CA Process Automation

Spectrum IM Connector Guide

Version 01.0.00

Contact CA

Contact CA Support

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

- 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

Provide Feedback

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

If you would like to provide feedback about CA product documentation, complete our short customer survey, which is available on the CA Support website at http://ca.com/docs.

Contents

Chapter 1: Overview

CA Process Automation enables you to define, build, orchestrate, manage, and report on automated processes. These processes can work across multiple operating systems and applications. CA Process Automation processes can automate a wide range of operational functions and roles within an organization.

CA Process Automation offers a set of application-specific Connectors to perform tasks such as gathering data, and performing actions on the target systems and target applications. A connector is a module that integrates third-party software with CA Process Automation.

Spectrum IM connector extends the capability of CA Process Automation by providing an additional module with operators that can be used by CA Process Automation processes to interact with CA Spectrum Infrastructure Management instances, using Spectrum's Java based CORBA APIs to manage CA Spectrum IM Alarms, Events, and Models.

This version of the Spectrum IM connector can be used to interact with versions 9.0, 9.1, and 9.2 of Spectrum IM. If you need to interact with versions 8.x of CA Spectrum IM, you have to install version 3.0 SP1 or earlier of the base CA Process Automation product (formerly CA IT Process Automation Management), and install the version of the Spectrum IM connector provided with those releases.

This section contains the following topics:

Intended Audiences

The *Spectrum IM Connector Guide* is intended for these primary audiences:

- Administrators who install or maintain the Spectrum IM connector or a CA Process Automation instance.
- Developers and Designers of CA Process Automation Processes who want their Processes to interact with Spectrum IM connector.

Terminology

Alarm

An alarm is an object which indicates that a user-actionable, abnormal condition exists in the managed environment.

Connector

A Connector is a module that integrates CA Process Automation with an external software program.

Event

An event is an object representing an instantaneous occurrence within CA Spectrum.

Model

A model represents elements of the computing infrastructure.

Module

A set of functionally related Operators sharing a common configuration.

Operator

A software element that implements an action and is one of the building blocks of a CA Process Automation Process.

Chapter 2: Installing the Spectrum IM Connector

This section contains the following topics:

Prerequisites

Following are the prerequisites for using the connector:

- Spectrum IM connector can be installed on any version of CA Process Automation from version 2.2 or higher.
- Minimum of one Spectrum IM instance of versions 9.0, 9.1, or 9.2 is required to use this connector.

Installation

The Spectrum IM Connector software includes the following versions of the installer:

- IT_PAM_connector_installer_CA_Spectrum_IM_1_x-w32.exe for installing on a Primary Domain Orchestrator running on 32-bit Microsoft Windows systems
- IT_PAM_connector_installer_CA_Spectrum_IM_1_x-w64.exe for installing on a Primary Domain Orchestrator running on 64-bit Microsoft Windows systems
- IT_PAM_connector_installer_CA_Spectrum_IM_1_x.sh for installing on a Primary Domain Orchestrator running on Linux and UNIX

  where x is the current version of the installer.

These installers are included within archive files specific to the operating system on which CA Process Automation is installed.

- IT_PAM_CA_Spectrum_IM_1_x_Connector.zip

  Zip file for Microsoft Windows systems
- IT_PAM_CA_Spectrum_IM_1_x_Connector.tar

  Tar file for Linux and UNIX systems

Before installing the Spectrum IM Connector software, shut down the Primary Domain Orchestrator.

- In a cluster configuration, the first Orchestrator installed is the Primary Domain Orchestrator. To determine whether the local host contains the Domain Orchestrator, open the OasisConfig.properties file and verify that oasis.server.isPrimary=true. The OasisConfig.properties file resides in <itpam_root_directory>\server\c2o\.config, where <itpam_root_directory> is the directory where CA IT PAM was installed.
- In a non-clustered configuration, the sole Domain Orchestrator is the Primary Domain Orchestrator.

To install the Connector

1. Log on to the machine and account used to install the CA IT PAM Primary Domain Orchestrator.
2. Run the appropriate version of the installer for your operating system. For Linux and UNIX systems, run the installer as follows to ensure that it is run under the Bourne shell, and to avoid the need to use chmod to set this file as an executable.
   `sh IT_PAM_connector_installer_CA_Spectrum_IM_1_x.sh`
3. When the Welcome screen displays, click Next to proceed or click Cancel to exit the installation.

   The Licensing terms for this Connector display.
4. If you choose to accept the licensing terms, select "I accept the agreement", and click Next to proceed; otherwise, click Cancel to terminate the installation.

5. Specify the directory where the CA IT PAM Primary Domain Orchestrator was installed (if it differs from the default value) by entering the complete path to the directory, or by clicking Browse and locating the directory.
6. Click Next.

   The installer verifies that the Primary Domain Orchestrator is down, and a dialog displays where you select the components to install. (You only have the choice of this Connector.)
7. Click the check box to select this Connector, and click Next to continue.

   The installer deploys the components of the new module to the Primary Domain Orchestrator (This may takes a few minutes).
8. Click Finish to complete the installation.
9. Restart the Primary Domain Orchestrator.

   The installation.log file is located here: <*itpam_root_directory*>\server\c2o where <itpam_root_directory> is the folder specified in Step 5.

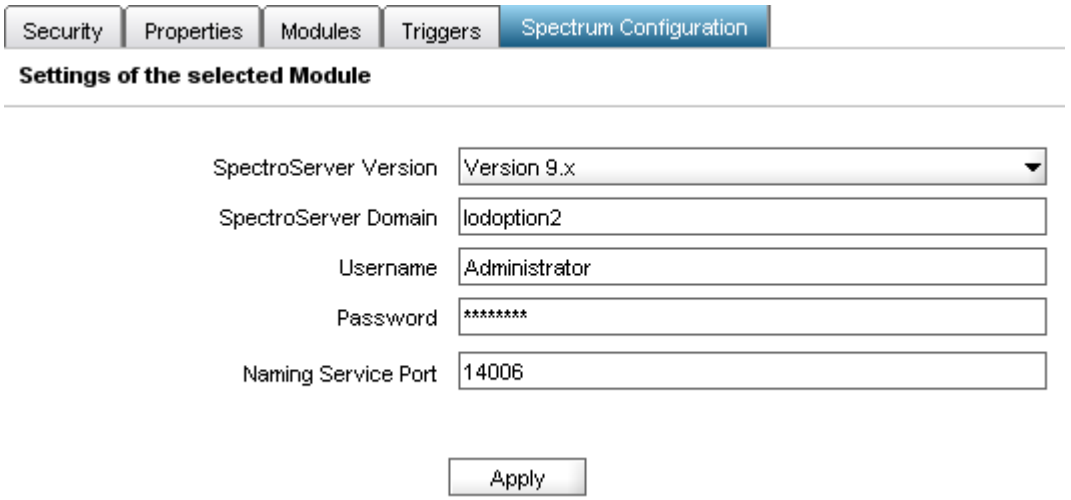
   The newly installed module is not available immediately on any secondary Orchestrators. All components are mirrored within the time specified by the mirroring interval for secondary Orchestrators.
10. After all components have been mirrored, stop and restart each Orchestrator and Agent to make the new module available for use.

Default Module Properties Settings

You configure the following for the SpectroSERVER that is going to be the default for Spectrum operators.

To configure the Spectrum IM connector

1. Provide the values as shown:



SpectroServer Version

Default Spectrum Version for Spectrum operations.

Note: Currently only Version 9.x is supported.

SpectroServer Domain

Default Spectrum Domain for Spectrum operations

Usename

Default username to authenticate with Spectrum

Password

Default Spectrum password

Naming Service Port

Naming Service Port for Spectrum

Default: 14006

These values can be either inherited from Domain or Environment to Orchestrators and Agents or can be set individually.

2. Click Apply to save the configuration.

   You can also configure the individual Spectrum operators to connect different SpectroSERVER. These values are defaults when an individual operator is not configured with these parameters.

Specify the Spectrum IM Module Properties

Module settings are inherited by default from the Domain to each Environment, and from each Environment to Orchestrators in that Environment. For a specific Environment, you can override module settings specified at the Domain level. For a specific Orchestrator, you can override module settings specified at the Environment level or inherited from the Domain.

To specify the module properties

1. In the CA Process Automation Client, select File, Open Configuration Browser (if it is not already open).
2. On the left side of the window, right-click the desired component, and select Lock:
   - Domain (Browser palette)
   - Environment (Browser palette)
   - Orchestrator (Orchestrators palette)
   - Agent (Agents palette)
3. Click the Modules tab in the right pane of the window.
4. Double-click the Spectrum IM Module to display the Properties tab for this module.
5. Specify the appropriate default settings for this module.
6. Click Apply.
7. Click the Save toolbar button.
8. Right-click the component you locked in Step 2 and select Unlock to unlock the component.

   Spectrum operators use the module-level properties as default values unless you specify the corresponding properties for each Operator.

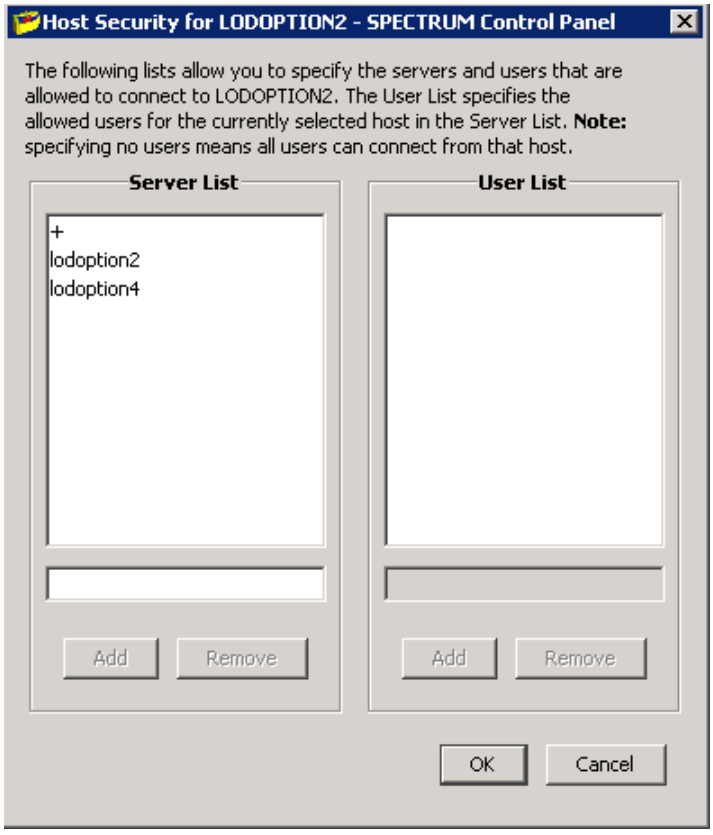
Chapter 3: Using the Spectrum IM Connector

This section contains the following topics:

Configure SpectroSERVER

To configure SpectroSERVER

1. Open the SPECTRUM Control Panel.
2. Add the CA Process Automation Touchpoint host to the Host Security list.

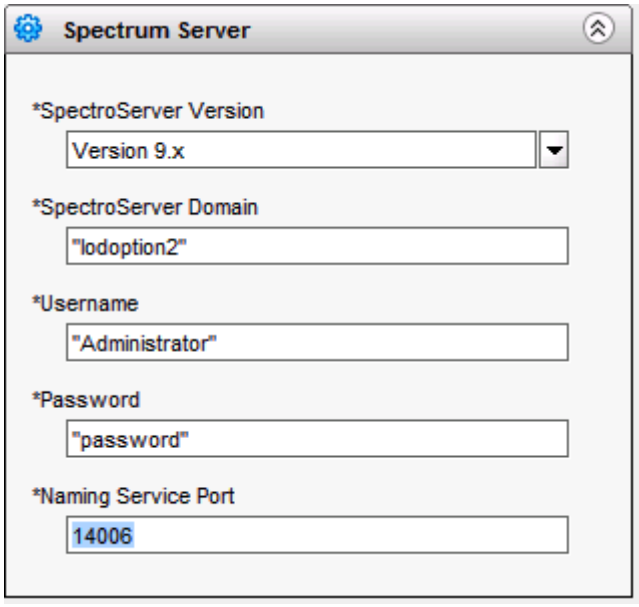


Note: Adding + to the list allows any host to connect to the SpectroSERVER.

Configure Spectrum IM Operators

Spectrum Server Configuration

All Spectrum operators have Spectrum Server Configuration. Based on the Touchpoint the operator is running, the operator inherits the values of the parameters configured from Domain / Environment / Orchestrator / Agent Module configuration parameters. These values can be overwritten by providing values for fields under the Spectrum Server palette.



SpectroServer Version

Default Spectrum Version for Spectrum operations.

Note: Currently only Version 9.x is supported.

SpectroServer Domain

Spectrum Domain for Spectrum operations; leave blank to use module defaults.

Usename

Username of user to authenticate with Spectrum; leave blank to use module defaults.

Password

Spectrum password; leave blank to use module defaults.

Naming Service Port

Naming Service Port for Spectrum.

Default: 14006

Operators for Spectrum IM Alarms

Spectrum IM Alarms are identified uniquely by a SpectroSERVER with its ID. This Unique ID is required in most of alarm operators. A list alarm operator, however, returns certain standard values from an alarm based on the filters configured and Alarm ID is one of the standard values returned.

To find the Alarm ID, you can login to SPECTRUM debug screen and click on Alarm Table. For example, log on to http://<*spectrum_host*>:<*spectrum_web_port*>/spectrum/debug/index.jsp for finding the Alarm ID. Provide the values for <spectrum_host> and <spectrum_web_port based> on your installation.

A sample Alarm ID is 49ea4b6a-03f9-1000-0070-005056b05efd.

Acknowledge Alarm

Acknowledges or Reset Acknowledge an alarm specified by Alarm ID.

Input Parameters

Alarm ID

Unique ID of alarm for the SpectroSERVER

Alarm Acknowledge Status

Options:

Acknowledge or Reset Acknowledge

Output Parameters

None

Clear Alarm

Clears an Alarm specified by Alarm ID.

Input Parameters

Alarm ID

Unique ID of alarm for the SpectroSERVER

Output Parameters

None

Get Alarm Attributes

Gets values for a list of attribute IDs of an Alarm specified by Alarm ID.

Input Parameters

Alarm ID

Unique ID of alarm for the SpectroSERVER

Use dynamic array for attributes

Enable check box to use ICA Process Automation Array as list of attribute IDs of Alarm to return values.

Dynamic array of attribute IDs of Alarm to get values for

IT PAM Dataset String Array with the attribute IDs of Alarm to return values for. Can either be hex or decimal value of the attribute ID. Non integer values will result with value 0 or invalid value.

Static list of attribute IDs of Alarm to get values for

Add attribute IDs of Alarm to the list to return values for. Can either be hex or decimal value of the attribute ID. Non integer values will result with value 0 or invalid value. Individual values are expressions, hence values require quotes (“”) around them.

Output Parameters

Result of successful operation appears in the Operation Results page.

Attr_hex_attr_ID_value

A ValueMap type variable is created for each attribute requested by user. Operator will try to retrieve the attribute properties along with attribute value. The value of attribute shows up as a field called value in the ValueMap. This field can be of type Boolean, Date, Double, Integer, Long, String or String Array depending on the type of attribute received.

Note: Attributes generally defined in hex are returned as Strings rather than Integers even though they are Integers.

Following attribute properties are populated, if it can be retrieved, along with the value field:

ID

Type:

Hex String

Name

Type:

Text String

TypeID

Type:

Hex String

TypeName

Type:

Text String

GroupID

Type:

Hex String

Readable

Type:

Boolean

Writable

Type:

Boolean

IsList

Type:

Boolean

If attribute cannot be found or could not be retrieved, a blank ValueMap with no fields is created.

Get Alarm Detail

Gets all possible data for an Alarm specified by Alarm ID. The operator looks for alarm attributes available in that Spectrum Domain and retrieves values for all attributes found.

Input Parameters

Alarm ID

Unique ID of alarm for the SpectroSERVER

Output Parameters

Result of successful operation appears in the Operation Results page.

Alarm

A single ValueMap type variable called Alarm will be created with names of each attributes obtained as its fields. The values of the attributes will show up as fields <attribute_name> in the Alarm ValueMap. This field can be of type Boolean, Date, Integer, String or String Array depending on the type of attribute received.

Note: Attributes generally defined in hex are returned as Strings rather than Integers even though they are Integers.

List Alarms

Gets a list of Alarms filtered by certain criteria. Some standard useful attributes for the alarms are returned as result. Any field configured in properties of List Alarms that can be calculated to a non-blank field will be used as a filter. Alarm attributes need to satisfy all configured filters, that is, all filters are logical ANDed when retrieving list of alarms.

Input Parameters

Model ID

Hex or Integer Model ID or Handle for which to look for alarms.

Min and Max Severity

Integer values for minimum and maximum severity to filter alarms.

Note: Max value must be specified as leaving it blank will result in maximum value to be zero. Hence no result will be returned.

Earliest and Latest Creation Date

ITPAM Date type values for Earliest and Latest Creation date and time of alarms.

A date type variable can be created dynamically using JavaScript functions like now() and today() or can be parsed from a string using a format like parseDate(date_string, format_string). For example: parseDate("01/01/2009 11:11:11", "MM/dd/yyyy HH:mm:ss").The format can be any valid Java SimpleDateFormat.

If a String type variable is passed, only date of the string will be parsed if it is a valid date. For example if "01/01/2009 11:11:11" is passed, it will form a filter with "01/01/2009 00:00:00" ignoring the time.

Troubleshooter

Substring value that should match troubleshooter (assignment) field of the alarms to filter. Cases are ignored.

Trouble Ticket ID

Substring value that should match Trouble Ticket ID field of the alarms to filter. Cases are ignored.

Status

Substring value that should match status field of the alarms to filter. Cases are ignored.

Acknowledged

Acknowledged status of alarms to filter. Choose from dropdown.

Clearable by user

User clearable property of alarms to filter. Choose from dropdown.

Maximum Alarms to fetch

A number greater than 64 indicating maximum number of alarms to fetch.

Default: 1024

Output Parameters

Result of successful operation appears in the Operation Results page.

Alarms

An Array variable of ValueMaps called Alarms will be created with each item in the array containing data from an alarm retrieved. So the size of the array will be the number of alarms retrieved that satisfied the filter conditions. Values of following attributes are returned for each alarm as ValueMap fields. Each index in Alarms array is a ValueMap.

ID

ServerSpecificID (Integer) – ID of the alarm specific to that SpectroSERVER. It may not be unique in the Domain. This is the ID that shows up when *show alarms* vnmsh command is run.

Type:

Text String

Severity

Type:

Integer

Cause

Type:

Hex String

Preexisting

Type:

Boolean

Priority

Type:

Integer

ModelName

Type:

Text String

Trouble_Ticket_ID

Type:

Text String

User_Clearable

Type:

Boolean

WebContextURL

Type:

Text String

EMS_Model_Handle

Type:

Hex String

Primary_Alarm

Type:

Boolean

Alarm_Source

Type:

Text String

Alarm_Title

Type:

Text String

Acknowledged

Type:

Boolean

Occurrences

Type:

Integer

Creation_Date

Type:

Date

Alarm_Status

Type:

Text String

Troubleshooter

Type:

Text String

Note: Attributes generally defined in hex are returned as Strings rather than Integers even though they are Integers.

List Alarm Attributes

Gets a list of all attributes from Alarm Domain of SpectroSERVER. This operator does not require any input as it is simply talking to the SpectroSERVER and retrieving all possible attributes that an Alarm could have. Certain properties of the attributes that could be retrieved will be listed in an array of ValueMap.

Note: Some attributes which cannot be retrieved using API will not be retrieved. But you may still retrieve the value of that attribute for an Alarm using Get Alarm Attributes operator, where you can pass attribute ID for which to retrieve the value.

Input Parameters

None

Output Parameters

Result of successful operation appears in the Operation Results page.

AlarmAttributes

An Array variable of ValueMaps called AlarmAttributes will be created with each item in the array containing data from an alarm attribute retrieved. So the size of the array will be the number of alarm attributes retrieved from the SpectroSERVER.

Following properties are returned for each attribute as ValueMap fields. Each index in AlarmAttributes array is a ValueMap:

ID

Type:

Hex String

Name

Type:

Text String

Type:

Text String

GroupID

Type:

Hex String

Readable

Type:

Boolean

Writable

Type:

Boolean

IsList

Type:

Boolean

Note: Attributes generally defined in hex are returned as Strings rather than Integers even though they are Integers.

Set Alarm Attributes

Sets attributes of an alarm. The attributes are uniquely identified by their IDs, which is an Integer, normally represented as Hex String. A static list of these attributes and their corresponding values can be set in the properties window of the operation. Optionally, user can also create an array of ValueMaps with two fields in each value map called Keys and Values, that correspond to the attribute IDs and Values respectively.

Input Parameters

Alarm ID

Unique ID of alarm for the SpectroSERVER

Use Array of ValueMaps for Attribute IDs and Values

Enable check box to use IT PAM ValueMap Array as list of attribute IDs and values to set in Alarm specified by Alarm ID.

Array of ValueMaps with Keys and Values for Alarm Attribute IDs and Values

An array of ValueMaps. Each ValueMap needs to have two fields called “Keys” and “Values”. Set the “Keys” field in that to the attribute ID to update and “Values” field to the value to be set. An attribute that is of Boolean type can be set to values [True/False, Yes/No, 1/0]. Some fields may be expecting values that need to set from a pre-defined list. Eg: Troubleshooter field needs to set to one of the troubleshooters that are present in the list of troubleshooters.

For more information about how to create the input parameter programmatically, see Creating IT PAM ValueMap Array Dynamically.

List of Attribute IDs and Values to set in Alarm

Two separate list of IDs and Values respectively. These are IDs and Values of the attributes to be set in alarm. The IDs set in the first list correspond to the Values set in the second list in order. So, first ID in the IDs list will be set to the first value in Values list. An attribute that is of Boolean type can be set to values [True/False, Yes/No, 1/0]. Some fields may be expecting values that need to set from a pre-defined list. Eg: Troubleshooter field needs to set to one of the troubleshooters that are present in the list of troubleshooters.

Note: The number of IDs and Values set on the list should be the same.

Output Parameters

Result of successful operation appears in the Operation Results page.

None

Update Alarm

Updates certain fields of the alarm specified by Alarm ID. The fields that are set to blank will not be updated. In order to set attributes to blank, you have to use Set Alarm Attributes operator.

Input Parameters

Alarm ID

Unique ID of alarm for the SpectroSERVER

Troubleshooter

Set a value to set troubleshooter or assignment for the alarm. This value needs to be a valid troubleshooter in Troubleshooters list of the SpectroSERVER which is being updated. In order to view or create the Troubleshooters, you can go to OneClick, Tools, Utilities, and Troubleshooters.

Trouble Ticket

Set the value of Trouble Ticket ID associated with the Alarm.

Set trouble ticket as URL

Enable check-box to create a hyperlink in trouble ticket field of the Alarm. The label of the hyperlink will be trouble ticket ID and the URL will be set to the value defined in next field.

Trouble ticket URL

URL to be set in the trouble ticket field. The label of this URL will be trouble ticket ID.

Alarm Status

Set the value of Alarm Status. This is a generic text value.

Alarm Acknowledge Status

Select from dropdown whether to acknowledge, reset acknowledge or don't change acknowledge status.

Web Context URL

Set value for Web Context URL field for the alarm. This is a valid URL value and would appear as hyperlink in OneClick console, in the alarm details tab.

Output Parameters

Result of successful operation appears in the Operation Results page.

None

Operators for Spectrum IM Events

Create Event

Creates an event on a model with cause ID specified.

Input Parameters

Model ID

Hex or Integer Model ID / Handle on which to create event.

Event Cause ID

Hex or Integer Cause ID to create events with. The Cause ID needs to be associated with a Cause file in SPECTRUM.

The default event cause files are under:

<SPECROOT>\SG-Support\CsEvFormat

Custom cause files can be created under:

<SPECROOT>\custom\CsEvFormat

Cause file names are of format Eventxxxxxxxx, where xxxxxxxx is an eight digit hex number that represents the cause ID.

Example for the contents of a custom cause file:

{d "%w- %d %m-, %Y - %T"} At {S 1} IT PAM created event for model {m} - {S 2}

The values in curly brackets { } are set dynamically. Please look into SPECTRUM doc on what variables can be used.

The values {S 1} and {S 2} can be set from IT PAM create event operator. The values represent type S (String) and ID (1, 2 or any other number). While creating event you will need to pass an ID and a value to be set in the event. Currently, only string type values can be set for an event.

After adding the Event files, on Spectrum Web console (http://spectrumhost:port/spectrum), go to Administration, EvFormat/PCause Configuration, and click Reload. Then restart OneClick console, for OneClick to recognize the event files created.

Use Array of ValueMaps for Attribute IDs and Values

Enable check box to use IT PAM ValueMap Array as list of attribute IDs and values to set in event being created

Array of ValueMaps with Keys and Values for Event Attribute IDs and Values

An array of ValueMaps. Each ValueMap needs to have two fields called Keys and Values. Set the Keys field in that to the attribute ID to update and Values field to the value to be set.

For the example above, an Array with 2 ValueMaps need to be created. The Keys fields of the ValueMaps will have values 1 and 2 and the Values fields of the ValueMaps will have the values that need to be put in the event.

For more information about how to create the input parameter programmatically, see Creating IT PAM ValueMap Array Dynamically.

List of Attribute IDs and Values to set in Event

Two separate list of IDs and Values respectively. These are IDs and Values of the attributes to be set in event. The IDs set in the first list correspond to the Values set in the second list in order. So, first ID in the IDs list will be set to the first value in Values list.

For the example above, the IDs list will have values 1 and 2 and the Values list will have the corresponding values.

Note: The number of IDs and Values set on the list should be the same.

Output Parameters

Result of successful operation appears in the Operation Results page.

EventID

Unique ID of the event created.

List Events

Gets a list of Events filtered by certain criteria. Some standard useful attributes for the events are returned as result. Any field configured in properties of List Events that can be calculated to a non-blank field will be used as a filter. Event attributes need to satisfy all configured filters, that is, all filters are logical ANDed when retrieving list of events.

Input Parameters

Event ID

Unique ID of event to return. This will need to match the event ID exactly and a single event will be returned.

Min and Max Severity

Integer values for minimum and maximum severity to filter events.

Note: Max value must be specified as leaving it blank will result in maximum value to be zero. Hence it returns no result.

Earliest and Latest Creation Date

ITPAM Date type values for Earliest and Latest Creation date and time of events.

A date type variable can be created dynamically using JavaScript functions like now() and today() or can be parsed from a string using a format like parseDate(date_string, format_string). For example: parseDate("01/01/2009 11:11:11", "MM/dd/yyyy HH:mm:ss").The format can be any valid Java SimpleDateFormat.

If a String type variable is passed, only date of the string will be parsed if it is a valid date.

For example, if "01/01/2009 11:11:11" is passed, it will form a filter with "01/01/2009 00:00:00" ignoring the time.

Created by

Substring value that should match event creator field of the events to filter. Cases are ignored.

Model Name

Substring value that should match model name field of the events to filter. Cases are ignored.

Model Type Name

Substring value that should match model type name field of the events to filter. Cases are ignored.

Acknowledged

Acknowledged status of alarms to filter. Choose from dropdown.

Clearable by user

User clearable property of alarms to filter. Choose from dropdown.

Maximum Events to fetch

A number greater than 64 indicating maximum number of alarms to fetch.

Default: 1024

Output Parameters

Result of successful operation appears in the Operation Results page.

Events

An Array variable of ValueMaps called Events will be created with each item in the array containing data from an event retrieved. So the size of the array will be the number of events retrieved that satisfied the filter conditions.

Values of following attributes are returned for each event as ValueMap fields. Each index in Events array is a ValueMap.

ID

Type:

Text String

ServerSpecificID

ID of the event specific to that SpectroSERVER. It may not be unique in the Domain. This is the ID that shows up when “show events” vnmsh command is run.

Type:

Integer

Severity

Type:

Integer

DateCreated

Type:

Date

Creator

Type:

Text String

EventType

Type:

Hex String

ModelHandle

Type:

Hex String

ModelName

Type:

Text String

ModelTypeName

Type:

Text String

Note: Attributes generally defined in hex are returned as Strings rather than Integers even though they are Integers.

Operators for Spectrum IM Models

Get Model Attributes

Gets values for a list of attribute IDs of an Alarm specified by Alarm ID.

Input Parameters

Model ID

Unique ID of Model for the SpectroSERVER.

Use dynamic array for attributes

Enable check box to use IT PAM Array as list of attribute IDs of Model to return values for.

Dynamic array of attribute IDs of Model to get values for

IT PAM Dataset String Array with the attribute IDs of Model to return values for. Can either be hex or decimal value of the attribute ID. Non integer values will result with value 0 or invalid value.

Static list of attribute IDs of Model to get values for

Add attribute IDs of Model to the list to return values for. Can either be hex or decimal value of the attribute ID. Non integer values results with value 0 or invalid value. Individual values are expressions, hence values require quotes (“”) around them.

Output Parameters

Result of successful operation appears in the Operation Results page.

Attr_hex_attr_ID_value

A ValueMap type variable will be created for each attribute requested by user. Operator will try to retrieve the attribute properties along with attribute value.

The value of attribute will show up as a field called Value in the ValueMap. This field can be of type String or String Array depending on the type of attribute received.

Note: For models there are only two data types which are String and String Arrays.

String values of following data types are populated as follows:

- Boolean: Value is either 1 for true or 0 for false.
- Date: String of format MMM dd, yyy hh:mm:ss AM/PM.

  For example: Feb 19, 2009 6:32:18 PM.
- Integer: Integer value in string.

Following attribute properties are populated, if it can be retrieved, along with the value field:

ID

Type:

Hex String

Name

Type:

Text String

TypeID

Type:

Hex String

TypeName

Type:

Text String

GroupID

Type:

Hex String

Readable

Type:

Boolean

Writable

Type:

Boolean

IsList

Type:

Boolean

If attribute cannot be found or could not be retrieved, a blank ValueMap with no fields is created.

Get Model Detail

Gets all possible data for a Model specified by Model ID. The operator looks for model attributes available in that Spectrum Domain based on the Model Type and retrieves values for all attributes found.

Input Parameters

Model ID

Unique ID of model for the SpectroSERVER

Output Parameters

Model

A single ValueMap type variable called Model will be created with names of each attributes obtained as its fields.

The values of the attributes will show up as fields <attribute_name> in the Model ValueMap. This field can be of type String or String Array depending on the type of attribute received.

Note: For models there are only two data types which are String and String Arrays.

String values of following data types are populated as follows:

- Boolean: Value is either 1 for true or 0 for false.
- Date: String of format MMM dd, yyy hh:mm:ss AM/PM.

  For example, Feb 19, 2009 6:32:18 PM.
- Integer: Integer value in string.

List Models

Gets a list of Models filtered by certain criteria. Some standard useful attributes for the models are returned as result. Any field configured in properties of List Models that can be calculated to a non-blank field will be used as a filter. Model attributes need to satisfy all configured filters, that is, all filters are logical ANDed when retrieving list of models.

Input Parameters

Model Name

- with prefix
- containing string
- with suffix

Partial or full name of the model to filter. There are three name filters. As is evident from the name, models can be filtered using a substring with which the model name starts or is contained in the model name or the model name ends. The filters are not case sensitive.

Model IP Address

String value that should exactly match the IP Address of the model to be filtered.

For example, 192.168.111.111

Use subnet mask for IP Address

Enable this check-box if a subnet mask is to be used to get list of models within a defined subnet

Subnet mask

Substring value that defines a subnet.

For example, 255.255.255.0

Model type name

String value that exactly matches the model type name. The filter ignores cases.

Model in maintenance mode

Select Yes or No from dropdown to get models that are or are not in maintenance mode. If this field is left blank, maintenance mode attribute of the models are ignored.

This field can be set programmatically as follows:

- Strings [Yes, True, 1]

  In maintenance

- Strings [No, False, 0]

  `Not in maintenance`

Model in hibernate mode

Select Yes or No from dropdown to get models that are or are not in hibernate mode. If this field is left blank, hibernate mode attribute of the models are ignored.

This field can be set programmatically as follows:

- Strings [Yes, True, 1]

  In hibernate

- Strings [No, False, 0]

  Not in hibernate

Maximum Alarms to fetch

A number greater than 64 indicating maximum number of models to fetch.

Default: 1024

Output Parameters

Result of successful operation appears in the Operation Results page.

Models

An Array variable of ValueMaps called Models will be created with each item in the array containing data from an alarm retrieved. So the size of the array will be the number of models retrieved that satisfied the filter conditions.

Values of following attributes are returned for each model as ValueMap fields. Each index in Models array is a ValueMap.

Model_Handle

Type:

Hex String

Model_Name

Type:

Text String

Model_Type_Name

Type:

Text String

Model_Type_Handle

Type:

Hex String

Device_Type

Type:

Text String

Network_Address

Type:

Text String

Secure_Domain_Add

Type:

Text String

Create_Time

Type:

Date

Modify_Time

Type:

Date

MAC_Address

Type:

Text String

System_UpTime

Type:

Integer

Contact_Person

Type:

Text String

Location

Type:

Text String

Description

Type:

Text String

Polling_Interval

Type:

Integer

Is_Managed

Type:

Boolean

Is_Not_Hibernate

Type:

Boolean

Notes

Type:

Text String

Note: Attributes generally defined in hex are returned as Strings rather than Integers even though they are Integers.

List Model Attributes

Gets a list of all attributes from SpectroSERVER for a model. Attributes of a Model depend on the Model Type it belongs to. So this operator identifies the type a model belongs to and then retrieves attributes for that Model Type. Certain properties of the attributes that could be retrieved will be listed in an array of ValueMap.

Note: Some attributes which cannot be retrieved using API will not be retrieved. But you may still retrieve the value of that attribute for a Model using Get Model Attributes operator, where you can pass attribute ID for which to retrieve the value.

Input Parameters

Model ID

Unique ID of model for the SpectroSERVER

Output Parameters

Result of successful operation appears in the Operation Results page.

ID

Hex String of only the ID of the model, without the domain ID.

Name

Text String containing name of the Model

DomainID

Hex String of the Domain ID where the model resides. ModelID is usually logical AND of DomainID and ID

TypeID

Hex String of the Model Type ID

TypeName

Text String of Model Type Name

ModelAttributes

An Array variable of ValueMaps called ModelAttributes will be created with each item in the array containing data from a model attribute retrieved. So the size of the array will be the number of model attributes retrieved from the SpectroSERVER.

Following properties are returned for each attribute as ValueMap fields. Each index in ModelAttributes array is a ValueMap.

ID

Type:

Hex String

Name

Type:

Text String

TypeID

Type:

Hex String

TypeName

Type:

Text String

GroupID

Type:

Hex String

Readable

Type:

Boolean

Writable

Type:

Boolean

IsList

Type:

Boolean

Note: Attributes generally defined in hex are returned as Strings rather than Integers even though they are Integers.

Set Model Attributes

Set attributes of a model. The attributes are uniquely identified by their IDs, which is an Integer, normally represented as Hex String. A static list of these attributes and their corresponding values can be set in the properties window of the operation. Optionally, user can also create an array of ValueMaps with two fields in each value map called Keys and Values, which correspond to the attribute IDs and Values respectively.

Input Parameters

Model ID

Unique ID of model for the SpectroSERVER

Use Array of ValueMaps for Attribute IDs and Values

Enable check box to use IT PAM ValueMap Array as list of attribute IDs and values to set in Model specified by Model ID.

Array of ValueMaps with Keys and Values for Model Attribute IDs and Values

An array of ValueMaps. Each ValueMap needs to have two fields called Keys and Values. Set the Keys field in that to the attribute ID to update and Values field to the value to be set. An attribute that is of Boolean type can be set to values [True/False, Yes/No, 1/0]. Some fields may be expecting values that need to set from a predefined list.

For example, Model_Class field needs to set to one of the classes that are present in the list of classes represented by an integer value like 12 for Model_Class Pingable.

For more information about how to create the input parameter programmatically, see Creating IT PAM ValueMap Array Dynamically.

List of Attribute IDs and Values to set in Model

Two separate list of IDs and Values respectively. These are IDs and Values of the attributes to be set in model. The IDs set in the first list correspond to the Values set in the second list in order. So, first ID in the IDs list will be set to the first value in Values list. An attribute that is of Boolean type can be set to values [True/False, Yes/No, 1/0]. Some fields may be expecting values that need to set from a predefined list.

For example, Model_Class field needs to set to one of the classes that are present in the list of classes represented by an integer value like 12 for Model_Class Pingable.

Note: The number of IDs and Values set on the list should be the same.

Output Parameters

Result of successful operation appears in the Operation Results page.

None

Set Model Hibernate

Puts the model in hibernate mode or gets the model out of hibernate mode.

Input Parameters

Model ID

Unique ID of model for the SpectroSERVER

Model in hibernate

Options:

Yes or No

This field can be set programmatically as follows:

- Strings [Yes, True, 1]

  In hibernate

- Strings [No, False, 0]

  Not in hibernate

Output Parameters

None

Set Model Maintenance

Puts model in maintenance mode or get model out of maintenance mode.

Input Parameters

Model ID

Unique ID of model for the SpectroSERVER

Model in Maintenance

Options:

Yes or No

This field can be set programmatically as follows:

- Strings [Yes, True, 1]

  In Maintenance

- Strings [No, False, 0]

  Not in Maintenance

Output Parameters

None

Update Model

Updates certain fields of the model specified by Model ID. The fields that are set to blank will not be updated. In order to set attributes to blank, you have to use Set Model Attributes operator.

Input Parameters

Model ID

Unique ID of model for the SpectroSERVER

Description

Text string to set description of Model.

Contact Person

Text string to set contact person of Model.

Device Location

Text string to set device location of Model.

Device Type

Text string to set device type of Model.

Model Class

Integer value to set model class of Model. This is a value that is present in SPECTRUM as one of the classes. Eg: 12 for class “Pingable”. Check SPECTRUM documents for other possible values.

ICMP Packet Size

Integer value to set Ping packet size for a model.

Enable Event Create

Select from dropdown whether to enable or disable event create for the model.

Options:

Yes or No

This field can be set programmatically as follows:

- Strings [Yes, True, 1]

  Enable Event Create

- Strings [No, False, 0]

  Disable Event Create

In Maintenance

Select from dropdown whether to put model in maintenance mode or get model out of maintenance mode.

Options:

Yes or No

This field can be set programmatically as follows:

- Strings [Yes, True, 1]

  In Maintenance

- Strings [No, False, 0]

  Not In Maintenance

In Hibernate

Select from dropdown whether to put model in hibernate mode or get model out of hibernate mode.

Options:

Yes or No

This field can be set programmatically as follows:

- Strings [Yes, True, 1]

  In Hibernate

- Strings [No, False, 0]

  Not In Hibernate

Polling Interval (sec)

Integer value to set number of seconds for polling interval of a model.

Orange Threshold

Integer value to set orange threshold of a model.

Red Threshold

Integer value to set red threshold of a model.

Orange Threshold

Integer value to set yellow threshold of a model.

Notes

Text string to set notes for a model.

Output Parameters

None

Chapter 4: Troubleshooting

This section contains the following topics:

Spectrum Operator Cannot Connect to Spectrum Server

Valid On Windows and UNIX

Symptom:

Spectrum operator cannot connect to Spectrum server. The error message is “Failed to connect to Spectrum Domain with ID=Server_name”.

Solution 1:

Make sure Spectrum Server setting is set correct, and the SpectroServer is running.

Note: The supported SpectroServer version is 9.x.

Solution 2:

Check that you are able to *telnet* the configured spectrum host and port from the CA Process Automation.

To do this, open a Command Prompt window and execute the command:

`telnet <Spectrum Host> <Spectrum Port>`

Appendix A: Creating CA Process Automation ValueMap Array Dynamically

To create an Array of ValueMaps dynamically

Use the following JavaScript code:

```
var tempValueMapArray = new Array(10);
tempValueMapArray[0] = newValueMap();
tempValueMapArray[0].Keys = “0x10ab1”;
tempValueMapArray[0].Values = “Value to set”;
Process.valueMapArray = tempValueMapArray;
```

Note: You need to create a temporary variable which is not a variable defined in the context of the process where you are using this code. So, tempValueMapArray should not be a variable that is already defined in CA Process Automation process instance.

Index