.

This example also creates an association of relation 0x10002 between container model "Net" of the Network model type and device model 172.24.94.94, as long as both models exist on the SpectroSERVER.

Destroy Information

Use the Destroy element to delete container models, device models, connections, and associations. When you destroy a device, all port and application models that are associated with the device are also destroyed.

Example: Destroy a LAN Container, a Connection, and an Association

In the following example, the LAN topology container named newLAN is destroyed. All models within this container are sent to the Lost and Found, unless they are specified to be destroyed. This example also destroys a connection between two devices, Test1 and Test2, which is specified at the port level. If a Collects association exists between the container model "Net" of model type Network and device 172.24.94.94, it is destroyed.

```
<?xml version = "1.0" standalone = "no" ?>
<!DOCTYPE Import SYSTEM ".modelinggateway.dtd">

<Import>
    <Destroy>
        <Topology_Container model_type="Lan" name="newLAN"/>

        <Connection>
          <Device ip_dnsname= "Test1">
           <Port identifier_name= "ifIndex" identifier_value= "1"/>
          </Device>
          <Device ip_dnsname= "Test2">
           <Port identifier_name="ipAddress"
              identifier_value = "10.253.8.18"/>
          </Device>
        </Connection>

        <Association relation="Collects">
       <Left_Model> <Topology_Container name="Net"
                                        model_type="Network" /></Left_Model>
       <Right_Model> <Device ip_dnsname="172.24.94.94" /></Right_Model>
    </Association>

    </Destroy>

</Import>
```

Important! To destroy a model representing a device that has already been removed from the network, use the IP address of the device rather than the DNS name when specifying the ip_dnsname attribute of the Device in the Destroy element of an XML file. Once the device has been removed from the network, the DNS entry for that device no longer exists. And, the Modeling Gateway cannot identify the appropriate model to delete.

The .modelinggatewayresource.xml File

The Topology_Container, Location_Container, and Device elements have a model_type attribute that must have a value equal to a valid CA Spectrum model type. CA Spectrum uniquely identifies model types using a hexadecimal number. These hexadecimal values have been enumerated in the resource file .modelinggatewayresource.xml. This file pairs a text value for the model type with the unique hexadecimal identifier. The text values are then displayed in the DTD.

Many of the attributes that are defined in the DTD correspond to CA Spectrum attributes. CA Spectrum attributes are uniquely identified in CA Spectrum using a hexadecimal number. The .modelinggatewayresource.xml file does not use these hexadecimal values in the DTD or in the XML file. Instead, the file pairs the hexadecimal identifiers of the CA Spectrum attributes with more intuitive text-based names.

Both the ModelType element and the Attribute element of the .modelinggatewayresource.xml file can be customized.

Note: The .modelinggatewayresource.xml file is also used for exporting topology data from CA Spectrum. For more information, see Export Topology Data from CA Spectrum (see page 37).

Define Character Set Encoding

The CA Spectrum XML input file is encoded with UTF-8 by default. To import special characters or foreign languages, specify the appropriate character set encoding in the XML file header, as shown in the following example.

Example: Set the Input File Character Encoding to Greek

The following example shows how you would modify the XML file to set the character encoding to Greek:

```
<?xml version="1.0" encoding="ISO-8859-7" standalone="no"?>
<!DOCTYPE Import SYSTEM ".modelinggateway.dtd">
```

View CA Spectrum Character Set Encoding Information

If you need to determine what character set encoding CA Spectrum is using you can do so from the OneClick Administration Pages.

To view CA Spectrum character set encoding information

1. Click Administration in the OneClick home page.

   The Administration Pages open.

2. Click Character Set in the panel on the left.

   The Character Set Encoding page opens, displaying a list of encodings and their applicable languages.

Comma-Delimited Input Files

The Modeling Gateway can specify ATM and Frame Relay connectivity in an XML input file. The toolkit can also accept ATM and Frame Relay connectivity information from a comma-delimited ASCII text file. This file can be used to import information about connections between:

- Two ATM circuits
- Two Frame Relay circuits
- An ATM and a Frame Relay circuit

You can specify that a live pipe is created in OneClick to represent the connection. Multiple connections can be specified in the same input file.

Note: The device models that are involved in these connections must previously exist in CA Spectrum.

Comma-Delimited Input File Syntax

The following example shows the format that is used in the comma-delimited input file:

```
<Device_IP>, <OID>, <Device_IP>, <OID>, <CircuitName>, <CircuitID>, <Pipe>
```

Device_IP

Specifies the IP address of each device that is involved in the connection. Required.

OID

Specifies the OID instance of frCircuitTable, atmVclTable, or atmVplTable to specify the circuit link on the device.

CircuitName

(Optional) Specifies the name of the circuit involved.

CircuitID

(Optional) Specifies the ID of the circuit involved.

Pipe

(Optional) Has two possible values: CREATE_PIPE or NO_CREATE_PIPE. If the value is set to CREATE_PIPE, live pipes are created between the connections specified. If the value is set to NO_CREATE_PIPE, live pipes are not created between the connections specified. If no value is specified for this parameter, a default value of CREATE_PIPE is assumed.

Example: Specified Connection between Frame Relay Circuits

The following example shows an input file that specifies the connection between two Frame Relay circuits. A live pipe is created between these two ports.

```
172.19.57.93, 4.161, 192.168.125.161, 2.161, FR_Circuit_Name, Circuit_Id_123,
CREATE_PIPE
```

Run the modelinggateway Tool for Import

To run the modelinggateway tool for an import, use the following syntax.

Windows

```
modelinggateway.bat -vnm vnm_name -i import_file [-o outputfile] [-debug debugfile]
```

Solaris/Linux

```
modelinggateway -vnm vnm_name -i import_file [-o outputfile] [-debug debugfile]
```

-vnm *vnm_name*

Specifies the name of the SpectroSERVER host.

- i *import_file*

Specifies the name of the XML file containing the necessary input information that is compiled with .modelinggateway.dtd.

-o *outputfile*

(Optional) Logs the error information to the file named in the *outputfile* parameter. If this option is not used, the error information is logged to a file named *import_file*.log. *Import_file* is the name of the XML file.

-debug *debugfile*

(Optional) Indicates that you would like to create a debugging output file during the import process. When using the -debug option, you can provide your own debug file name for output. If you do not supply a value for *debugfile*, the debug file name defaults to the *import_file* name suffixed with ".debug."

Note: The -debug option requires disk space on the machine where Modeling Gateway is run. For example, a large debugging output file can result when the number of models in the *import_file* is large or when the device models have large interface densities.

ImportConfiguration Element

To control certain aspects of how Modeling Gateway imports data, use the ImportConfiguration element.

The ImportConfiguration element has the following syntax:

```
<ImportConfiguration
    do_not_process_pre_existing_devices_under_container_node = "false"
    import_to_primary_ss_only = "false"
    max_device_creation_threads = "50"
/>
```

do_not_process_pre_existing_devices_under_container_node

Specifies whether CA Spectrum processes devices that are found under a container element which previously exist in CA Spectrum.

Default: false

import_to_primary_ss_only

Specifies whether Modeling Gateway connects to the secondary SpectroSERVER when the primary SpectroSERVER is down.

Default: false

max_device_creation_threads

Specifies how many device models can be created and activated simultaneously.

Note: Setting this value to an amount higher than 50 can result in too much SNMP traffic.

Default: 50

Import Comma-Delimited Files

The previous section described how to import input files using the modelinggateway import tool. You can also import Frame Relay and ATM connection data from the OneClick interface.

Follow these steps:

1. Click the applicable VNM model in the OneClick Console.
2. Click the Information tab in the Component Detail panel.
3. Click Logical Connection Import to expand the section.
4. Click Import, locate the comma-delimited file containing the data that you want to import into CA Spectrum, and then click Open.

   OneClick imports the data. The Import Results dialog appears, providing you with information about the success of the import.

View Import Information

The CA Spectrum Modeling Gateway provides mechanisms to help ensure the safety and accuracy of each database import. CA Spectrum Modeling Gateway maintains an audit trail that includes a record of each creation, deletion, association, and update made. You can view data about the import within OneClick and you can also track information about import problems in the error and debug logs.

View Modeling Gateway Results in OneClick

You can verify the results of the Modeling Gateway import from the VNM model, Information tab in the OneClick Console.

Follow these steps:

1. Click the applicable VNM model in the OneClick Console.
2. Click the Information tab in the Component Detail panel.
3. Click Modeling Gateway to expand the section.
4. Review the table in the Modeling Gateway section for information about recent imports. The table contains the following information:

   Import File

   Displays the name of the import file.

   Log File

   Displays the name of the log file.

   Start Time

   Indicates when the topology import process began.

   End Time

   Indicates when the topology import process completed.

   Progress

   The progress field shows the status of a topology import that has not yet finished. The possible values for this field are:

   - Initializing
   - Identifying Models
   - Creating Models
   - Activating Models
   - Mapping Connectivity
   - Placing Models
   - Creating Connections
   - Updating Models
   - Destroying Models
   - Complete
   - Disconnected

Errors

Displays the number of errors that are generated during the import process.

Models Created

Displays the number of models that are created during the import process.

Models Destroyed

Displays the number of models that are eliminated during the import process.

Models Updated

Displays the number of models that are updated during the import process.

Connections Created

Displays the number of connections that are created during the import process.

Connections Removed

Displays the number of connections that are removed during the import process.

5. Click the Max Records set link to modify the number of import files that are listed in the table.
6. Click any of the table column headers to sort the data as needed.
7. Enter text in the Filter field to restrict the import data to specific criteria.
8. Click Update to check the status of an import as it is processing.

   The screen refreshes and displays the most recent import information available.

Error Log

All errors and their possible causes are logged in an error log file. By default, the import tool creates an error log named <*nameofimportfile*>.log where <*nameofimportfile*> is the name of your import file. You can also specify a particular name for your log file using the syntax that is specified in the section on the import tool. When the import is complete, the log file appears in the SS-Tools directory. The log file records the number of successful creations, deletions, and updates of models and connections. The log file also records each single failure that occurred during the importing process.

Chapter 3: Export Topology Data from CA Spectrum

This section contains the following topics:

About Exporting Topology Data from CA Spectrum

Modeling Gateway supports exporting topology information and configuration settings from a SpectroSERVER. The information is exported into an XML-formatted file, which can then be imported into a specified SpectroSERVER using the Modeling Gateway.

The following types of information are exported by default:

- Device elements, with configuration attributes.
- Port elements, with configuration attributes.
- Container elements, with configuration attributes.
- Connections (resolved and unresolved, WA_Link connections).
- Universe topology hierarchy.
- Layout for each view in the Universe topology, including annotations and zoom information but excluding background images.
- User models and the entire user scheme such as user-related relations, attributes, and models like LicenseRole, AccessGroup, PrivilegeRole, and UserGroup.
- Discovery configurations.
- Service Management schemes and attributes.
- Static and dynamic global collections including all the models in each global collection, all dynamic collection criteria, zoomed list, grouped list, and topology layout.

Important! The purpose of the converter tool is to partition a SpectroSERVER database across two or more SpectroSERVERs, or to change the landscape handle of a single SpectroSERVER database. If you are using the Modeling Gateway export functionality for the same purpose as the converter tool, be aware that not all information is exported. To configure behaviors and set up modeling for data that is not currently supported in the export capabilities, some manual work is required after the exported data is imported. Examples of types of data which are not exported includes, but is not limited to, Service Performance Manager (SPM) tests and Policy Manager policies.

Configure Export Settings

You can control what is exported by modifying the ExportConfiguration element in the .modelinggatewayresource.xml file. By default, all topology and modeling information under the Universe container is exported. You can modify the RootContainerToExport element to specify a different root container from which to export. All the contents of the root container and each of its subcontainers are exported.

Note: You do not need to use the DTD for exporting data; the DTD is used only for importing data.

The attributes that are exported for devices, containers, and ports are defined in the following elements respectively: DeviceExportAttributes, ContainerExportAttributes, and PortExportAttributes. Add and subtract attributes from these elements as needed.

The SpectrumConfigurationExport element controls what types of CA Spectrum configuration data are exported when the export_spectrum_settings flag in the ExportConfiguration element is set to true.

For example, the following element controls the attributes that are exported for the LostFound model:

```
<SpectrumConfigurationExport model_type="LostFound" >
     <Automatic_Model_Destruction attribute_id="0x11de1" />
     <Model_Destruction_Interval_Hours attribute_id="0x11de3" />
     <Model_Destruction_Interval_Minutes attribute_id="0x11de4" />
</SpectrumConfigurationExport>
```

More information:

ExportConfiguration Element (see page 39)

ExportConfiguration Element

To control export behavior, use the ExportConfiguration element in the .modelinggatewayresource.xml file.

The ExportConfiguration element has the following syntax:

```
<ExportConfiguration
    export_devices           = "true"
    export_containers        = "true"
    export_port_attributes   = "true"
    export_links             = "true"
    export_topology_layout   = "true"
    export_annotation        = "true"
    export_WA_Link_models    = "true"
    export_spectrum_settings = "true"
    export_user_models       = "true"
    export_service_modeling  = "true"
    export_schedules         = "true"
    export_global_collections = "true"
    export_discovery_configs = "true"
    export_from_primary_ss_only = "false"
    export_policy_manager = "true"

/>
```

export_devices

Exports device models.

export_containers

Exports container models.

export_port_attributes

Exports port attributes.

export_links

Exports device links.

export_topology_layout

Exports device and container models' x,y coordinates in the topology.

export_annotation

Exports the annotations and model group information.

export_WA_Link_models

Exports WA_Link models. If you decide not to export WA_Link models, they are treated as transparent. Wide area links between two device models are exported as a direct link.

export_spectrum_settings

Exports CA Spectrum settings such as the settings for fault isolation, Discovery, and VNM control.

export_user_models

Exports user models, user licenses, user privileges, user preferences, and all other user-related relations attributes and models.

export_servicemodeling

Exports service management schemes and attributes.

Note: For more information about Service Manager, see *Service Manager User Guide*.

export_schedules

Exports the schedules.

export_global_collections

Exports static and dynamic global collections including all the models in each global collection, all dynamic collection criteria, zoomed list, grouped list, and topology layout.

export_discovery_configs

Exports the Discovery configurations.

export_from_primary_ss_only

Specifies whether Modeling Gateway connects to the secondary SpectroSERVER when the primary SpectroSERVER is down.

export_policy_manager

Exports the Policy Manager policies. All related models, policies, rules, permissions, and templates are included.

Example:

The following example exports everything except for port attribute information. This example also tells Modeling Gateway *not* to connect to the secondary SpectroSERVER when the primary is down.

```
<ExportConfiguration
    export_devices          = "true"
    export_containers       = "true"
    export_port_attributes  = "false"
    export_links            = "true"
    export_topology_layout  = "true"
    export_annotation       = "true"
```

```
    export_WA_Link_models    = "true"
    export_spectrum_settings = "true"
    export_user_models       = "true"
    export_service_modeling  = "true"
    export_schedules         = "true"
    export_global_collections = "true"
    export_discovery_configs = "true"
    export_from_primary_ss_only = "true"
    export_policy_manager = "true"

/>
```

The modelinggateway Tool for Export

The Modeling Gateway command-line tool, 'modelinggateway,' (modelinggateway.bat on Windows) is located in the SS-Tools directory. Its syntax for export is:

Windows

`modelinggateway.bat -vnm` *`vnm_name`* `[-cmdb] -e` *`export_file`* `[-o` *`outputfile`*`] [-debug` *`debugfile`*`]`

Solaris/Linux

`modelinggateway -vnm` *`vnm_name`* `[-cmdb] -e` *`export_file`* `[-o` *`outputfile`*`] [-debug` *`debugfile`*`]`

-vnm *vnm_name*

Specifies the name of the SpectroSERVER host.

-cmdb

(Optional) Exports the contents of your SpectroSERVER in a format that can be used when integrating CA Spectrum with CA CMDB. For more information on implementing this integration, contact CA Support.

-e *export_file*

Exports CA Spectrum topology data.

-o *outputfile*

(Optional) Logs the error information to the file named in the *outputfile* parameter. If this option is not used, the error information is logged to a file named *export_file*.log. *Export_file* is the name of the XML file.

-debug *debugfile*

(Optional) Indicates that you would like to create a debugging output file during the export process. When using the -debug option, you can provide your own debug file name for output. If you do not supply a value for *debugfile*, the debug file name defaults to the *export_file* name suffixed with .debug.

Note: The -debug option requires disk space on the machine where Modeling Gateway is run. The number of models in the *export_file* affects the size of the debugging output file: the greater the number of models in the database, the larger the debug file produced.

Note: To run the modelinggateway tool on another server, move the modelinggateway tool and all of its support files to that server. For more information, see the *Distributed SpectroSERVER Administrator Guide*.

Export CA Spectrum Topology Data

Export CA Spectrum topology data using the modelinggateway tool.

Follow these steps:

To export CA Spectrum topology data, use the -e flag. For example, running the following command exports the data from the SpectroSERVER on NOC1_Spectrum into a Modeling Gateway formatted xml file named NOC1_data.xml:

```
modelinggateway -vnm NOC1_Spectrum -e NOC1_data.xml
```

Import Modeling Gateway XML file

You can import the data from a Modeling Gateway formatted XML file into CA Spectrum.

Follow these steps:

To import from a Modeling Gateway formatted XML file, use the -i flag. For example, running the following command imports the data from NOC1_data.xml into the SpectroSERVER at NOC2_Spectrum.

```
modelinggateway -vnm NOC2_Spectrum -i NOC1_data.xml
```

Appendix A: Document Type Definition Elements

This section describes the functionality of each element that is defined in the Document Type Definition (DTD). This section also provides context for each one. This section *does not* describe the XML syntax that is used in the DTD. For syntax information, see an XML reference.

Association

Syntax

Parent Elements:

- Update
- Destroy

Child Elements:

- Left_Model
- Right_Model

Rules: The Association element must contain one Left_Model element and one Right_Model element.

Usage

The Association element creates or destroys associations between models.

If the Association element is used as a child of the Destroy element, the association that is specified is destroyed. If the Association element is used as a child of the Update element, the association that is specified is created.

Attributes

relation

Specifies the name or handle of the CA Spectrum relation between the Left_Model and Right_Model in this association.

Connection

Syntax

Parent Elements:

- Topology
- Topology_Container
- Update
- Destroy

Child Elements: Device

Rules: The Connection element must contain two device elements.

Usage

The Connection element specifies a connection between two devices. The Connection element must always contain two device elements and each of these Device elements can contain zero or one Port element. If a port or ports are specified, the connection is resolved at the port level. If a port or ports are not specified, Discovery is triggered to find the port or ports for the connection.

If the Connection element is used as a child of the Destroy element, the connection that is specified is destroyed. If the connection is used in any other context, the connection is created.

Attributes

create_pipe

Indicates whether a graphical representation of the specified connection is shown in OneClick.

Default: True

Correlation

Syntax

Parent Element: Import

Child Element: Correlation_Domain

Rule: The Correlation element can contain any number of child elements.

Usage

The Correlation element represents a Correlation Manager model and is used with the CA Spectrum Service Manager. See the *Service Manager User Guide* for usage details.

Attributes

None.

Correlation_Domain

Syntax

Parent Element: Correlation

Child Elements:

- Device
- Port
- Model_Attr
- GenericView_Container

Rule: The Correlation_Domain element can contain any number of child elements.

Usage

The Correlation_Domain element is used with the CA Spectrum Service Manager. See the *Service Manager User Guide* for usage details.

Attributes

See the *Service Manager User Guide* for attribute definitions.

Attribute	Data Type	Default Value	Possible Values
name *(required)*	Character	N/A	N/A

CustomerManager

Syntax

Parent Element: SM_Service_Mgt

Child Elements:

- SM_Customer
- SM_CustomerGroup

Rule: The CustomerManager element can contain any number of these child elements.

Usage

The CustomerManager element is used with the CA Spectrum Service Manager. See the *Service Manager User Guide* for usage details.

Attributes

Note: See the *Service Manager User Guide* for attribute definitions.

Attribute	Data Type	Default Value	Possible Values
name *(required)*	Character	N/A	N/A
containment_relation	Character	Groups_Customers	N/A
model_type	Character	CustomerManager	N/A

Destroy

Syntax

Parent Element: Import

Child Elements:

- Topology_Container
- Location_Container
- GenericView_Container
- Device
- Model
- Connection
- EventModel
- SM_Service

- SM_AttrMonitor
- SM_LatencyMon
- SM_ConnectMon
- SM_SLA
- SM_Guarantee
- SM_Customer
- Association

Rule: The Destroy element can contain as many of each of these child elements as necessary.

Usage

Use the Destroy element to remove container models, device models, connections, and associations. You cannot nest elements in the Destroy element to express hierarchies or destroy hierarchies. The only hierarchy that is allowed in the Destroy element is the Connection-Device-Port hierarchy, which specifies at the port level the connection to destroy. You could destroy a container model without destroying the device models or other container models that are contained inside it. In this case, the remaining models are placed in the CA Spectrum Lost and Found.

Attributes

The Destroy element does not have any attributes.

Device

Syntax

Parent Elements:

- Topology
- Location
- Topology_Container
- Left_Model
- Right_Model
- Location_Container
- GenericView
- GenericView_Container

- Connection
- Update
- Destroy
- SM_Service
- SM_AttrMonitor

Child Elements:

- Port
- Schedule

Rules:

- **Port:** If a Device element is contained within a Connection element, only one port element is allowed. If a Device element in contained within an Update element to update ports, more than one port element is allowed. The ports are ignored when a Device element is contained in a View, a Container, or a Destroy element.
- **Schedule:** The Device element can contain one Schedule element.

Usage

To create, destroy, or update a device model, use the Device element. The Device element lets you define the device model using ether an IP address or a DNS name.

Note: If community_string and agent_port attributes are not provided during the device creation, CA Spectrum creates the device using the predefined SNMP credentials. These credentials are configured in the Modeling and Protocol Options section of the AutoDiscovery Control subview in the VNM model Information tab in OneClick.

Attributes

ip_dnsname

Specifies the IP address or DNS name of the device. If the device does not support SNMP communication, you can use a unique string here with model_type specified.

secdomain_ipname

(Optional) Specifies the IP address of a host running an SDConnector in the secure domain where the device is located.

Default: 0.0.0.0

model_handle

(Optional) Specifies the model_handle to identify an existing device model.

Note: If you provide a model_handle, the value of ip_dnsname is ignored.

model_type

(Optional) The CA Spectrum model type that is used to model your device. This device model can be any intelligent model type that is defined in the .modelinggatewayresource.xml file.

Note: If you have provided a valid IP address or DNS name, you do not need to specify a value here.

community_string

(Optional) The community string of the device.

Note: If community_string is not included, CA Spectrum uses the first SNMP Community Strings value to create the device. These values are specified in the VNM model Information tab, AutoDiscovery Control subview, Modeling and Protocol Options subview in OneClick.

agent_port

(Optional) Controls the port number that is used when communicating with the SNMP agent of a device.

Note: If agent_port is not included, CA Spectrum uses the first SNMP Ports value to create the device. These values are specified in the VNM model Information tab, AutoDiscovery Control subview, Modeling and Protocol Options subview in OneClick.

is_managed

(Optional) Puts the device model in maintenance mode, when set to true.

Default: True

poll_interval

(Optional) Specifies the time interval, in seconds, that the SpectroSERVER reads all attributes of the device model that are flagged as POLLED.

log_ration

(Optional) Specifies the number of SpectroSERVER polls of a device that occur before logging the poll results in the database.

poll_status

(Optional) Lets you disable the SpectroSERVER polls of a device by setting the polling status to false.

model_name

(Optional) Specifies the name of the model.

DeviceType

(Optional) Specifies the device type of the model.

Note: See the *Certification User Guide* for more information about device types.

reconfig

(Optional) Specifies whether Modeling Gateway sends an action to the SpectroSERVER to reconfigure the device model.

discover_connections

(Optional) Runs Discovery on any newly created device models to map automatically the model connectivity, when set to true.

EventModel

Syntax

Parent Element:

- Topology_Container
- Left_Model
- Right_Model

Child Element: None

Rule: N/A

Usage

To import the EventModel models for use with a Southbound Gateway integration, use the EventModel element. For more information about Southbound Gateway, see the *Southbound Gateway Toolkit Guide*.

Attributes

model_name

Specifies the unique name of the model that is instantiated or identified.

unique_id

Specifies the identifier that is used to define uniquely the event source that this EventModel model represents. For more information, see the *Southbound Gateway Toolkit Guide*.

model_handle

(Optional) Specifies the model_handle to identify an existing device model.

Security_String

(Optional) Specifies the security string for the EventModel.

Default: public

manager_name

(Optional) Specifies the name of the third-party application that is using the Southbound Gateway.

Note: Use the default value for any application that is not listed here.

Default: 0

1

NetMentor

2

SSM

3

Omni2000

GenericView

Syntax

Parent Element: Import

Child Elements:

- GenericView_Container
- Device

Rule: The GenericView element can contain any number of these child elements.

Usage

To create a customized hierarchical view other than the Topology view and Location view, use the GenericView element. You can modify this element to meet the needs of your integration.

Attributes

containment_relation

Specifies a relation handle that defines which CA Spectrum relation defines the containment relationship within this view.

Limits: Must be a CA Spectrum containment relation.

model_type

Specifies a model type that represents the top container model that is defined for this view. This model_type must be specified with its model handle in the .modelinggatewayresource.xml file.

name

Specifies the unique name of the instantiated container model highest in the GenericView hierarchy.

complete_topology

(Optional) Destroys any unspecified, existing container and device model in the GenericView view when set to true. Also, destroys these models in the subcontainers of that view.

GenericView_Container

Syntax

Parent Element: GenericView

Child Elements:

- GenericView_Container
- Device

Rule: The GenericView_Container element can contain any number of child elements.

Usage

To create a container model in the Generic view, use the GenericView_Container element. Both the GenericView and GenericView_Container elements are used to create a customized view. Therefore, as the integrator, you decide when or how to use this container.

Attributes

name

Specifies the name of the model that is instantiated or identified. The model_type and the name attribute are required to identify uniquely the GenericView_Container. By default, this attribute is used to set the value of the CA Spectrum model name attribute (attr id 0x1006e). However, this attribute can be changed to any other attribute in the .modelinggatewayresource.xml file. You can change the .modelinggatewayresource.xml so that the name maps to a different attribute. In this case, that new attribute (along with the model type) is used to identify the container. This behavior lets two containers have the same model name.

model_type

Specifies the CA Spectrum model type that is used to create the model. This model_type must be specified with its model handle in the .modelinggatewayresource.xml file. The model_type and the name attribute are required to identify uniquely the GenericView_Container.

containment_relation

(Optional) The name of the CA Spectrum relation that exists between the Generic_Container and the models within the container. If no value for this attribute is specified, the containment relationship of the parent model is inherited.

GlobalCollection

Syntax

Parent Element: Import

Child Elements:

- Device
- Topology_Container
- Location_Container

Usage

Represents a GlobalCollection model.

Attributes

name

Specifies a name for this global collection.

containment_relation

Specifies a relation handle that defines which CA Spectrum relation defines the containment relationship within this view.

Default: "GlobalCollect"

collectionDescription

(Optional) Describes the global collection.

Security_String

(Optional) Specifies the security string for the global collection.

Import

Syntax

Parent Element: None

Child Elements:

- Topology
- Location
- GenericView
- Update
- Destroy
- SM_Service_Mgt
- Correlation
- GlobalCollection

Rule: The Import element can contain one of each of these child elements.

Usage

The Import element is the root element, and it must be included in each input file.

Attributes

model_activation_time

Specifies the maximum number of minutes that are allowed for each device model activation.

Data Type: Character

Default: 5 minutes

Left_Model

Syntax

Parent Element: Association

Child Elements:

- Device
- Port

- Topology_Container
- Location_Container
- EventModel
- Model

Rules: The Left_Model element can contain only one child element.

Usage

The Left_Model element defines the left side model in an association.

Attributes

None.

List_Value

Syntax

Parent Element: Model_Attr

Child Element: None

Rule: N/A

Usage

To specify a CA Spectrum list attribute value, use the List_Value element.

Attributes

None.

Location

Syntax

Parent Element: Import

Child Elements:

- Location_Container
- Device

Rule: The Location element can contain any number of child elements.

Usage

To specify that you would like to create models in the Location view (World topology) of OneClick, use the Location element.

Attributes

complete_topology

(Optional) When set to true, any unspecified, existing containers and device models in the Location view, or any subcontainers, are destroyed during the import.

Default: False

Data Type: Boolean

Location_Container

Syntax

Parent Elements:

- Location
- Location_Container
- Left_Model
- Right_Model
- GlobalCollection

Child Elements:

- Location_Container
- Device

Rule: The Location_Container element can contain any number of child elements.

Usage

To create or specify a Location_Container model that is used to group models and other location containers in the Location view, use the Location_Container element.

Attributes

name

The name of the model that is instantiated or identified. The model_type and the name attribute are required to identify uniquely the Location_Container.
By default, this attribute is used to set the value of the CA Spectrum model name attribute (attr id 0x1006e). However, this attribute can be changed to any other attribute in the .modelinggatewayresource.xml file. You can change the .modelinggatewayresource.xml so that the name maps to a different attribute. In this case, that new attribute (along with the model type) is used to identify the container. This behavior lets two containers have the same model name.

model_type

Indicates the type of model you want to create. The model_type and the name attribute are required to identify uniquely the Location_Container. Possible values include:

- Country
- Region
- Site
- Building
- Floor
- Section
- Room

model_handle

(Optional) Can be used to identify models. If you provide a model_handle, the values of name and model_type are ignored.

Security_String

(Optional) Defines the requirements for access to a model by CA Spectrum users. Each security string consists of one or more Security Community entries and is assigned to models.

model_name

(Optional) To change the name of the model, use the name and the model_name attributes. The name attribute specifies the old name and the model_name attribute specifies the new name.

model_modify_author

(Optional) Writes data to the CA Spectrum attribute mdl_modfy_athr.

complete_topology

(Optional) When set to true, any unspecified, existing containers and device models in the Location view, or any subcontainers, are destroyed during the import.

Default: False

Model_Attr

Syntax

Parent Element: Correlation_Domain

Child Element: List_Value

Rule: The Model_Attr element can contain any number of child elements.

Usage

To specify CA Spectrum attributes whose values contain multiple lines of text or a list of values, use the Model_Attr element.

Attributes

attr_id

Indicates the CA Spectrum attribute ID of the attribute you are specifying.

Data Type: Character

Model

Syntax

Parent Elements:

- Left_Model
- Right_Model

Child Elements: None

Usage

To represent any CA Spectrum model, use the Model element.

Attributes

name

Specifies a name for this model.

model_type

Specifies the model type of this model.

model_handle

(Optional) Specifies a model_handle for this model. If a model_handle value is specified, the name and model_type values are ignored.

MonitorPolicy_Attr

Syntax

Parent Element:

- SM_Service
- SM_AttrMonitor

Child Element: None

Rule: N/A

Usage

The MonitorPolicy_Attr element is used with the CA Spectrum Service Manager. See the *Service Manager User Guide* for usage details.

Attributes

None.

Port

Syntax

Parent Elements:

- Device
- Left_Model
- Right_Model
- Correlation_Domain

- SM_Service
- SM_AttrMonitor

Child Element: None

Rule: N/A

Usage

The Port element is used to specify connections at the port level or to update port attributes. When updating, the parent Device element is contained within an Update element. When specifying a connection, the parent device element is contained within a Connection element.

Attributes

identifier_name

Works with the identifier_value attribute to identify uniquely the port. The identifier_name can be any one of the MIB OID names that are listed in the Possible Values column. The portID value can be used to identify a port by its Component_OID attribute (0x1006a). If the Port element represents a Frame Relay virtual circuit, use frCircuitTableInstance. If the Port element represents an ATM virtual channel or path link, use atmVclTableInstance.

The portID value can be used to identify a port by its Component_OID attribute (0x1006a). If the Port element represents a Frame Relay virtual circuit, use frCircuitTableInstance.If the Port element represents an ATM virtual channel or path link, use atmVclTableInstance. Possible values include:

- ifIndex
- ipAddress
- ifPhysAddress
- ifName
- ifAlias
- model_name
- portDescription
- portID
- frCircuitTableInstance
- atmVclTableInstance
- atmVplTableInstance

identifier_value

Specifies the value of the identifier_name selection.

model_handle

(Optional) Specifies the model_handle to identify an existing model.

Note: If you provide a model_handle, the values for identifier_name and identifier_value are ignored.

ip_dnsname

(Optional) Specifies the IP address or DNS name of the port model. If the port model does not support SNMP communication, you can use a unique string here with model_type specified.

model_name

(Optional) Specifies the name of the model you are updating.

circuit_id

(Optional) Identifies the circuit that is involved in an ATM or Frame Relay connection by ID.

circuit_name

(Optional) Identifies the circuit that is involved in an ATM or Frame Relay connection by name.

log_ratio

(Optional) Specifies the number of port model polls that occur before logging the poll results in the database.

poll_interval

(Optional) Specifies the time interval, in seconds, that the SpectroSERVER reads all attributes of the port model that is flagged as POLLED.

poll_status

(Optional) Lets an administrator disable port model polls by setting polling status to False.

Right_Model

Syntax

Parent Element: Association

Child Elements:

- Device
- Port
- Topology_Container

- Location_Container
- EventModel
- Model

Rules: The Right_Model element can contain only one child element.

Usage

The Right_Model element defines the right side model in an association.

Attributes

None.

RTM_Test

Syntax

Parent Elements:

- SM_Service
- SM_AttrMonitor

Child Element: None

Rule: N/A

Usage

The RTM_Test element is used with the CA Spectrum Service Manager. See the *Service Manager User Guide* for usage details.

Attributes

See the *Service Manager User Guide* for attribute definitions.

Attribute	Data Type	Default Value	Possible Values
name *(required)*	Character	N/A	N/A
model_type	Character	RTM_Test	N/A

Schedule

Syntax

Parent Elements: Device

Child Element: None

Usage

To create a maintenance mode schedule for a particular device model, use the Schedule element.

Attributes

name

Specifies the name of the schedule.

SCHED_Recurrence

Specifies how often the device model is put into maintenance mode.

1

Always (24x7)

2

Daily

3

Weekly

4

Monthly

5

Yearly

Default: 1

SCHED_Start_Hour

(Optional) Specifies the hour that the device goes into maintenance mode.

Limits: 0-23

SCHED_Start_Minute

(Optional) Specifies the minute the device goes into maintenance mode.

Limits: 0-59

SCHED_Start_DoW

(Optional) Specifies the day of the week the device goes into maintenance mode.

0

Sunday

1

Monday

2

Tuesday

3

Wednesday

4

Thursday

5

Friday

6

Saturday

SCHED_Start_DoM

(Optional) Specifies the day of the month the device goes into maintenance mode.

Limits: 1-31

SCHED_Start_Month

(Optional) Specifies the month that the device goes into maintenance mode.

0

January

1

February

2

March

3

April

4

May

5

June

6

July

7

August

8

September

9

October

10

November

11

December

SCHED_Duration

(Optional) Specifies the length of time the device is in maintenance mode, which is defined in seconds.

Default: 0

SCHED_Recurrence_Multiplier

(Optional) Specifies the number of recurrence units (days, weeks, months, years) that determine the length of time between the start of each scheduled maintenance mode.

Default: 1

SCHED_Daily_Repeat_Limit

(Optional) Specifies the number of consecutive days to repeat the scheduled maintenance (specified by SCHED_Start_Hour and SCHED_Start_Minute) during each recurrence period. This attribute is only applicable to a weekly, monthly, or yearly recurrence.

SM_AttrMonitor

Syntax

Parent Element: SM_Service

Child Element: None

Rules: N/A

Usage

The SM_AttrMonitor element is used with the CA Spectrum Service Manager. See the *Service Manager User Guide* for usage details.

Attributes

See the *Service Manager User Guide* for attribute definitions.

Attribute	Data Type	Default Value	Possible Values
name *(required)*	Character	N/A	N/A
containment_relation	Character	N/A	SLMMonitors SLMWatchesContainer
AttrToWatch	Character	N/A	N/A
MonitorPolicy_ID	Character	N/A	N/A
is_managed	Boolean	N/A	True False
Generate_Service_Alarms	Boolean	N/A	True False
model_type	Character	SM_AttrMonitor	N/A

SM_Customer

Syntax

Parent Elements:

- SM_CustomerGroup
- CustomerManager

Child Element: None

Rule: N/A

Usage

The SM_Customer element is used with the CA Spectrum Service Manager. See the *Service Manager User Guide* for usage details.

Attributes

See the *Service Manager User Guide* for attribute definitions.

Attribute	Data Type	Default Value	Possible Values
name *(required)*	Character	N/A	N/A
containment_relation	Character	N/A	SlmAgreesTo SlmUses
Security_String	Character	N/A	N/A
CustomerID	Character	N/A	N/A
Criticality	Character	N/A	N/A
CustomerField4	Character	N/A	N/A
CustomerField5	Character	N/A	N/A
CustomerField6	Character	N/A	N/A
CustomerField7	Character	N/A	N/A

Attribute	Data Type	Default Value	Possible Values
Contact_Name	Character	N/A	N/A
Contact_Title	Character	N/A	N/A
Contact_Location	Character	N/A	N/A
Email_Address	Character	N/A	N/A
Phone_Number	Character	N/A	N/A
Mobile_Phone_Number	Character	N/A	N/A
Pager_Number	Character	N/A	N/A
Fax_Number	Character	N/A	N/A
User_Defined_1	Character	N/A	N/A
User_Defined_2	Character	N/A	N/A
User_Defined_3	Character	N/A	N/A
User_Defined_4	Character	N/A	N/A
Secondary_Contact_Name	Character	N/A	N/A
Secondary_Contact_Location	Character	N/A	N/A
Secondary_Email_Address	Character	N/A	N/A
Secondary_Phone_Number	Character	N/A	N/A
Secondary_Mobile_Phone_ Number	Character	N/A	N/A
Secondary_Pager_Number	Character	N/A	N/A

Attribute	Data Type	Default Value	Possible Values
Secondary_Fax_Number	Character	N/A	N/A
Secondary_User_Defined_1	Character	N/A	N/A
Secondary_User_Defined_2	Character	N/A	N/A
Secondary_User_Defined_3	Character	N/A	N/A
Secondary_User_Defined_4	Character	N/A	N/A
model_type	Character	SM_Customer	N/A

SM_CustomerGroup

Syntax

Parent Elements:

- CustomerManager
- SM_CustomerGroup

Child Elements:

- SM_CustomerGroup
- SM_Customer

Rule: The SM_CustomerGroup element can contain any number of these child elements.

Usage

The SM_CustomerGroup element is used with the CA Spectrum Service Manager. See the *Service Manager User Guide* for usage details.

Attributes

See the *Service Manager User Guide* for attribute definitions.

Attribute	Data Type	Default Value	Possible Values
name *(required)*	Character	NA	N/A
containment_relation	Character	Groups_Customer	N/A
model_type	Character	SM_CustomerGroup	N/A

SM_Guarantee

Syntax

Parent Element: SM_SLA

Child Element: None

Rule: N/A

Usage

The SM_Guarantee element is used with the CA Spectrum Service Manager. See the *Service Manager User Guide* for usage details.

Attributes

See the *Service Manager User Guide* for attribute definitions.

Attribute	Data Type	Default Value	Possible Values
name *(required)*	Character	N/A	N/A
containment_relation	Character	SlmIsMeasuredBy	N/A
is_managed	Boolean	N/A	True False

Attribute	Data Type	Default Value	Possible Values
DegradedTimeViolationLevel	Character	N/A	N/A
DegradedTimeWarningLevel	Character	N/A	N/A
DownTimeViolationLevel	Character	N/A	N/A
DownTimeWarningLevel	Character	N/A	N/A
LorTimeViolationLevel	Character	N/A	N/A
LorTimeWarningLevel	Character	N/A	N/A
model_type	Character	SM_Guarantee	N/A

SM_LatencyMon

Syntax

Parent Elements:

- SM_Guarantee
- SM_AttrMonitor
- SM_Service

Child Elements:

- Topology_Container
- MonitorPolicy_Attr

Rule: The SM_LatencyMon element can contain any number of these child elements.

Usage

The SM_LatencyMon element is used with the CA Spectrum Service Manager. See the *Service Manager User Guide* for usage details.

Attributes

See the *Service Manager User Guide* for attribute definitions.

Attribute	Data Type	Default Value	Possible Values
name *(required)*	Character	N/A	N/A
containment_relation	Character	N/A	SlmMonitors SlmWatchesContainer
is_managed	Boolean	N/A	True False
DefaultMaxRTT	Character	N/A	N/A
DefaultMeasureInterval	Character	N/A	N/A
mode_type	Character	SM_LatencyMon	N/A

SM_Service

Syntax

Parent Elements:

- SM_Service
- SM_ServiceMgr

Child Elements:

- SM_Service
- SM_AttrMonitor

Rule: The SM_Service element can contain any number of these child elements.

Usage

The SM_Service element is used with the CA Spectrum Service Manager. See the *Service Manager User Guide* for usage details.

Attributes

See the *Service Manager User Guide* for attribute definitions.

Attribute	Data Type	Default Value	Possible Values
name *(required)*	Character	N/A	N/A
Criticality	Character	N/A	N/A
containment_relation	Character	N/A	SlmMonitors SlmWatchesContainer
AttrToWatch	Character	N/A	N/A
MonitorPolicy_ID	Character	N/A	N/A
is_managed	Boolean	N/A	True False
Generate_Service_Alarms	Boolean	N/A	True False
Security_String	Character	N/A	N/A
model_type	Character	SM_Service	N/A

SM_Service_Mgt

Syntax

Parent Element: Import

Child Elements:

- SM_ServiceMgr
- CustomerManager
- SM_SLA_Mgr

Rule: Only a single instance of each of these child elements can exist in SM_Service_Mgt.

Usage

The SM_Service_Mgt element is used with the CA Spectrum Service Manager. See the *Service Manager User Guide* for usage details.

Attributes

See the *Service Manager User Guide* for attribute definitions.

Attribute	Data Type	Default Value	Possible Values
name *(required)*	Character	Service Management	N/A
containment_relation	Character		N/A
model_type	Character		N/A

SM_ServiceMgr

Syntax

Parent Element: SM_Service_Mgt

Child Element: SM_Service

Rule: The SM_ServiceMgr element can contain any number of this child element.

Usage

The SM_ServiceMgr element is used with the CA Spectrum Service Manager. See the *Service Manager User Guide* for usage details.

Attributes

See the *Service Manager User Guide* for attribute definitions.

Attribute	Data Type	Default Value	Possible Values
name	Character	N/A	N/A
containment_relation	Character	SlmContains	N/A
model_type	Character	SM_ServiceMgr	N/A

SM_SLA

Syntax

Parent Element: SM_SLA_Mgr

Child Element: SM_Guarantee

Rule: The SM_SLA element can contain any number of this child element.

Usage

The SM_SLA element is used with the CA Spectrum Service Manager. See the *Service Manager User Guide* for usage details.

Attributes

See the *Service Manager User Guide* for attribute definitions.

Attribute	Data Type	Default Value	Possible Values
name *(required)*	Character	N/A	N/A
containment_relation	Character	SlmContains	N/A
is_managed	Boolean	N/A	True False
Security_String	Character	N/A	N/A
model_type	Character	SM_SLA	N/A

SM_SLA_Mgr

Syntax

Parent Element: SM_Service_Mgt

Child Element: SM_SLA

Rule: The SM_SLA_Mgr element can contain any number of this child element.

Usage

The SM_SLA_Mgr element is used with the CA Spectrum Service Manager. See the *Service Manager User Guide* for usage details.

Attributes

See the *Service Manager User Guide* for attribute definitions.

Attribute	Data Type	Default Value	Possible Values
name	Character		N/A
containment_relation	Character	SlmContains	N/A
model_type	Character	SM_ServiceMgr	N/A

Topology

Syntax

Parent Element: Import

Child Elements:

- Topology_Container
- Device
- Connection

Rule: The Topology element can contain any number of these child elements.

Usage

To create models in the OneClick Topology view (Universe topology), use the Topology element.

Attributes

complete_topology

Destroys any unspecified, existing container and device model in the Topology view during the import when set to true. Also, destroys any subcontainers of that view.

Default: False

discover_connections

When set to true, Discovery runs on any newly created device models to map automatically the connectivity of the model.

Default: False

Topology_Container

Syntax

Parent Elements:

- Topology
- Topology_Container
- SM_Service
- SM_AttrMonitor

Child Elements:

- Topology_Container
- Device
- EventModel
- Connection

Rule: The Topology_Container element can contain any number of these child elements.

Usage

To create or specify a Topology_Container model that is used to group models and other topology containers in the Topology view, use the Topology_Container element. Possible model types are listed in the following table in the model_type section.

Attributes

model_type

Indicates the type of model you want to create. The model_type and the name attribute are required to identify uniquely the Topology_Container.

- Network
- LAN
- IPClassA
- IPClassB
- IPClassC
- LAN_802_3
- LAN_803_5
- EventAdmin
- ATM_Network

model_handle

(Optional) Can be used to identify existing models. If you provide a model_handle, the values of model_type and model_name are ignored.

Security_String

(Optional) Specifies the assigned CA Spectrum security level for the model.

subnet_address

(Optional) Specifies the subnet address of the device.

subnet_mask

(Optional) Specifies the mask that determines what subnet the device IP address belongs to.

model_name

(Optional) Specifies a model name for the container model.

trapIPAddress

(Optional) For the EventAdmin models only.

x_coordinate

(Optional) Specifies the x coordinate of the model in the topology.

y_coordinate

(Optional) Specifies the y coordinate of the model in the topology.

complete_topology

(Optional) When set to True, any unspecified, existing container and device model in this Topology_Container and, any subcontainers, are destroyed during the import.

Default: False

discover_connections

(Optional) Specifies whether CA Spectrum discovers and models devices that are connected to this model.

Update

Syntax

Parent Element: Import

Child Elements:

- Topology_Container
- Location_Container
- GenericView_Container
- Connection
- Device
- Model
- EventModel
- SM_Service
- SM_AttrMonitor
- SM_LatencyMon
- SM_ConnectMon
- SM_SLA
- SM_Guarantee
- SM_Customer
- Association

Rule: The Update element can contain any number these child elements.

Usage

To update attributes for any device, container, or port subelements, use the Update element.

Note: The hierarchical specifications are not allowed when using this element, except for using the Port element within the Device element.

Attributes

None.

Appendix B: Document Type Definition File

This section contains the Document Type Definition (DTD), which defines XML elements and attributes for import.

Note: The following code might not be the latest version of this file. For the latest version of the DTD, use the actual file that shipped with your Modeling Gateway toolkit. The file is located in the SS-Tools directory and is named .modelinggateway.dtd.

```
<!-- ********************************************************** -->
<!-- The root Import element contains 0 or 1 of the Topology,     -->
<!-- Location, GenericView, Update, Destroy elements,             -->
<!-- SM_Service_Mgt, Correlation, and Global Collection.          -->
<!--                                                              -->
<!-- This element has one attribute model_activation_time which   -->
<!-- is the maximum waiting time in minutes, for each device model-->
<!-- activation. It defaults to 5 minutes.                        -->
<!-- ********************************************************** -->

<!ELEMENT Import ( ( Topology       |
                     Location       |
                     GenericView    |
                     Update         |
                     Destroy        |
                     SM_Service_Mgt |
                     Correlation    |
                     GlobalCollection )*) >

<!ATTLIST Import model_activation_time  CDATA     "5">

<!-- ************************************************** -->
<!-- The Topology element, which is used for the Topology -->
<!-- view, contains any number of Topology_Container,     -->
<!-- Device and Connection elements.                      -->
<!--                                                      -->
<!-- This element has two attributes, complete_topology   -->
<!-- and discover_connection.                             -->
<!--                                                      -->
<!-- ************************************************** -->

<!ELEMENT Topology ((Topology_Container | Device | Connection)*) >
<!ATTLIST Topology
        complete_topology    (false | true)     #IMPLIED
        discover_connections (false | true)     #IMPLIED>
```

```
<!-- ************************************************* -->
<!-- The Topology_Container element, which is used for a  -->
<!-- topology container model, contains any number of     -->
<!-- Topology_Container, Device and Connection elements.  -->
<!--                                                      -->
<!-- "model_type" and "name" are the required attributes  -->
<!-- to uniquely identify Topology_Container models.      -->
<!--                                                      -->
<!-- "model_handle" can also be used to identify models.  -->
<!-- If "model_handle" is provided, the values of "name"  -->
<!-- and "model_type" will be ignored.                    -->
<!--                                                      -->
<!-- "trapIPAddress" attribute should be only used for    -->
<!-- EventAdmin models.                                   -->
<!--                                                      -->
<!-- ************************************************* -->

<!ELEMENT Topology_Container ((Topology_Container |
                               Device |
                               EventModel |
                               Connection )*) >

<!ATTLIST Topology_Container
        name          CDATA                 #REQUIRED
        model_type          ( Network      |
                              Lan          |
                              IPClassA     |
                              IPClassB     |
                              IPClassC     |
                              LAN_802_3    |
                              LAN_803_5    |
                              EventAdmin   |
                              ATM_Network  ) #REQUIRED

        model_handle          CDATA          #IMPLIED
        Security_String       CDATA          #IMPLIED
        subnet_address        CDATA          #IMPLIED
        subnet_mask           CDATA          #IMPLIED
        model_name            CDATA          #IMPLIED
        trapIPAddress         CDATA          #IMPLIED
        x_coordinate          CDATA          #IMPLIED
        y_coordinate          CDATA          #IMPLIED
        complete_topology    (false | true)  #IMPLIED
        discover_connections (false | true)  #IMPLIED >
```

```
<!-- ************************************************** -->
<!-- Location element, which is for Location view,      -->
<!-- contains any number of Location_Container and      -->
<!-- Device elements                                    -->
<!--                                                    -->
<!-- This element has attribute complete_topology.      -->
<!-- ************************************************** -->

<!ELEMENT Location ( (Location_Container  | Device )* ) >
<!ATTLIST Location
        complete_topology    (false | true)     #IMPLIED>

<!-- ************************************************** -->
<!-- The Location_Container element, which is used for  -->
<!-- Location container, may contain any number of      -->
<!-- Location_Container and Device elements.            -->
<!--                                                    -->
<!-- "model_type" and "name" are the required attributes -->
<!-- to uniquely identify Location_Container models.    -->
<!--                                                    -->
<!-- "model_handle" can also be used to identify models. -->
<!-- If "model_handle" is provided, the values of "name" -->
<!-- and "model_type" will be ignored.                  -->
<!-- ************************************************** -->

<!ELEMENT Location_Container ( ( Location_Container | Device )* )>

<!ATTLIST Location_Container
        name        CDATA                   #REQUIRED
        model_type        ( Country    |
                        Region     |
                        Site       |
                        Building   |
                        Floor      |
                        Section    |
                        Room       )     #REQUIRED

        model_handle        CDATA           #IMPLIED
        Security_String     CDATA           #IMPLIED
        model_name          CDATA           #IMPLIED
        model_modify_author CDATA           #IMPLIED
        complete_topology   (false | true)  #IMPLIED >

<!-- ********************************************************** -->
<!-- Device element is used for device model.                   -->
<!--                                                            -->
```

```
<!-- The "ip_dnsname" is the required attribute to uniquely       -->
<!-- identify device models.                                      -->
<!--                                                              -->
<!-- model_handle" can also be used to identify device models.    -->
<!-- If "model_handle" is provided, the value of "ip_dnsname"     -->
<!-- will be ignored.                                             -->
<!--                                                              -->
<!-- CAUTION:                                                     -->
<!--                                                              -->
<!-- 1. If the attribute is_managed is set to false, the device   -->
<!--    is not contactable.  You must therefore set the model_type-->
<!--    attribute.                                                -->
<!--                                                              -->
<!-- 2. A Device can contain 0, 1 or more than 1 Port in the      -->
<!--    following situations respectively:                        -->
<!--                                                              -->
<!-- (a) If a Device is in a Container or a Destroy element,      -->
<!--     the Port element is not needed.  If there are Port       -->
<!--     elements provided, they will be ignored.                 -->
<!--                                                              -->
<!-- (b) If a Device is in a Connection element, only one Port    -->
<!--     element is allowed.                                      -->
<!--                                                              -->
<!-- (c) If a Device is contained in an Update element to update  -->
<!--     ports, than one Port elements are allowed.               -->
<!--                                                              -->
<!-- (d) If you specify discover_connections="true" on a device   -->
<!--     tag, do not also specify it on that device's parent      -->
<!--     container.  This has performance and efficiency related  -->
<!--     issues as specifying that attribute on a container       -->
<!--     causes Spectrum to discover connections on each model    -->
<!--     in that container anyway.                                -->
<!--                                                              -->
<!-- ************************************************************ -->

<!ELEMENT Device ( Port* | Schedule ) >

<!ATTLIST Device
        ip_dnsname        CDATA             #REQUIRED
        secdomain_ipname  CDATA             #IMPLIED

        model_handle      CDATA             #IMPLIED
        model_type        CDATA             #IMPLIED
        community_string  CDATA             #IMPLIED
       agent_port        CDATA             #IMPLIED
        poll_interval     CDATA             #IMPLIED
        log_ratio         CDATA             #IMPLIED
        model_name        CDATA             #IMPLIED
        DeviceType        CDATA             #IMPLIED
```

```
        x_coordinate      CDATA              #IMPLIED
        y_coordinate      CDATA              #IMPLIED
         is_managed        (true  | false)    #IMPLIED
         reconfig          (true  | false)    #IMPLIED
         poll_status       (true  | false)    #IMPLIED
         discover_connections (false | true)  #IMPLIED >

<!-- ********************************************************** -->
<!-- Port element is used for device port model.                -->
<!--                                                            -->
<!-- "identifier_name" and "identifier_value" are the required  -->
<!-- attributes to uniquely identify port models.               -->
<!--                                                            -->
<!-- "model_handle" can also be used to identify port models.   -->
<!-- If "model_handle" is provided, the value of identifier_name -->
<!-- and identifier_value will be ignored.                      -->
<!-- ********************************************************** -->

<!ELEMENT Port ( Port* ) >

<!ATTLIST Port
        identifier_name ( portDescription        |
                          model_name             |
                          ifIndex                |
                          ipAddress              |
                          ifPhysAddress          |
                          ifName                 |
                          ifAlias                |
                          portID                 |
                          frCircuitTableInstance |
                          atmVclTableInstance    |
                          atmVplTableInstance    ) #REQUIRED
        identifier_value  CDATA                    #REQUIRED

        model_handle      CDATA                    #IMPLIED
        ip_dnsname        CDATA                    #IMPLIED
        model_type        CDATA                    #IMPLIED
        model_name        CDATA                    #IMPLIED
        circuit_id        CDATA                    #IMPLIED
        circuit_name      CDATA                    #IMPLIED
        log_ratio         CDATA                    #IMPLIED
        poll_interval     CDATA                    #IMPLIED
        poll_status       (false | true)           #IMPLIED >
```

```
<!-- ********************************************************************** -->
<!-- Represents a Schedule model                                            -->
<!--                                                                        -->
<!-- SCHED_Recurrence can have the following values                         -->
<!--                                                                        -->
<!--  1 = Always (24 x 7)                                                   -->
<!--  2 = Daily                                                             -->
<!--  3 = Weekly                                                            -->
<!--  4 = Monthly                                                           -->
<!--  5 = Yearly                                                            -->
<!--  6 = Once                                                              -->
<!--                                                                        -->
<!--  SCHED_Start_Hour: value range 0-23                                    -->
<!--  SCHED_Start_Minute: value range 0-59                                  -->
<!--  SCHED_Start_DoW: Day of the week (range 0-6 where Sunday is 0) for    -->
<!--                   Weekly recurrence                                    -->
<!--  SCHED_Start_DoM: Day of the month (range 1-31) for Monthly and Yearly -->
<!--                   recurrence                                           -->
<!--  SCHED_Start_Month: range 0-11 where January is 0 for Yearly           -->
<!--                     recurrence                                         -->
<!--  SCHED_Duration: active period duration in seconds. May be 0 (default) -->
<!--  SCHED_Recurrence_Multiplier: number of recurrence units  that         -->
<!--                               determine the length of time between the -->
<!--                               start of each active period.             -->
<!--                               Default is 1.                            -->
<!--  SCHED_Daily_Repeat_Limit: number of consecutive days to repeat a      -->
<!--                            daily schedule (specified by                -->
<!--                            SCHED_Start_Hour and SCHED_Start_Minute)    -->
<!--                            at the start of each recurrence period.     -->
<!--                            Only applicable tp Weekly, Monthly or       -->
<!--                            Yearly recurrence.                          -->
<!-- SCHED_DayBitMask: The days of the week on which a WEEKLY Schedule      -->
<!--                   should be ACTIVE.  Values are:                       -->
<!--                   Sunday = 1,                                          -->
<!--                   Monday = 2,                                          -->
<!--                   Tuesday = 4,                                         -->
<!--                   Wednesday = 8,                                       -->
<!--                   Thursday = 16,                                       -->
<!--                   Friday = 32,                                         -->
<!--                   Saturday = 64                                        -->
<!--                   e.g if we want Mon, Wed and Fri the value would be   -->
<!--                   2+8+32=42                                            -->
<!--  SCHED_Start_MoY,                                                      -->
<!--  SCHED_START_YEAR,                                                     -->
```

```
<!--  SCHED_START_DAY: Used in conjunction with SCHED_START_MONTH to       -->
<!--                   indicate that a schedule should be active on some   -->
<!--                   day on the future.  Both SCHED_START_YEAR and       -->
<!--                   SCHED_START_DAY must be non-zero.                   -->
<!--                   Otherwise the schedule behaves normally and goes    -->
<!--                   active as early as today.                           -->
<!--                   Note that SCHED_START_YEAR should be specified as   -->
<!--                   the number of years since 1900.                     -->
<!-- SCHED_Description: Description for this Schedule.                     -->
<!--                                                                       -->
<!-- ********************************************************************* -->

<!ELEMENT Schedule ( #PCDATA) >

<!ATTLIST Schedule
        name                        CDATA               #REQUIRED
        SCHED_Recurrence            ( 1 | 2 | 3 |
                                        4 | 5 | 6 )        #REQUIRED
        SCHED_Daily_Repeat_Limit    CDATA               #REQUIRED
        SCHED_Duration              CDATA               #REQUIRED
        SCHED_Recurrence_Multiplier CDATA               #REQUIRED
        SCHED_Start_DoM             CDATA               #REQUIRED
        SCHED_Start_DoW             CDATA               #REQUIRED
        SCHED_Start_Hour            CDATA               #REQUIRED
        SCHED_Start_Minute          CDATA               #REQUIRED
        SCHED_Start_Month           CDATA               #REQUIRED
        SCHED_Start_Day             CDATA               #REQUIRED
        SCHED_DayBitMask            CDATA               #REQUIRED
        SCHED_Start_Year            CDATA               #REQUIRED
        SCHED_Start_MoY             CDATA               #REQUIRED
        SCHED_Description           CDATA               #REQUIRED
        >
<!-- ************************************************************* -->
<!-- Element used to represent EventModels.                        -->
<!--                                                               -->
<!-- The unique id must be specified for performance reasons.      -->
<!-- ************************************************************* -->

<!ELEMENT EventModel ( #PCDATA ) >

<!ATTLIST EventModel
        model_name        CDATA       #REQUIRED
        unique_id         CDATA       #REQUIRED
        model_handle      CDATA       #IMPLIED
        Security_String   CDATA       "public"
        manager_name      CDATA       "0">
```

```
<!-- ********************************************************** -->
<!-- Connection element represents device connectivity.         -->
<!-- It contains two Device elements involved in the connection.   -->
<!-- Each Device may have 0 or 1 Port element.                  -->
<!--                                                            -->
<!-- Connection has one attribute, create_pipe. Normally for ATM   -->
<!-- circuit links, create_pipe is set to false to avoid the    -->
<!-- creation of numerous pipes within the view. In this case,  -->
<!-- one can use ATM Manager to view the connections. It's up   -->
<!-- to users to decide the setting for create_pipe. By default,   -->
<!-- a pipe will be created for each connection.                -->
<!--                                                            -->
<!-- ********************************************************** -->

<!ELEMENT Connection (Device, Device)>

<!ATTLIST Connection  create_pipe  (true | false)  "true" >

<!-- ********************************************************** -->
<!-- Update element is to update SPECTRUM model attributes and    -->
<!-- associations.                                              -->
<!--                                                            -->
<!-- The Update element is allowed to contain any number of     -->
<!-- Container, Device and Association elements to be updated.    -->
<!-- To update a port, the Port element needs to placed inside a  -->
<!-- Device element and then place the Device element into the    -->
<!-- Update element.                                            -->
<!-- ********************************************************** -->
```

```
<!ELEMENT Update  ( ( Topology_Container           |
                     Location_Container           |
                     GenericView_Container        |
                     Connection                   |
                     Device                       |
                     EventModel                   |
                     SM_Service                   |
                     SM_AttrMonitor               |
                     SM_LatencyMon                |
                     SM_ConnectMon                |
                     SM_SLA                       |
                     SM_Guarantee                 |
                     SM_Customer                  |
                     Association
                   )* ) >

<!-- ********************************************************** -->
<!-- Destroy elemen: destroy SPECTRUM models and associations.    -->
<!--                                                            -->
<!-- The Destroy element is allowed to contain any number of      -->
<!-- Container, Device and Connection and Association elements    -->
<!-- to be destroyed.                                           -->
<!-- ********************************************************** -->

<!ELEMENT Destroy ( ( Topology_Container           |
                     Location_Container           |
                     GenericView_Container        |
                     Device                       |
                     Connection                   |
                     EventModel                   |
                     SM_Service                   |
                     SM_AttrMonitor               |
                     SM_LatencyMon                |
                     SM_ConnectMon                |
                     SM_SLA                       |
                     SM_Guarantee                 |
                     SM_Customer                  |
                     Association

                   )* ) >
```

```
<!-- ********************************************************** -->
<!-- Association element defines SPECTRUM association between two -->
<!-- models for creation or destroy.                            -->
<!--                                                            -->
<!-- Association element contains one Left_Model and one        -->
<!-- Right_Model element.                                       -->
<!-- ********************************************************** -->

<!ELEMENT Association ((Left_Model | Right_Model)*) >
<!ATTLIST Association relation CDATA #REQUIRED >


<!-- ********************************************************** -->
<!-- Left_Model element defines left side model in a Spectrum   -->
<!-- association.                                               -->
<!--                                                            -->
<!-- Left_Model element is only allowed to contain one child    -->
<!-- element.                                                   -->
<!-- ********************************************************** -->

<!ELEMENT Left_Model (Device                        |
                      Port                          |
                      Topology_Container            |
                      Location_Container            |
                      EventModel                    |
                      Model
                     ) >



<!-- ********************************************************** -->
<!-- Right_Model element defines right side model in a Spectrum -->
<!-- association.                                               -->
<!--                                                            -->
<!-- Right_Model element is only allowed to contain one child   -->
<!-- element.                                                   -->
<!-- ********************************************************** -->
<!ELEMENT Right_Model (Device                       |
                       Port                         |
                       Topology_Container           |
                       Location_Container           |
                       EventModel                   |
                       Model
                      ) >


<!-- ********************************************************** -->
<!-- Model element can be used to represent any SPECTRUM models. -->
<!--                                                            -->
<!-- model_type and name have to be provided to define a model. -->
<!-- "model_type" and "name" should be used to uniquely identify -->
```

```
<!-- models.   However, if "model_handle" is provided, the values -->
<!-- of "model_type" and "name" will not be used.                 -->
<!-- ************************************************************ -->

<!ELEMENT Model ( #PCDATA ) >
<!ATTLIST Model
          name           CDATA    #REQUIRED
          model_type     CDATA    #REQUIRED
          model_handle   CDATA    #IMPLIED >


<!-- ************************************************************ -->
<!-- GenericView element, which is used for customized view,      -->
<!-- may contain any number of GenericView_Container and Device   -->
<!-- elements.                                                    -->
<!--                                                              -->
<!-- This element has 3 required attributes containment_relation, -->
<!-- model_type and name.                                         -->
<!-- ************************************************************ -->

<!ELEMENT GenericView ((GenericView_Container | Device )*) >

<!ATTLIST GenericView
        containment_relation  CDATA          #REQUIRED
        model_type            CDATA          #REQUIRED
        name                  CDATA          #REQUIRED
        complete_topology     (false | true) #IMPLIED >

<!-- ************************************************************ -->
<!-- GenericView_Container element, which is used for the        -->
<!-- GenericView container, may contain any number of            -->
<!-- GenericView_Container, and Device elements.                 -->
<!--                                                             -->
<!-- This element requires model_type and name attribute.        -->
<!-- Attribute containment_relation is not required.  If it's not-->
<!-- specified, the parent's containment_relation will be        -->
<!-- inheritted.  The Model_Attr can be used for multi-lines     -->
<!-- text string SPECTRUM attrs, or list attributes.             -->
<!-- ************************************************************ -->

<!ELEMENT GenericView_Container ( GenericView_Container |
        Device
        )*>
<!ATTLIST GenericView_Container
        name                CDATA          #REQUIRED
        model_type          CDATA          #REQUIRED
        containment_relation CDATA         #IMPLIED >
```

```
<!-- ********************************************************** -->
<!-- Model_Attr is used for multi-lines text string or list      -->
<!-- SPECTRUM attributes.                                        -->
<!--                                                             -->
<!-- attr_id is required for this element to specify the         -->
<!-- SPECTRUM attribute.  This element can contain multi-lines   -->
<!-- of text strings for SPECTRUM Text String attr type, or      -->
<!-- multiple List_Value elements for a SPECTRUM list attribute. -->
<!-- ********************************************************** -->

<!ELEMENT Model_Attr ( #PCDATA | List_Value )* >
<!ATTLIST Model_Attr
                        attr_id   CDATA          #REQUIRED >

<!-- ********************************************************** -->
<!-- List_Value is used for SPECTRUM list attribute values.      -->
<!--                                                             -->
<!-- Each List_Value conatins a PCDATA and serves for one        -->
<!-- instance value for the list attribute.                      -->
<!-- ********************************************************** -->
<!ELEMENT List_Value ( #PCDATA ) >

<!-- ********************************************************** -->
<!-- Service Level Management topology elements.                -->
<!-- ********************************************************** -->

<!ELEMENT CustomerManager ( SM_Customer |
                                SM_CustomerGroup
                                )*>

<!ATTLIST CustomerManager
        name                  CDATA                #IMPLIED
        containment_relation  ( Groups_Customers )  #IMPLIED
        model_type            ( CustomerManager )   #IMPLIED
        >

<!ELEMENT SM_ServiceMgr ( SM_Service )*>

<!ATTLIST SM_ServiceMgr
        name            CDATA                          #IMPLIED
        containment_relation          ( SlmContains )   #IMPLIED
        model_type                    ( SM_ServiceMgr ) #IMPLIED
        >

<!ELEMENT SM_SLA_Mgr ( SM_SLA )*>
```

```
<!ATTLIST SM_SLA_Mgr
        name            CDATA                       #IMPLIED
        containment_relation  ( SlmContainsSLAs )   #IMPLIED
        model_type            ( SM_SLA_Mgr )        #IMPLIED
        >

<!ELEMENT SM_Service_Mgt ( CustomerManager |
                        SM_ServiceMgr   |
                        SM_SLA_Mgr
                        )*>

<!ATTLIST SM_Service_Mgt
        name            CDATA                           #IMPLIED
        containment_relation  ( SlmHasServiceComponent )   #IMPLIED
        model_type            ( SM_Service_Mgt )           #IMPLIED
        >

<!-- The Correlation element represents the root model Correlation_Manager -->
<!-- (0x10469).  It can only contain Correlation_Domain elements, and has  -->
<!-- no attributes.  All Correlation_Domains will be related to the        -->
<!-- Correlation Manager by the CORRELATES relation.                       -->

<!-- Since Correlation_Manager is a unique model, this element does not    -->
<!-- actually cause the SpectroSERVER to create a model.  It represents    -->
<!-- a pre-existing model.                                                 -->

<!ELEMENT Correlation ( Correlation_Domain )*>

<!-- The Correlation_Domain can contain any number of Devices, Ports,      -->
<!-- Model_Attrs, or GenericView_Containers.  They will all be related     -->
<!-- to the Correlation_Domain by the CORRELATES relation.  Correlation_    -->
<!-- Domains also have no attributes.                                      -->

<!ELEMENT Correlation_Domain (  Device               |
                                Port                |
                                Model_Attr          |
                                GenericView_Container
                                )*>

<!ATTLIST Correlation_Domain
        name            CDATA                   #REQUIRED
        >

<!ELEMENT RTM_Test ( #PCDATA) >

<!ATTLIST RTM_Test
        name            CDATA             #REQUIRED
        model_type      ( RTM_Test )      #IMPLIED
        >
```

```
<!ELEMENT SM_Service ( SM_Service         |
                       SM_AttrMonitor     |
                       SM_LatencyMon      |
                       SM_ConnectMon      |
                       Device             |
                       Port               |
                       Topology_Container |
                       RTM_Test           |
                       MonitorPolicy_Attr |
                       Schedule
                       )*>

<!-- ********************************************************************** -->
<!-- Represents a SM_Service model                                          -->
<!--                                                                        -->
<!-- Criticality can be one of the following                                -->
<!--                                                                        -->
<!--  10 - Low                                                              -->
<!--  15 - Medium Low                                                       -->
<!--  20 - Medium                                                           -->
<!--  25 - Medium High                                                      -->
<!--  10 - High                                                             -->
<!--                                                                        -->
<!-- AttrToWatch can be one of the following                                -->
<!--                                                                        -->
<!--  Condition - can be used for most models, polices 1-5                  -->
<!--  Contact_Status - typically used for device models, policies 10-13     -->
<!--  Port_Status - used for interface models, policies 14-17               -->
<!--  LatestErrorStatus - used for RTM_Test models, policies 18-21          -->
<!--  or Response_Time                                                      -->
<!--  RM_Condition - SM_AttrMonitor or SM_Service models, policies 6-9      -->
<!--  or Service_Health                                                     -->
<!--                                                                        -->
<!-- MonitorPolicy_ID - 1-21 Index of SLM_DefaultPolicies of GlobalConfig   -->
<!--                                                                        -->
<!--  1  - Condition Rollup                                                 -->
<!--  2  - Condition Redundancy                                             -->
<!--  3  - Condition High Sensitivity                                       -->
<!--  4  - Condition Low Sensitivity                                        -->
<!--  5  - Condition Percentage                                             -->
<!--  6  - Service Health Redundancy                                        -->
<!--  7  - Service Health High Sensitivity                                  -->
<!--  8  - Service Health Low Sensitivity                                   -->
<!--  9  - Service Health Percentage                                        -->
<!--  10 - Contact Status Redundancy                                        -->
<!--  11 - Contact Status High Sensitivity                                  -->
<!--  12 - Contact Status Low Sensitivity                                   -->
```

```
<!--  13 - Contact Status Percentage                                    -->
<!--  14 - Port Status Redundancy                                       -->
<!--  15 - Port Status High Sensitivity                                 -->
<!--  16 - Port Status Low Sensitivity                                  -->
<!--  17 - Port Status Percentage                                       -->
<!--  18 - Response Time Redundancy                                     -->
<!--  19 - Response Time High Sensitivity                               -->
<!--  20 - Response Time Low Sensitivity                                -->
<!--  21 - Response Time Low Percentage                                 -->
<!--                                                                    -->
<!-- ****************************************************************** -->

<!ATTLIST SM_Service
        name              CDATA                          #REQUIRED
        containment_relation  ( SlmMonitors |
                                SlmWatchesContainer |
                                MaintenenceScheduledBy ) #IMPLIED
        Criticality  ( 10 | 15 | 20 | 25 | 30 )          #IMPLIED

        AttrToWatch             CDATA                    #IMPLIED
        MonitorPolicy_ID        CDATA                    #IMPLIED
        is_managed              (true  | false)          #IMPLIED
        Generate_Service_Alarms (true  | false)          #IMPLIED
        Security_String         CDATA                    #IMPLIED
        model_type        ( SM_Service )                 #IMPLIED
        >

<!ELEMENT SM_AttrMonitor ( SM_Service         |
                        SM_AttrMonitor     |
                        SM_LatencyMon      |
                        SM_ConnectMon      |
                        Device             |
                        Port               |
                        Topology_Container |
                        RTM_Test           |
                        MonitorPolicy_Attr |
                        )*>


<!-- ****************************************************************** -->
<!-- Represents a SM_AttrMonitor model                                  -->
<!--                                                                    -->
<!-- AttrToWatch can be one of the following                            -->
<!--                                                                    -->
<!--  Condition - can be used for most models, polices 1-5              -->
<!--  Contact_Status - typically used for device models, policies 10-13 -->
<!--  Port_Status - used for interface models, policies 14-17           -->
<!--  LatestErrorStatus - used for RTM_Test models, policies 18-21      -->
```

```
<!--  or Response_Time                                                    -->
<!--  RM_Condition - SM_AttrMonitor or SM_Service models, policies 6-9    -->
<!--  or Service_Health                                                   -->
<!--                                                                      -->
<!-- MonitorPolicy_ID - 1-21 Index of SLM_DefaultPolicies of GlobalConfig -->
<!--                                                                      -->
<!--  1  - Condition Rollup                                               -->
<!--  2  - Condition Redundancy                                           -->
<!--  3  - Condition High Sensitivity                                     -->
<!--  4  - Condition Low Sensitivity                                      -->
<!--  5  - Condition Percentage                                           -->
<!--  6  - Service Health Redundancy                                      -->
<!--  7  - Service Health High Sensitivity                                -->
<!--  8  - Service Health Low Sensitivity                                 -->
<!--  9  - Service Health Percentage
<!--  10 - Contact Status Redundancy                                      -->
<!--  11 - Contact Status High Sensitivity                                -->
<!--  12 - Contact Status Low Sensitivity                                 -->
<!--  13 - Contact Status Percentage                                      -->
<!--  14 - Port Status Redundancy                                         -->
<!--  15 - Port Status High Sensitivity                                   -->
<!--  16 - Port Status Low Sensitivity                                    -->
<!--  17 - Port Status Percentage                                         -->
<!--  18 - Response Time Redundancy                                       -->
<!--  19 - Response Time High Sensitivity                                 -->
<!--  20 - Response Time Low Sensitivity                                  -->
<!--  21 - Response Time Low Percentage                                   -->
<!--                                                                      -->
<!--  Special_Cause_List - List or range of alarm causes which can be     -->
<!--                       used to specify included or excluded from      -->
<!--                       impacting the service health of the Service    -->
<!--                       or Resource Monitor model.  Available only     -->
<!--                       when AttrToWatch is Condition.                 -->
<!--                                                                      -->
<!--  Cause_List_Control - Specifies how Special_Cause_List is used.      -->
<!--                       0 - Unused                                     -->
<!--                       1 - Inclusive                                  -->
<!--                       2 - Exclusive                                  -->
<!--                                                                      -->
<!-- ******************************************************************** -->
```

```
<!ATTLIST SM_AttrMonitor
        name                    CDATA                       #REQUIRED
        containment_relation    ( SlmMonitors |
                                  SlmWatchesContainer )  #IMPLIED
        is_managed              (true  | false)           #IMPLIED
        AttrToWatch             CDATA                       #IMPLIED
        MonitorPolicy_ID        CDATA                       #IMPLIED
        Generate_Service_Alarms (true  | false)           #IMPLIED
        model_type      ( SM_AttrMonitor )              #IMPLIED
        >

<!ELEMENT MonitorPolicy_Attr ( #PCDATA ) >

<!ELEMENT SM_SLA ( SM_Service   |
                SM_Guarantee |
                Schedule
                )*>

<!-- ********************************************************************** -->
<!-- Represents a SM_Guarantee model                                        -->
<!--                                                                        -->
<!-- SLAControl can have the following values                               -->
<!--                                                                        -->
<!--  0 = Inactive                                                          -->
<!--  1 = Active                                                            -->
<!--                                                                        -->
<!-- ********************************************************************** -->

<!ATTLIST SM_SLA
        name              CDATA                   #REQUIRED
        containment_relation  ( SlmHasGuarantee |
                                  SlmGuarantees |
                                  SlaPeriod )       #IMPLIED
        is_managed          (true  | false)       #IMPLIED
        SLA_Control         ( 0 | 1 )             #IMPLIED
        SLA_ExpirationDate    CDATA               #IMPLIED
        SLA_Notes             CDATA               #IMPLIED
        SLA_Description       CDATA               #IMPLIED
        Security_String       CDATA               #IMPLIED
        model_type          ( SM_SLA )            #IMPLIED
        >

<!ELEMENT SM_Guarantee ( SM_Service      |
                        SM_AttrMonitor  |
                        SM_LatencyMon   |
                        SM_ConnectMon   |
                        Schedule
                        )*>
```

```
<!-- ********************************************************************** -->
<!-- Represents a SM_Guarantee model                                        -->
<!--                                                                        -->
<!-- GuaranteeControl can have the following values                         -->
<!--                                                                        -->
<!--  0 = Inactive                                                          -->
<!--  1 = Active                                                            -->
<!--                                                                        -->
<!-- GuranteeType can have the following values                             -->
<!--                                                                        -->
<!--  0 = Availability                                                      -->
<!--  1 = Performance                                                       -->
<!--  2 = Mean Time To Repair                                               -->
<!--  3 = Maximum Outage Time                                               -->
<!--                                                                        -->
<!-- ServiceHealthType can have the following values                        -->
<!--                                                                        -->
<!--  1 - Down                                                              -->
<!--  2 - Degraded                                                          -->
<!--                                                                        -->
<!-- ********************************************************************** -->

<!ATTLIST SM_Guarantee
        name                        CDATA          #REQUIRED
        containment_relation  ( SlmIsMeasuredBy |
                                SlmSchedulesGuarantee )  #IMPLIED
        is_managed            (true  | false)      #IMPLIED
        GuaranteeControl      ( 0 | 1 )            #IMPLIED
        GuaranteeType         ( 0 | 1 | 2 | 3 )    #REQUIRED
        ServiceHealthType     ( 1 | 2 )            #IMPLIED
        WarningThreshold            CDATA          #IMPLIED
        WarningThresholdPercent     CDATA          #IMPLIED
        ViolationThreshold          CDATA          #IMPLIED
        ViolationThresholdPercent   CDATA          #IMPLIED
        GuaranteeNotes              CDATA          #IMPLIED
        GuaranteeDescription        CDATA          #IMPLIED
        model_type             ( SM_Guarantee )    #IMPLIED
        MOT_Threshold               CDATA          #IMPLIED
        MTTR_Threshold              CDATA          #IMPLIED
        MTBF_Threshold              CDATA          #IMPLIED
        >

<!ELEMENT SM_LatencyMon ( Topology_Container |
                          MonitorPolicy_Attr )*>
```

```
<!ATTLIST SM_LatencyMon
        name                    CDATA                #REQUIRED
        containment_relation  ( SlmMonitors |
                                   SlmWatchesContainer ) #IMPLIED
        is_managed            ( true  | false )      #IMPLIED
        DefaultMaxRTT           CDATA                #IMPLIED
        DefaultMeasureInterval  CDATA                #IMPLIED
        model_type            ( SM_LatencyMon )      #IMPLIED
        >

<!ELEMENT SM_CustomerGroup ( SM_CustomerGroup |
                        SM_Customer
                        )*>

<!ATTLIST SM_CustomerGroup
        name                           CDATA          #REQUIRED
        containment_relation ( Groups_Customers )     #IMPLIED
        model_type           ( SM_CustomerGroup )     #IMPLIED
        >

<!ELEMENT SM_Customer ( SM_Service |
                           SM_SLA     |
                         )*>

<!-- ********************************************************************** -->
<!-- Represents a SM_Customer model                                         -->
<!--                                                                        -->
<!-- Criticality can be one of the following                                -->
<!--                                                                        -->
<!--  10 - Low                                                              -->
<!--  15 - Medium Low                                                       -->
<!--  20 - Medium                                                           -->
<!--  25 - Medium High                                                      -->
<!--  10 - High                                                             -->
<!--                                                                        -->
<!-- ********************************************************************** -->

<!ATTLIST SM_Customer
        name                           CDATA  #REQUIRED
        containment_relation  ( SlmAgreesTo |
                                  SlmUses )     #IMPLIED
        Security_String                CDATA  #IMPLIED
        CustomerID                     CDATA  #IMPLIED
        Criticality                    CDATA  #IMPLIED
        CustomerField4                 CDATA  #IMPLIED
        CustomerField5                 CDATA  #IMPLIED
        CustomerField6                 CDATA  #IMPLIED
        CustomerField7                 CDATA  #IMPLIED
        Contact_Name                   CDATA  #IMPLIED
```

```
        Contact_Title                 CDATA  #IMPLIED
        Contact_Location              CDATA  #IMPLIED
        Email_Address                 CDATA  #IMPLIED
        Phone_Number                  CDATA  #IMPLIED
        Mobile_Phone_Number           CDATA  #IMPLIED
        Pager_Number                  CDATA  #IMPLIED
        Fax_Number                    CDATA  #IMPLIED
        User_Defined_1                CDATA  #IMPLIED
        User_Defined_2                CDATA  #IMPLIED
        User_Defined_3                CDATA  #IMPLIED
        User_Defined_4                CDATA  #IMPLIED
        Secondary_Contact_Name        CDATA  #IMPLIED
        Secondary_Contact_Title       CDATA  #IMPLIED
        Secondary_Contact_Location    CDATA  #IMPLIED
        Secondary_Email_Address       CDATA  #IMPLIED
        Secondary_Phone_Number        CDATA  #IMPLIED
        Secondary_Mobile_Phone_Number CDATA  #IMPLIED
        Secondary_Pager_Number        CDATA  #IMPLIED
        Secondary_Fax_Number          CDATA  #IMPLIED
        Secondary_User_Defined_1      CDATA  #IMPLIED
        Secondary_User_Defined_2      CDATA  #IMPLIED
        Secondary_User_Defined_3      CDATA  #IMPLIED
        Secondary_User_Defined_4      CDATA  #IMPLIED
        model_type           ( SM_Customer )  #IMPLIED
        >

<!-- ********************************************************************** -->
<!-- Represents a GlobalCollection model                                    -->
<!--                                                                        -->
<!-- ********************************************************************** -->
<!ELEMENT GlobalCollection (( Device            |
                      Topology_Container |
                      Location_Container )*) >


<!ATTLIST GlobalCollection
        name                 CDATA              #REQUIRED
        containment_relation  CDATA             "GlobalCollect"
        collectionDescription CDATA             #IMPLIED
        Security_String       CDATA             #IMPLIED >
```

Appendix C: XML Examples

This section contains XML examples to help you work with the DTD.

Note: The element names in each example are highlighted in bold. Making the names bold makes the examples easier to read; it is not intended to imply formatting necessary for the XML input file.

Example 1: Importing into the Topology View

This example shows a basic input file that imports information into the CA Spectrum Topology view. This file creates a Network container model in the Topology view. In the Network container, a LAN container model is created. Within the LAN container two devices are created. The DNS name deadlock identifies one device, and the IP address identifies the other device.

The complete_topology attribute of the Topology element is set to False. In this case, CA Spectrum respects other models that previously exist in the Topology view. Consequently, this file only looks to create models for entries that are listed in the XML file that are not already modeled. The models that are created are placed in the Topology hierarchy as specified in the file. Models that previously exist in the Topology hierarchy are not rediscovered but are moved to the container specified in the file.

Note: When complete_topology is set to false, existing models that are in the container but not listed in the import file are *not* sent to the Lost and Found. These models *are* sent to Lost and Found when complete_topology is set to true.

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE Import SYSTEM ".modelinggateway.dtd">
<Import>
<!-- ********************************** -->
<!-- This part is for Topology view import -->
<!-- ********************************** -->
<Topology complete_topology="false">
        <Device ip_dnsname="10.253.9.17" model_type="GnSNMPDev"
                community_string="public"/>
        <Device ip_dnsname="nmcss52-5" />
        <Topology_Container model_type="Network" name="My Network"
                Security_String="public" subnet_address="10.253.0.0"
                subnet_mask="255.255.0.0">
```

```
                <Topology_Container model_type="Lan" name="Lan1"
                        Security_String="public" subnet_address="10.253.9.0"
                        subnet_mask="255.255.255.0">
                        <Device ip_dnsname="deadlock" />
                        <Device ip_dnsname="10.253.9.18" poll_interval="333" />
                </Topology_Container>
        </Topology_Container>
</Topology>
</Import>
```

Example 2: Creating Connections

The following example shows the creation of a connection between two ATM circuits and the creation of a connection between two Frame Relay circuits. The example also shows a connection between a FrameRelay DLCI port and an ATM VCL port.

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE Import SYSTEM ".modelinggateway.dtd">
<Import>
<Topology complete_topology="false">
        <Connection create_pipe="false">
                <Device ip_dnsname="10.253.32.225">
                        <Port identifier_name="atmVclTableInstance"
                                identifier_value="5.0.5"
                                circuit_name="ATM Link1"
                                circuit_id = "ATM 5017" />
                </Device>
                <Device ip_dnsname="192.168.52.25">
                        <Port identifier_name="atmVclTableInstance"
                                identifier_value="3.0.12"
                                circuit_name="ATM Link1"
                                circuit_id = "ATM 5017" />
                </Device>
        </Connection>
        <Connection>
                <Device ip_dnsname="10.253.9.18">
                        <Port identifier_name="frCircuitTableInstance"
                                identifier_value="2.27"/>
                </Device>
                <Device ip_dnsname="nmcss52-5">
                        <Port identifier_name="frCircuitTableInstance
                                identifier_value="4.161"/>
                </Device>
        </Connection>
```

```
        <!-- ****************************************** -->
        <!-- Connection between a FrameRelay DLCI port and -->
        <!-- an ATM VCL port.                           -->
        <!-- ****************************************** -->
        <Connection>
                <Device ip_dnsname="10.253.9.18">
                        <Port identifier_name="frCircuitTableInstance"
                                identifier_value="2.27"/>
                </Device>
                <Device ip_dnsname="10.253.32.225">
                        <Port identifier_name="atmVclTableInstance"
                                identifier_value="5.0.17"/>
                </Device>
        </Connection>
        <Connection>
                <Device ip_dnsname="nmcss52-5">
                        <Port identifier_name="ifIndex" identifier_value="3"/>
                </Device>
                <Device ip_dnsname="10.253.9.17">
                        <Port identifier_name="ifPhysAddress"
                                identifier_value="0:4:27:C:91:C0"/>
                </Device>
        </Connection>
<Topology>
</Import>
```

Example 3: Updating and Destroying

This example illustrates the use of the Update and the Destroy elements.

The Update element contains a Location_Container element. This example updates the model name by using the name and model_name attributes of the Location_Container element. The name attribute is set equal to the current name and identifies the model to be updated. The model_name attribute updates the value of the name attribute to Peace2.

The Update element also contains a Device element and Port element. The attributes identifer_name and identifier_value are used to identify the port to be updated. The other attributes that are specified are those attributes whose values are updated. The port model name is changed to port 2 and the poll_status is changed to False.

The Destroy element eliminates the device model deadlock. Any connections or ports that are associated with deadlock are automatically destroyed. The Building container model Durham is also eliminated. Any models that are contained within the Durham building container are sent to the Lost and Found.

The Destroy element also removes the connection between the specified port on device nmcss52-5 and the specified port on nmcss52-3.

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE Import SYSTEM ".modelinggateway.dtd">

<Import>

<!-- ********************************************** -->
<!-- Model update.....................................-->
<!-- ********************************************** -->
        <Update>
<!-- ********************************************** -->
<!-- Change container Peace's model name from Peace to-->
<!-- Peace2 ................................. ....-->
<!-- ********************************************** -->
                <Location_Container model_type="Building" name="Peace"
                        model_name="Peace2"/>
<!-- ****************************************** -->
<!-- Update port ifIndex=2 on device nmcss52-5  -->
<!-- ****************************************** -->
                <Device ip_dnsname="nmcss52-5">
                        <Port identifier_name="ifIndex" identifier_value="2"
                                model_name="port 2" poll_status="false" />
                </Device>
        </Update>
  <!-- ****************************** -->
  <!-- Destroy models and connections. -->
  <!-- ****************************** -->
        <Destroy>
                <Device ip_dnsname="deadlock"/>
                <Location_Container  model_type="Building" name="Durham" />
                <Connection>
                        <Device ip_dnsname="nmcss52-5">
                                <Port identifier_name="ifIndex"
                                        identifier_value="1"/>
                        </Device>
                        <Device ip_dnsname="10.253.9.17">
                                <Port identifier_name="ipAddress"
                                        identifier_value="10.253.8.18"/>
                        </Device>
                </Connection>
        </Destroy>
</Import>
```

Example 4: Creating, Updating, and Destroying

The following XML file illustrates most of the functionality of the elements that are contained within the DTD. This file creates data in both the Topology and Location views, creates connections, updates attributes, and destroys models and connections.

The first section of the XML file creates models and connections between these models in the Topology view. The section begins with the Topology element, <Topology...>. The container and device models are created first and then connections are established. This section ends when the Topology element closes, </Topology>.

After the Topology element is closed, there is a section where another connection is created. This section illustrates that you can create connections without nesting the Connection element within the Topology element.

The next section of the file begins with the Location element, <Location...>. This section demonstrates creating a container and device models in the Location view. This section ends when the Location element closes, </Location>.

The next section begins with the Update element, <Update>. In this section, attribute values of a container, a device, and a port are modified. You cannot know the current attribute values of each of the elements that are represented by looking at the XML file. Therefore, it is not easy to discern which elements are being updated. In general, each element contains an attribute that uniquely identifies the model or port to be updated. The rest of the attributes are specified to update their values. For example, the first element is the Location_Container element. The name attribute uniquely identifies the model. The model_type attribute is specified to update its value, perhaps from Region to Building. The only hierarchy that can be defined in the Update element is the Device/Port hierarchy that specifies the port you would like to update. This section ends when the Update element closes, </Update>.

The last section of this file uses the Destroy element. The Destroy element eliminates:

- The device at 10.253.9.19
- The container model Durham
- The connection between the specified ports on device nmcss52-5 and the device at 10.253.9.17

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE Import SYSTEM ".modelinggateway.dtd">
<Import>
```

```
<!-- ********************************** -->
<!-- This part is for Topology view import -->
<!-- ********************************** -->
        <Topology discover_connections="false" complete_topology="false">
                <Device ip_dnsname="10.253.9.109" model_type="GnSNMPDev"
                        community_string="public" is_managed="false"/>
                <Device ip_dnsname="10.253.9.17"
                        poll_interval="333" log_ratio="11"/>
                <Device ip_dnsname="10.253.9.19" community_string="public"/>
                <Device ip_dnsname="nmcss52-5" />
                <Topology_Container model_type="Network" name="My Network"
                        Security_String="public" subnet_address="10.253.0.0"
                        subnet_mask="255.255.0.0" complete_topology="true">
                        <Topology_Container model_type="Lan"

                                name="MyLan" Security_String="public"

                                subnet_address="10.253.9.0"
                                subnet_mask="255.255.255.0">
                                <Device ip_dnsname="10.253.9.18"
                                        community_string="public"
                                        poll_interval="333"
                                        log_ratio="5"/>
                        </Topology_Container>
                </Topology_Container>
                <Topology_Container model_type="IPClassC" name="my_net"
                        subnet_address="172.19.57.0">
                        <Device  model_type="Pingable"
                                ip_dnsname="172.19.57.91"/>
                        <Device  model_type="Fanout" ip_dnsname="1.2.3.4"/>
                        <Device  ip_dnsname="10.253.9.16"
                                community_string="public"/>
                </Topology_Container>
                <Topology_Container model_type="Lan" name="lan2"
                        Security_String="public" subnet_address="10.253.7.0"
                        subnet_mask="255.255.255.0" complete_topology="true">
                        <Device ip_dnsname="10.253.7.17"
                                community_string="public"
                                poll_interval="333" log_ratio="11"/>
                        <Device ip_dnsname="10.253.32.101"/>
                        <Device ip_dnsname="192.168.125.161"
                                model_type="GnSNMPDev"/>
                </Topology_Container>
```

```
        <Device ip_dnsname="172.19.57.92" />
        <Device ip_dnsname="172.19.57.93" />
        <Device ip_dnsname="10.253.32.225" model_type="M46_04"/>
        <Connection>
                <Device ip_dnsname="172.19.57.93">
                        <Port identifier_name= "frCircuitTableInstance"
                                identifier_value="4.161"/>
                </Device>
                <Device ip_dnsname="192.168.125.161">
                        <Port identifier_name ="frCircuitTableInstance"
                                identifier_value="2.161"/>
                </Device>
        </Connection>
        <Connection create_pipe="false">
                <Device ip_dnsname="10.253.32.101">
                        <Port identifier_name ="atmVclTableInstance"
                                identifier_value="3.1.52"/>
                </Device>
                <Device ip_dnsname="10.253.32.225">
                        <Port identifier_name ="atmVclTableInstance"
                                identifier_value="5.0.68"
                                circuit_name="ATM 68"
                                circuit_id ="ATM ID 68"/>
                </Device>
        </Connection>
        <Connection>
                <Device ip_dnsname="nmcss52-5">
                        <Port identifier_name="ifIndex"
                                identifier_value="1"/>
                </Device>
                <Device ip_dnsname="10.253.9.17">
                        <Port identifier_name="ipAddress"
                                identifier_value="10.253.8.18"/>
                </Device>
        </Connection>
</Topology>
<Connection>
        <Device ip_dnsname="172.19.57.92">
                <Port identifier_name="ifPhysAddress"
                        identifier_value="0:E0:63:7C:19:61"/>
        </Device>
        <Device ip_dnsname="10.253.9.17">
                <Port identifier_name="ipAddress"
                        identifier_value="10.253.8.65"/>
        </Device>
</Connection>
```

```
<!-- ********************************** -->
<!-- This part is for Location view import -->
<!-- ********************************** -->
        <Location complete_topology="true">
                <Location_Container model_type="Country" name="USA"
                        Security_String="whatever">
                        <Location_Container model_type="Region"
                                name="New Hampshire"
                                complete_topology="false">
                                <Location_Container model_type="Site"
                                name="Durham"
                                        <Device ip_dnsname = "10.253.32.10"/>
                                        <Device ip_dnsname = "172.19.57.93" />
                                </Location_Container>
                        </Location_Container>
                </Location_Container>
                <Location_Container model_type="Building" name="Durham"
                        Security_String="public">
                        <Location_Container model_type="Room" name="my_room"
                                Security_String="hahaha">
                                <Device ip_dnsname="10.253.9.16"
                                        community_string="public"/>
                                <Device ip_dnsname="10.253.9.17" />
                                <Device ip_dnsname = "10.253.9.18"/>
                        </Location_Container>
                </Location_Container>
                <Location_Container model_type="Building" name="Peace"
                        Security_String="aprisma">
                        <Location_Container model_type="Room" name= "Lab 1">
                                <Device ip_dnsname="10.253.7.17"
                                        community_string="public"/>
                                <Device ip_dnsname="192.168.125.161"/>
                        </Location_Container>
                </Location_Container>
        </Location>
<!-- ************************************** -->
<!-- This part is for model update            -->
<!-- ************************************** -->
        <Update>
                <Location_Container model_type="Building" name="Peace"
                        model_modify_author="ltang"/>
                <Device ip_dnsname="172.19.57.93" poll_interval="101"
                        model_name="haha" />
```

```
    <!-- ************************************** -->
    <!-- This part is to update the port ifIndex=2 -->
    <!-- on device nmcss52-5                    -->
    <!-- ************************************** -->
                <Device ip_dnsname="nmcss52-5">
                        <Port identifier_name="ifIndex" identifier_value="2"
                                model_name="port 2" poll_interval="1103"
                                poll_status="false" log_ratio="12"/>
                </Device>
                <Topology_Container model_type="Lan" name="lan2"
                        Security_String="top secret"/>
        </Update>
<!-- ******************************************** -->
<!-- This part is for model and connection deletion -->
<!-- ******************************************** -->
        <Destroy>
                <Device ip_dnsname="10.253.9.19"/>
                <Location_Container model_type="Building" name="Durham"/>
                <Connection>
                        <Device ip_dnsname="nmcss52-5">
                                <Port identifier_name="ifIndex"
                                        identifier_value="1"/>
                        </Device>
                        <Device ip_dnsname="10.253.9.17">
                                <Port identifier_name="ipAddress"
                                        identifier_value="10.253.8.18"/>
                        </Device>
                </Connection>
        </Destroy>
</Import>
```

Appendix D: .modelinggatewayresource.xml

This section contains a copy of the .modelinggatewayresource.xml file. However, this copy might not be the latest version of this file. For the latest version, use the actual file that shipped with your Modeling Gateway Toolkit.

```
<?xml version="1.0" standalone="no"?>

<TopologyImportExportResourceFile>

<!-- ************************************** -->
<!-- SPECTRUM attribute names and IDs that    -->
<!-- are used for topology export and import. -->
<!-- ************************************** -->

<Attributes
    circuit_id                  = "0xc4042f"
    circuit_name                = "0xc40430"
    community_string            = "0x10024"
    agent_port                  = "0x10023"
    DeviceType                  = "0x23000e"
    is_managed                  = "0x1295d"
    log_ratio                   = "0x10072"
    manager_name                = "0x3dc0009"
    model_modify_author         = "0x11025"
    model_name                  = "0x1006e"
    name                        = "0x1006e"
    poll_interval               = "0x10071"
    poll_status                 = "0x1154f"
    TryCount                    = "0x110c5"
    Security_String             = "0x10009"
    subnet_address              = "0x1027f"
    subnet_mask                 = "0x110b8"
    subnet_list                 = "0x11953"
    TimeOut                     = "0x110c4"
    trapIPAddress               = "0x3dc0007"
    unique_id                   = "0x3dc0004"
    Value_When_Orange           = "0x1000d"
    Value_When_Red              = "0x1000e"
    Value_When_Yellow           = "0x1000c"

    LatestErrorStatus           = "456008c"
    Response_Time               = "456008c"

    AttrToWatch                 = "0x12a43"
    MonitorPolicy               = "0x12a3e"
```

```
MonitorPolicy_ID              = "0x12a51"
Generate_Service_Alarms       = "0x12a66"
Special_Cause_List            = "0x12b47"
Cause_List_Control            = "0x12d50"

Contact_Status                = "0x10004"
Port_Status                   = "0x10f1b"
RM_Condition                  = "0x12a40"
Service_Health                = "0x12a40"
Criticality                   = "0x1290c"
Condition                     = "0x1000a"
Condition_Value               = "0x1000b"

AccumulationMethod            = "0x4500007"
GuaranteeControl              = "0x4500022"
GuaranteeNotes                = "0x4500021"
GuaranteeDescription          = "0x12a4b"
GuaranteeType                 = "0x4500018"
ServiceHealthType             = "0x4500019"
ViolationThreshold            = "0x450001e"
ViolationThresholdPercent     = "0x4500024"
WarningThreshold              = "0x450001d"
WarningThresholdPercent       = "0x4500023"

SCHED_Daily_Repeat_Limit      = "0x1299a"
SCHED_Duration                = "0x12993"
SCHED_Recurrence_Multiplier   = "0x1299b"
SCHED_Recurrence              = "0x12994"
SCHED_Start_DoM               = "0x12991"
SCHED_Start_DoW               = "0x12990"
SCHED_Start_Hour              = "0x1298f"
SCHED_Start_Minute            = "0x1298e"
SCHED_Start_Month             = "0x12992"
SCHED_Start_Day               = "0x129e4"
SCHED_DayBitMask              = "0x129da"
SCHED_Start_Year              = "0x129e3"
SCHED_Start_MoY               = "0x12b48"
SCHED_Description             = "0x12bbc"

SLA_Control                   = "0x4500015"
SLA_Notes                     = "0x4500017"
SLA_ExpirationDate            = "0x4500025"
SLA_Description               = "0x12a4b"

DefaultMaxRTT                 = "0x4500001"
DefaultMeasureInterval        = "0x4500002"

CustomerID                    = "0x12a44"
CustomerField4                = "0x12a39"
```

```
    CustomerField5                = "0x12a3a"
    CustomerField6                = "0x12a3b"
    CustomerField7                = "0x12a3c"

    Contact_Name                  = "0x12a20"
    Contact_Title                 = "0x12a21"
    Contact_Location              = "0x12a22"
    Email_Address                 = "0x12a27"
    Phone_Number                  = "0x12a23"
    Mobile_Phone_Number           = "0x12a24"
    Pager_Number                  = "0x12a25"
    Fax_Number                    = "0x12a26"
    User_Defined_1                = "0x12a28"
    User_Defined_2                = "0x12a29"
    User_Defined_3                = "0x12a2a"
    User_Defined_4                = "0x12a2b"

    Secondary_Contact_Name        = "0x12a2c"
    Secondary_Contact_Title       = "0x12a2d"
    Secondary_Contact_Location    = "0x12a2e"
    Secondary_Email_Address       = "0x12a33"
    Secondary_Phone_Number        = "0x12a2f"
    Secondary_Mobile_Phone_Number = "0x12a30"
    Secondary_Pager_Number        = "0x12a31"
    Secondary_Fax_Number          = "0x12a32"
    Secondary_User_Defined_1      = "0x12a34"
    Secondary_User_Defined_2      = "0x12a35"
    Secondary_User_Defined_3      = "0x12a36"
    Secondary_User_Defined_4      = "0x12a37"

    MOT_Threhold                 = "0x450002c"
    MTBF_Threshold                = "0x4500032"
    MTTR_Thresshold                = "0x450002f"

    Policy_Name_List              = "0x12a4a"
    collectionDescription         = "0x12a67"

/>

<!-- ****************************************** -->
<!-- SPECTRUM model type names and handles that  -->
<!-- can be used in topology import XML file.    -->
<!-- ****************************************** -->

<ModelTypes
    Universe                    = "0x10091"
    Network                     = "0x1002e"
    Lan                         = "0x1002d"
    IPClassA                    = "0x103d5"
```

```
IPClassB                = "0x103d6"
IPClassC                = "0x103d7"
LAN_802_3               = "0x1003c"
LAN_802_5               = "0x1003d"
ATM_NETWORK             = "0xaa000f"
EventAdmin              = "0x3dc0000"
GlobalCollection        = "0x10474"

World                   = "0x10040"
Country                 = "0x10041"
Region                  = "0x10042"
Site                    = "0x10043"
Sector                  = "0x10044"
Building                = "0x10045"
Section                 = "0x10046"
Floor                   = "0x10047"
Room                    = "0x10048"

Top_Org                 = "0x102cf"
Enterprise              = "0x102d0"
Subsidiary              = "0x102d1"
Division                = "0x102d2"
Department              = "0x102d3"
Org_Section             = "0x102d4"
Work_Group              = "0x102d5"
Org_Owns                = "0x102da"

Schedule                = "0x10456"
GnSNMPDev               = "0x3d0002"
Fanout                  = "0x100ae"
Pingable                = "0x10290"
WA_Link                 = "0x102e2"
Unplaced                = "0x103d8"

RTM_Test                = "0x4560000"
SM_Service              = "0x1046f"
SM_AttrMonitor          = "0x1046e"
SM_LatencyMon           = "0x4500001"
SM_SLA                  = "0x4500002"
SM_Guarantee            = "0x4500003"
SM_Customer             = "0x1046c"
SM_CustomerGroup        = "0x10477"
SM_ConnectMon           = "0x4500000"
SM_ServiceMgr           = "0x4500006"
CustomerManager         = "0x10478"
SM_Service_Mgt          = "0x4500007"
SM_SLA_Mgr              = "0x4500008"
```

```
    Correlation_Domain          = "0x10467"
    Correlation_Manager         = "0x10469"

/>

<Relations
    Collects                    = "0x10002"
    MaintenenceScheduledBy      = "0x10034"
    SlmAgreesTo                 = "0x4500000"
    SlmGuarantees               = "0x4500001"
    SlmHasGuarantee             = "0x4500002"
    SlmIsMeasuredBy             = "0x4500003"
    SlmMonitors                 = "0x4500004"
    SlmOwns                     = "0x4500005"
    SlmUses                     = "0x4500006"
    SlmWatchesContainer         = "0x4500007"
    SlmContains                 = "0x4500008"
    SlaPeriod                   = "0x4500009"
    SlmSchedulesGuarantee       = "0x450000c"
    SlmHasServiceComponent      = "0x450000a"
    SlmContainsSLAs             = "0x450000b"
    Groups_Customers            = "0x1003e"
    GlobalCollect               = "0x1003b"

/>

<!-- When do_not_process_pre_existing_devices_under_container_node    -->
<!-- is set to true, if a device under a container element is found   -->
<!-- pre-existing in Spectrum, Modeling Gateway will not process that -->
<!-- device, like updating attributes, creating connections, etc.     -->

<ImportConfiguration
    do_not_process_pre_existing_devices_under_container_node = "false"
    import_to_primary_ss_only = "false"
    max_device_creation_threads = "50"
/>

<!-- ******************************************************** -->
<!--                                                          -->
<!-- This is the Modeling Gateway export configuration.       -->
<!--                                                          -->
<!-- ExportConfiguration is the configuration that controls   -->
<!-- what to export.                                          -->
<!--                                                          -->
<!-- export_devices: export device models or not              -->
<!--                                                          -->
<!-- export_containers: export container models or not        -->
```

```
<!--                                                          -->
<!-- export_port_attributes: export port attributes or not    -->
<!--                                                          -->
<!-- export_links: export device links or not                 -->
<!--                                                          -->
<!-- export_topology_layout: export devices and containers    -->
<!--                          x,y coordinates or not          -->
<!--                                                          -->
<!-- export_annotation: export annotations or not             -->
<!--                                                          -->
<!-- export_WA_Link_models: export WA_Link models or not.     -->
<!--                        If it's not, WA_Link models will  -->
<!--                        be treated as transparent.  Link  -->
<!--                        between two device made thru a    -->
<!--                        WA_Link will be exported as a     -->
<!--                        direct link.                      -->
<!--                                                          -->
<!-- export_spectrum_settings: export SPECTRUM settings or not-->
<!--                           like the settings for Fault    -->
<!--                           Isolation, Auto Discovery,     -->
<!--                           VNM Control, etc...            -->
<!--                                                          -->
<!-- export_user_models: export SPECTRUM user modeling,       -->
<!--                     user licenses, privileges and etc.   -->
<!--                                                          -->
<!-- export_service_modeling: export SPECTRUM Service modeling-->
<!--                                                          -->
<!-- export_schedules: export SPECTRUM schedules.             -->
<!--                                                          -->
<!-- export_discovery_configs: export Auto Discovery's        -->
<!--                           configurations.                -->
<!--                                                          -->
<!-- ******************************************************** -->

<ExportConfiguration
    export_devices           = "true"
    export_containers        = "true"
    export_port_attributes   = "true"
    export_links             = "true"
    export_topology_layout   = "true"
    export_annotation        = "true"
    export_WA_Link_models    = "true"
    export_spectrum_settings = "true"
    export_user_models       = "true"
```

```
    export_service_modeling  = "true"
    export_schedules         = "true"
    export_global_collections = "true"
    export_discovery_configs = "true"
    export_from_primary_ss_only = "false"
    export_policy_manager = "true"
/>

<!-- ******************************************************** -->
<!--                                                          -->
<!-- RootContainerToExport specifies the root container in    -->
<!-- SPECTRUM to be exported.                                 -->
<!--                                                          -->
<!-- ******************************************************** -->

<RootContainerToExport model_type="Universe" model_name=""/>

<!-- ******************************************************** -->
<!--                                                          -->
<!-- DeviceExportAttributes is a list of device attributes to -->
<!-- be exported.  If the attribute ID has not been specified -->
<!-- above, "attribute_id" has to be assigned.                -->
<!--                                                          -->
<!-- ******************************************************** -->

<DeviceExportAttributes>

    <name/>
    <model_type attribute_id="0x10000"/>
    <community_string/>
    <agent_port/>
    <poll_interval/>
    <is_managed/>
    <poll_status/>
    <Security_String/>
    <TimeOut/>
    <TryCount/>
    <Criticality/>
    <Value_When_Orange/>
    <Value_When_Red/>
    <Value_When_Yellow/>
    <Redundancy_Admin_Status attribute_id="0x11d2c" />
    <Auto_Reconfigure_Interfaces attribute_id="0x11dd4" />
    <Discover_Connection_After_Linkup_Trap attribute_id="0x11d25" />
    <Device_Discovery_After_Reconfiguration attribute_id="0x11d27" />
    <Generate_Redundancy_Alarms attribute_id="0x11dd6" />
    <Create_Sub_Interfaces attribute_id="0x11f3c" />
    <Topology_Relocate_Model attribute_id="0x11a80" />
    <Disable_Trap_Events attribute_id="0x11cd0" />
```

```
    <Enable_Spectrum_Management attribute_id="0x1295d" />
    <Hibernate_Device attribute_id="0x12aca" />
    <Enable_Event_Creation attribute_id="0x129f8" />
    <Redundancy_Admin_Status attribute_id="0x11d2c" />
    <DeviceCPUUtilization_Threshold attribute_id="0x12ab9" />
    <DeviceCPUUtilization_Reset attribute_id="0x12abb" />
    <DeviceCPUUtilization_Duration attribute_id="0x12bce" />
    <DeviceMemoryUtilization_Threshold attribute_id="0x12aba" />
    <DeviceMemoryUtilization_Reset attribute_id="0x12abc" />
    <DeviceMemoryUtilization_Duration attribute_id="0x12bcf" />

</DeviceExportAttributes>

<!-- ******************************************************** -->
<!--                                                          -->
<!-- ContainerExportAttributes are the container attributes   -->
<!-- to be exported.  If an attribute ID has not been yet     -->
<!-- specified above, "attribute_id" has to be assigned.      -->
<!--                                                          -->
<!-- ******************************************************** -->

<ContainerExportAttributes>
    <name/>
    <Security_String/>
    <subnet_address/>
    <subnet_mask/>
    <subnet_list/>
    <Value_When_Orange/>
    <Value_When_Red/>
    <Value_When_Yellow/>
    <SelectMP_port attribute_id="0x118e4" />
 </ContainerExportAttributes>

<!-- ******************************************************** -->
<!--                                                          -->
<!-- PortExportAttributes are the port attributes to be       -->
<!-- exported.  If an attribute ID has not been specified     -->
<!-- above, "attribute_id" has to be assigned.                -->
<!--                                                          -->
<!-- When export_changed_attribute_only = "true", only the    -->
<!-- port attribute whose value is not equal to the default   -->
<!-- value will be exported.  Otherwise, all specified port   -->
<!-- attributes will be exported.                             -->
<!--                                                          -->
<!-- ******************************************************** -->
```

```
<PortExportAttributes export_changed_attribute_only="true" >
    <poll_interval/>
    <poll_status/>
    <ok_to_poll attribute_id="0x11dd8" />
    <PollPortStatus attribute_id="0x1280a" />
    <LockConnection attribute_id="0x129f1" />
    <TimeOut/>
    <TryCount/>
    <is_managed/>
    <Enable_Event_Creation/>
    <Criticality/>
    <Alarm_On_Link_Down_Trap attribute_id="0x11fc2" />
    <Assert_Link_Down_Alarm attribute_id="0x12957" />
    <Utilization_Threshold attribute_id="0x1294b" />
    <Utilization_Reset attribute_id="0x1294f" />
    <Utilization_Threshold_Violation_Duration attribute_id="0x12be4" />
    <Inbound_Utilization_Threshold attribute_id="0x12d9f" />
    <Inbound_Utilization_Reset attribute_id="0x12da0" />
    <Inbound_Utilization_Threshold_Violation_Duration attribute_id="0x12da2" />
    <Outbound_Utilization_Threshold attribute_id="0x12da3" />
    <Outbound_Utilization_Reset attribute_id="0x12da4" />
    <Outbound_Utilization_Threshold_Violation_Duration attribute_id="0x12da6" />
    <Total_Packet_Rate_Threshold attribute_id="0x12da7" />
    <Total_Packet_Rate_Reset attribute_id="0x12da8" />
    <Total_Packet_Rate_Threshold_Violation_Duration attribute_id="0x12be3" />
    <Error_Rate_Threshold attribute_id="0x1294d" />
    <Error_Rate_Threshold_Reset attribute_id="0x12951" />
    <Error_Rate_Threshold_Violation_Duration attribute_id="0x12be5" />
    <Discarded_Threshold attribute_id="0x1294e" />
    <Discarded_Threshold_Reset attribute_id="0x12952" />
    <Discarded_Threshold_Violation_Duration attribute_id="0x12be2" />
</PortExportAttributes>

<SpectrumConfigurationExport model_type="VNM">
    <Minimum_Disk_Space attribute_id="0x119d2" />
    <Security_String/>
    <Unmanaged_Trap_Handling attribute_id="0x11cce" />
    <Trap_Storm_Rate attribute_id="0x122db" />
    <Trap_Storm_Length attribute_id="0x122da" />
    <Auto_Connects attribute_id="0x11f99"/>
    <Device_Thresholds attribute_id="0x12acd" />
    <Use_Full_Qualified_Host_Name attribute_id="0x12984" />
    <Allow_Non_Admin_SNMP_Community_Edit attribute_id="0x12042" />
    <Edit_Notes_By_Read_Only_User attribute_id="0x12043" />
    <Set_isManaged_By_Read_Only_User attribute_id="0x129f3" />
    <Consolidate_Users_In_Group attribute_id="0x12a1d" />
```

```
    <Copy_Users_When_Copying_Group attribute_id="0x12a5e" />
    <VLAN_Configuration attribute_id="0x129ad" />
    <Log_When_Device_Cannot_Be_Contacted attribute_id="0x12943" />
</SpectrumConfigurationExport>

<SpectrumConfigurationExport model_type="TopologyWrkSpc">
    <Create_WA_Link_Model attribute_id="0x25e0033" />
    <Create_LAN_IP_Subnet_Model attribute_id="0x25e000d" />
    <Create_Physical_Addresses attribute_id="0x25e000c" />
    <Create_Fanout_Models attribute_id="0x25e002e" />
    <Run_ATM_Discovery attribute_id="0x25e002d" />
    <IP_Route_Tables attribute_id="0x25e0006" />
    <Source_Addr_Tables attribute_id="0x25e0025" />
    <Spanning_Tree_Tables attribute_id="0x25e0026" />
    <Proprietary_Disc_Tables attribute_id="0x25e002b" />
    <ARP_Tables attribute_id="0x25e003a" />
    <Traffic_Resolution attribute_id="0x25e002f" />
    <Unmanaged_SNMP_Disc attribute_id="0x25e0034" />
    <New_Device_In_Maint_Mode attribute_id="0x25e0035" />
</SpectrumConfigurationExport>

<SpectrumConfigurationExport model_type="LostFound" >
    <Automatic_Model_Destruction attribute_id="0x11de1" />
    <Model_Destruction_Interval_Hours attribute_id="0x11de3" />
    <Model_Destruction_Interval_Minutes attribute_id="0x11de4" />
</SpectrumConfigurationExport>

<SpectrumConfigurationExport model_type="FaultIsolation">
    <ICMP_Support_Enabled attribute_id="0x11d98" />
    <ICMP_Timeout attribute_id="0x11dab" />
    <ICMP_TryCount attribute_id="0x11dac" />
    <Lost_Device_TryCount attribute_id="0x12a0a" />
    <Contact_Lost_Model_Destruction attribute_id="0x11fa8" />
    <Destruction_Delay attribute_id="0x11fa9" />
    <Destruction_Event_Generation attribute_id="0x11faa" />
    <Router_Redundancy_Retry_Count attribute_id="0x12a09" />
    <Port_Fault_Correlation attribute_id="0x129e6" />
    <Unresolved_Fault_Alarm_Disposition attribute_id="0x129f4" />
    <WA_Link_Fault_Isolation_Mode attribute_id="0x12adc" />
</SpectrumConfigurationExport>

<SpectrumConfigurationExport model_type="LivePipes" >
    <Live_Pipe_Enabled attribute_id="0x11df9" />
    <Alarm_Linked_Port attribute_id="0x11fbd" />
    <Suppress_Linked_Port_Alarms attribute_id="0x11fbe" />
    <Port_Always_Down_Alarm_Suppression attribute_id="0x129fb" />
</SpectrumConfigurationExport>
```

```
<SpectrumConfigurationExport model_type="AlarmMgmt" >
    <Generate_Alarm_Event attribute_id="0x11f5f" />
    <Add_Event_To_Alarms attribute_id="0x11f5c" />
    <Use_Old_Alarm_Event attribute_id="0x11f5d" />
    <Alarm_Update_by_Read_Only attribute_id="0x11f5e" />
    <Alarm_Ageout_Time attribute_id="0x129ea" />
    <Disable_Initial_Alarms attribute_id="0x11f5a" />
    <Disable_Suppressed_Alarms attribute_id="0x11f5b" />
    <Disable_Maint_Alarms attribute_id="0x11f59" />
    <Alarm_Clear_By_Read_Only attribute_id="0x11fb2" />
    <Ageout_Residual_Alarm_Only attribute_id="0x129ec" />
</SpectrumConfigurationExport>

<SpectrumConfigurationExport model_type="PolicyManager" >
    <Policy_Distribution_Mode attribute_id="0x4ad0007" />
</SpectrumConfigurationExport>

<SpectrumConfigurationExport model_type="GlobalConfig" >
    <SNMPv3Profiles attribute_id="0x12bd4" />
    <HibernationCommSuccessTries attribute_id="0x12acb" />
</SpectrumConfigurationExport>

</TopologyImportExportResourceFile>
```

Index

F

G

H

I

L

M

O

P

R

S

T

U

W

X