InCharge™

Service Assurance Manager Configuration Guide

Version 6.2

smarts

Cisco Part Number: OL-6133-01

Contents

Preface

This document provides instructions for the configuration of InCharge Service Assurance Manager, specifically the Global Manager. Topics include creating user profiles, specifying sources of topology and events, and managing topology and notifications.

Intended Audience

This guide is intended for administrators who are responsible for deploying, installing, and configuring the Global Manager. IT managers who seek to understand the role of the Global Manager in the context of an InCharge solution may also find this guide useful.

In addition to the configuration guides for specific components, administrators should also read the *InCharge Service Assurance Manager Business Dashboard Configuration Guide* and the *InCharge System Administration Guide*.

Prerequisites

This guide assumes you have the administrative privileges and the necessary experience to properly install and configure network management software.

Document Organization

This guide consists of the following chapters:

1. OVERVIEW OF INCHARGE SERVICE ASSURANCE MANAGER	Describes Service Assurance and its architecture.
2. CONFIGURATION OVERVIEW	Describes Service Assurance configuration tasks.
3. USER CONFIGURATION FOR THE GLOBAL MANAGER	Describes configuration of user profiles, users, notification lists, and console operations.
4. SYSTEM CONFIGURATION FOR THE GLOBAL MANAGER	Describes configuration of system parameters contained in the *ics.conf* file.
5. MANAGING NOTIFICATIONS WITH THE GLOBAL MANAGER	Describes the properties of notifications and how to manage notifications with the Global Manager.
6. MANAGING TOPOLOGY WITH THE GLOBAL MANAGER	Describes how to organize and manage topology and assign custom icons to map nodes.
7. TOOL CONFIGURATION FOR THE GLOBAL MANAGER	Describes how to create and configure client, server, and automated tools.
8. ESCALATION CONFIGURATION FOR THE GLOBAL MANAGER	Describes how to automate responses to notifications using escalation policies.
9. WORKING WITH FILTERS	Describes how to create filters using the Filter Builder interface.
10. IMPORTING AND EXPORTING CONFIGURATIONS	Describes how to import, export, and modify the configuration of a Global Manager using XML.
A. A.ICIM CLASSES USED WITH MATCHING CRITERIA	Describes the attributes for the ICIM_Notification and UnitaryComputerSystem classes.
B. B.WILDCARD PATTERNS	Describes wildcards used to create matching patterns in the Filter Builder.
C. C.XML REFERENCE	Describes the DTDs and XML syntax for Global Manager configuration files.

Table 1: Document Organization

Documentation Conventions

Several conventions may be used in this document as shown in Table 2.

CONVENTION	EXPLANATION
`sample code`	Indicates code fragments and examples in Courier font
keyword	Indicates commands, keywords, literals, and operators in bold
%	Indicates C shell prompt
#	Indicates C shell superuser prompt
<parameter>	Indicates a user-supplied value or a list of non-terminal items in angle brackets
[option]	Indicates optional terms in brackets
/InCharge	Indicates directory path names in italics
yourDomain	Indicates a user-specific or user-supplied value in bold, italics
File > Open	Indicates a menu path in italics
▼▲	Indicates a command that is formatted so that it wraps over one or more lines. The command must be typed as one line.

Table 2: Documentation Conventions

Directory path names are shown with forward slashes (/). Users of the Windows operating systems should substitute back slashes (\) for forward slashes.

Also, if there are figures illustrating consoles in this document, they represent the consoles as they appear in Windows. Under UNIX, the consoles appear with slight differences. For example, in views that display items in a tree hierarchy such as the Topology Browser, a plus sign displays for Windows and an open circle displays for UNIX.

Finally, unless otherwise specified, the term InCharge Manager is used to refer to InCharge programs such as Domain Managers, Global Managers, and adapters.

InCharge Installation Directory

In this document, the term ***BASEDIR*** represents the location where InCharge software is installed.

- For UNIX, this location is: */opt/InCharge<n>/<productsuite>*.
- For Windows, this location is: *C:\InCharge<n>\<productsuite>*.

The *<n>* represents the InCharge software platform version number. The *<productsuite>* represents the InCharge product suite that the product is part of.

Table 3 defines the *<productsuite>* directory for each InCharge product.

PRODUCT SUITE	INCLUDES THESE PRODUCTS	DIRECTORY
InCharge IP Management Suite	• IP Availability Manager • IP Performance Manager • IP Discovery Manager • InCharge Adapter for HP OpenView NNM • InCharge Adapter for IBM/Tivoli NetView	/IP
InCharge Service Assurance Management Suite	• Service Assurance Manager • Global Console • Business Dashboard • Business Impact Manager • Report Manager • SAM Failover System • Notification Adapters • Adapter Platform • SQL Data Interface Adapter • SNMP Trap Adapter • Syslog Adapter • XML Adapter • InCharge Adapter for Remedy • InCharge Adapter for TIBCO Rendezvous • InCharge Adapter for Concord eHealth • InCharge Adapter for InfoVista • InCharge Adapter for NetIQ AppManager	/SAM
InCharge Application Management Suite	• Application Services Manager • Beacon for WebSphere • Application Connectivity Monitor	/APP
InCharge Security Infrastructure Management Suite	• Security Infrastructure Manager • Firewall Performance Manager • InCharge Adapter for Check Point/Nokia • InCharge Adapter for Cisco Security	/SIM
InCharge Software Development Kit	• Software Development Kit	/SDK

Table 3: Product Suite Directory for InCharge Products

For example, on UNIX operating systems, InCharge IP Availability Manager is, by default, installed to */opt/InCharge6/IP/smarts*. This location is referred to as ***BASEDIR****/smarts*.

Optionally, you can specify the root of ***BASEDIR*** to be something other than */opt/InCharge6* (on UNIX) or *C:\InCharge6* (on Windows), but you cannot change the *<productsuite>* location under the root directory.

For more information about the directory structure of InCharge software, refer to the *InCharge System Administration Guide*.

InCharge Service Assurance Manager Documentation

The following SMARTS documents are relevant to users of the InCharge Service Assurance Management product suite.

- *InCharge Service Assurance Management Suite Installation Guide*
- *An Introduction to InCharge Service Assurance Manager*
- *InCharge Operator's Guide*
- *InCharge Service Assurance Manager Configuration Guide*
- *InCharge Service Assurance Manager Business Dashboard Configuration Guide*
- *InCharge Service Assurance Manager User's Guide for Business Impact Manager*
- *InCharge Service Assurance Manager User's Guide for Report Manager*
- *InCharge Service Assurance Manager Failover System User's Guide*

The following SMARTS documents are relevant to InCharge Service Assurance Manager adapters.

- *InCharge Service Assurance Manager Notification Adapters User's Guide*
- *InCharge Service Assurance Manager SQL Data Interface Adapter User's Guide*
- *InCharge Service Assurance Manager Adapter Platform User's Guide*
- *InCharge XML Adapter User's Guide*
- *InCharge Service Assurance Manager User's Guide for Remedy Adapter*
- *InCharge Service Assurance Manager User's Guide for Concord eHealth Adapter*
- *InCharge Service Assurance Manager User's Guide for InfoVista Adapter*

Common Abbreviations and Acronyms

The following lists common abbreviations and acronyms that are used in the InCharge guides.

ASL	Adapter Scripting Language
CDP	Cisco Discovery Protocol
ICIM	InCharge Common Information Model
ICMP	Internet Control Message Protocol
IP	Internet Protocol
MSFC	Multilayer Switch Feature Card
MIB	Management Information Base
MODEL	Managed Object Definition Language
RSFC	Router Switch Feature Card
RSM	Router Switch Module
SNMP	Simple Network Management Protocol
TCP	Transmission Control Protocol
VLAN	Virtual Local Area Network

Technical Support

SMARTS provides technical support by e-mail or phone during normal business hours (8:00 A.M.—6:00 P.M. U.S. Eastern and Greenwich Mean Time). In addition, SMARTS offers the InCharge Express self-service web tool. The web tool allows customers to access a personalized web page and view, modify, or create help/trouble/support tickets. To access the self-service web tool, point your browser to:

https://websupport.smarts.com/SelfService/smarts/en-us

U.S.A Technical Support

E-Mail: *support@smarts.com*
Phone: +1.914.798.8600

EMEA Technical Support

E-Mail: *support-emea@smarts.com*
Phone: +44 (0) 1753.878140

Asia-Pac Technical Support

E-Mail: *support-asiapac@smarts.com*

You may also contact SMARTS at:

	U.S.A World Headquarters	United Kingdom
Address	SMARTS 44 South Broadway White Plains, New York 10601 U.S.A	SMARTS Gainsborough House 17-23 High Street Slough Berkshire SL1 1DY United Kingdom
Phone	+1.914.948.6200	+44 (0)1753.878110
Fax	+1.914.948.6270	+44 (0)1753.878111

For sales inquiries, contact SMARTS Sales at:
sales@smarts.com

SMARTS is on the World Wide Web at:
http://www.smarts.com

1

Overview of InCharge Service Assurance Manager

This chapter provides a brief overview of InCharge Service Assurance Manager (Service Assurance) and illustrates its architecture.

The successful deployment of Service Assurance requires knowledge of your operations environment and the management tools already in place. You can integrate Service Assurance with third-party tools and existing installations of InCharge applications.

Please refer to the *InCharge Service Assurance Management Suite Installation Guide* for the hardware and software requirements as well as the procedures for installing Service Assurance.

InCharge Service Assurance Manager includes the following components.

- Global Manager is the software component at the heart of Service Assurance. The Global Manager consolidates topology and event information from multiple underlying domains and provides a central point for monitoring and managing distributed analysis domains.
- Global Console is the primary tool for operators and administrators. The console enables operators to monitor the state of the managed environment and quickly respond to notifications. The Global Console also provides map and topology views. The Global Manager Administration Console enables administrators to create and manage user profiles, notification lists, tools, and escalation policies. As the primary tool for users, the Global Console is typically installed on many hosts.

- InCharge Business Dashboard displays views normally available from the Global Console as InCharge viewlets within your organization's Web Page or a third-party Web Portal. InCharge viewlets can display a variety of InCharge information including an InCharge notification log, status table, map, summary, notification properties, or containment view. Viewlets can also be customized to suit your needs.
- SAM Adapter Platform is a software component that imports SNMP traps and topology and event information from third-party applications or products, normalizes the imported data to the InCharge Common Information Model™ (ICIM), and provides this data to the Global Manager.
- InCharge Business Impact Manager extends the capabilities of InCharge Service Assurance Manager to calculate the business impact of events, and to propagate the impacts to affected business elements (ServiceOfferings, BusinessProcesses, and ServiceSubscribers). The propagated impacts are discrete notifications that are connected through a causal event chain to the authentic problem(s) in the infrastructure responsible for the service disruption.
- InCharge Report Manager enables you to save notifications into a relational database for historical reporting. It includes a database schema that is designed to store the information encapsulated in a notification, provides a daily summary of device availability, and generates reports from the database that can be viewed in a Web browser. In addition, Report Manager includes a reporting engine, default reports, and a customized Web application for requesting reports.

Architecture of Service Assurance

The following architecture scenarios describe the flow of topology and notifications between the different Service Assurance components.

Scenario 1

The components described in the first scenario include the following:

- Service Assurance Components
  - InCharge Broker
  - Global Manager
  - Global Console
  - InCharge Report Manager
- Underlying Domains
  - InCharge SAM Adapters
  - InCharge Availability Manager
  - InCharge Performance Manager

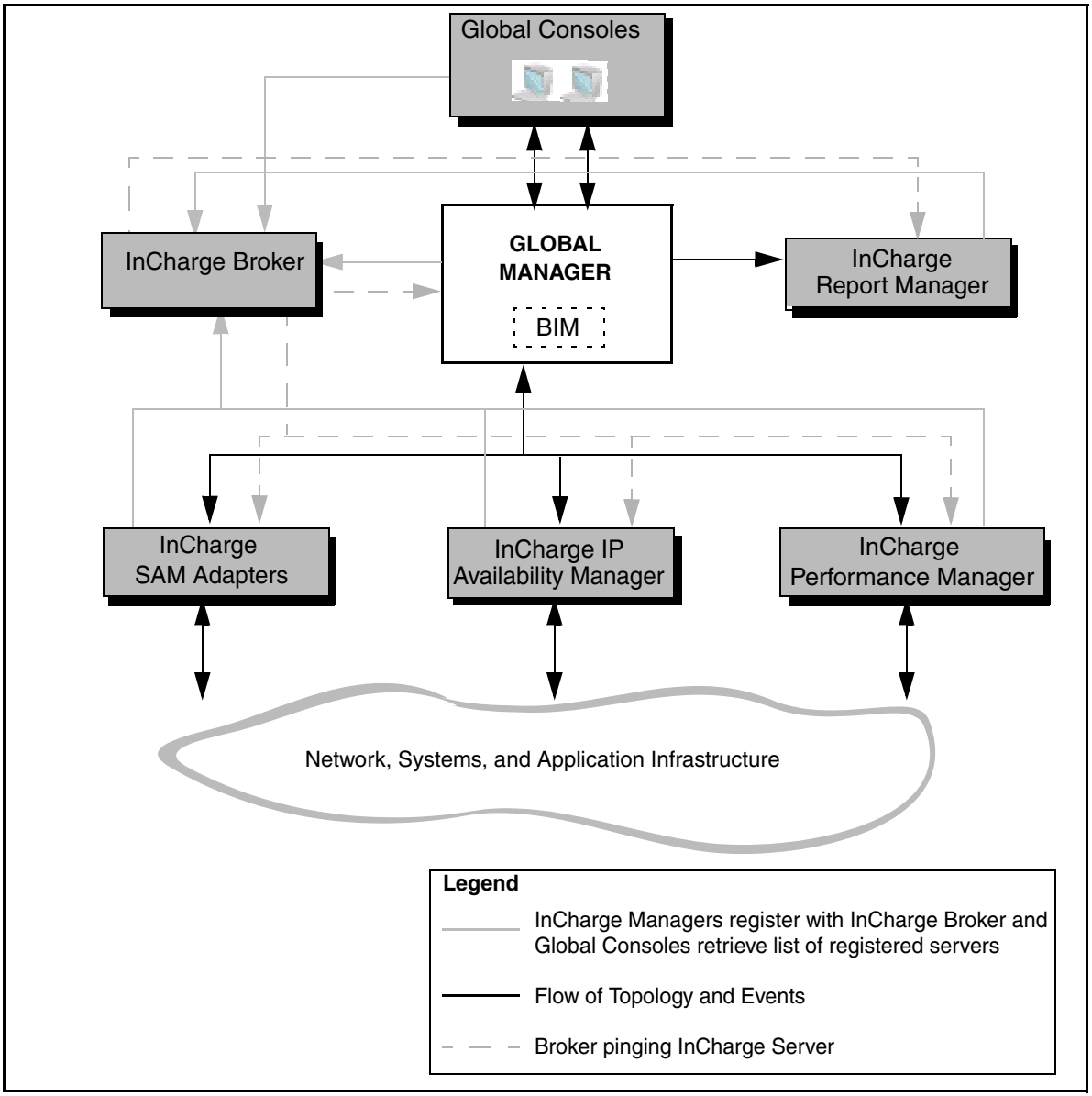

Figure 1: Scenario 1

Scenario 2

This second scenario includes the following: Adapter Platform and the InCharge Business Dashboard. Adapter Platform functions as an underlying domain, processing events and topology from disparate sources and sending them to the Global Manager. The InCharge Business Dashboard provides access to the Global Console using standard Web technologies such as HTML and Java.

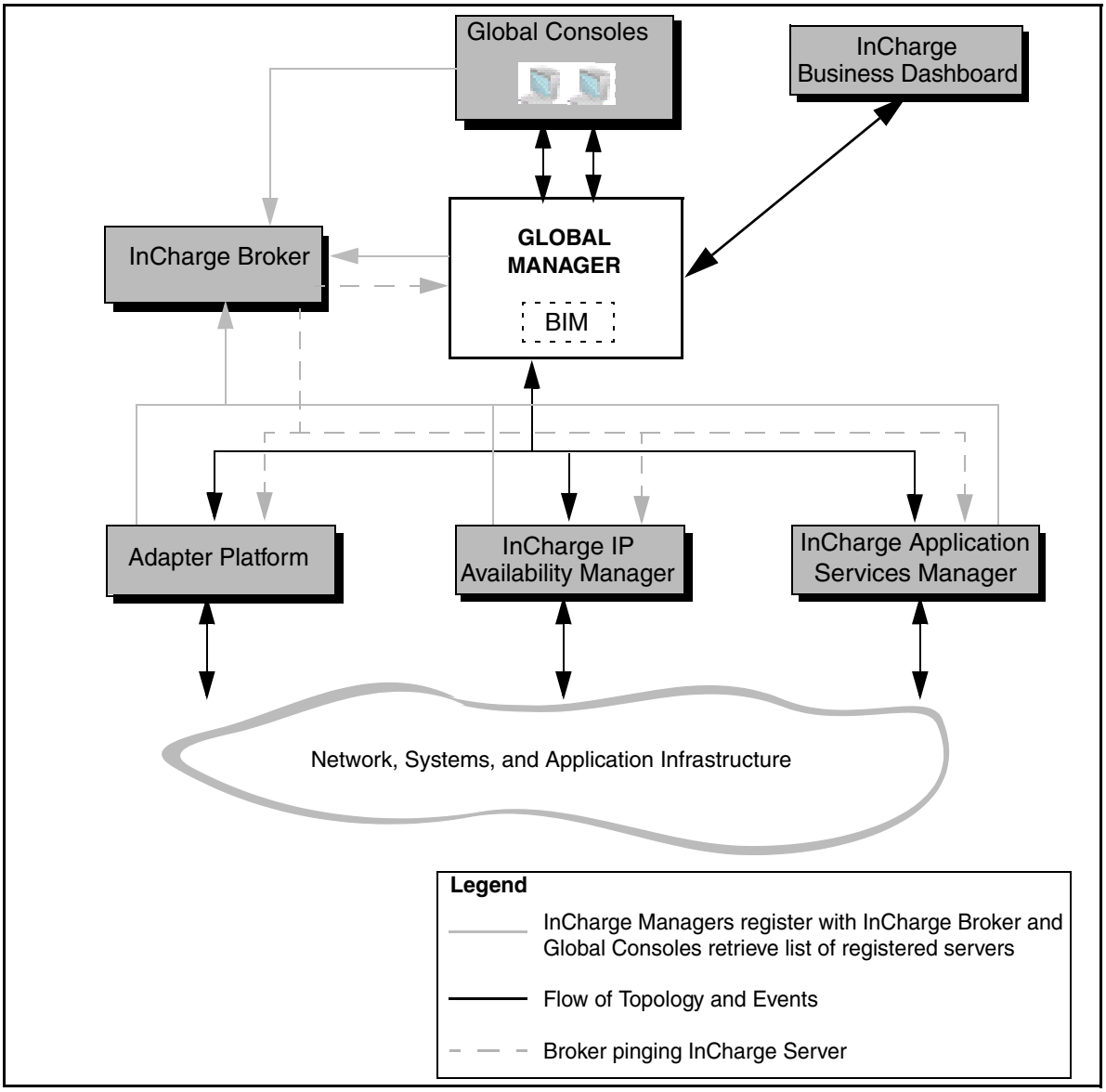

Figure 2: Scenario 2

The Adapter Platform processes events and topology from third-party sources and provides this information to the Global Manager. Adapter Platform can receive events and topology from the following sources:

- SNMP traps – To use process SNMP traps, you must configure the InCharge SNMP Trap Adapter to define the traps you want to send to the Adapter Platform server. Defining traps means that you map the incoming traps to a known notification by the InCharge Common Information Model (ICIM).
- System log files – To produce notifications from a system log file, you must configure the InCharge Syslog Adapter. You must specify the system log file that you want to process and make sure that the format of the log file matches the format described in the Syslog Adapter's ASL rule set.
- Third-party sources with **sm_ems** – To process events from a third party source, you must configure the event source to invoke the **sm_ems** utility. You must invoke **sm_ems** so that it can notify or clear events, update the values of attributes, or create topology elements.

For information about configuring these adapters and utilities, as well as the Adapter Platform server, see the *InCharge Service Assurance Manager Open Integration Configuration Guide*.

InCharge Business Dashboard

The InCharge Business Dashboard displays views normally available from the Global Console as InCharge viewlets within your organization's Web Page or a third-party Web Portal. InCharge viewlets can display a variety of InCharge information including an InCharge notification log, status table, map, summary, notification properties, or containment view. Viewlets can also be customized to suit your needs.

2

Configuration Overview

This chapter describes the configuration tasks associated with the setup and maintenance of a Global Manager. Configuration tasks include creating user profiles, specifying event and topology sources, and managing topology and events. This chapter also describes where you complete the described configuration tasks.

After you install the Global Manager, one of your first configuration tasks is to specify one or more underlying domains in the *ics.conf* configuration file. This process is described in *Defining InCharge Manager Parameters* on page 41. This will provide a source of topology and events and enable the Global Manager to begin working.

While there is much you can do to tailor Service Assurance to your particular environment, the set of default configuration parameters provide a base from which to start. You can further customize Service Assurance while the Global Manager is running.

Global Manager Configuration Tasks

This section categorizes configuration options into groups of related tasks.

- Creating user profiles
- Specifying underlying domains and system-wide parameters
- Managing notifications and topologies
- Automating Responses to Notifications

Each of these groups includes several options, as described below.

Creating User Profiles

A user profile defines the environment for one or more users, that is, anyone who requires access to the Global Manager through the Global Console. You can create user profiles based on role (administrator/operator), job function (network management/helpdesk), or a combination of these and more. With a user profile, you can specify all of the following:

- Notification lists determine which notifications the Global Manager shows to a particular user. You define one or more filters in a notification list to determine which notifications are available through that notification list. For more information about notification lists, see *Managing Notifications with the Global Manager* on page 55.
- Console operations are user actions on the console and determine a user's access privileges to various functions and commands provided by the Global Console. Such actions range from adding a view, to acknowledging a notification, and saving a console. You can select the console operations associated with each user profile. For more information about configuring console operations, see *Console Operations* on page 34.
- Tools provide a means by which users can respond to a notification. Such a response might include pinging a device to see if it is reachable or opening a trouble ticket. By defining the tools associated with a user profile, you determine what actions a user can invoke in response to a notification. For more information on creating tools, see *Tool Configuration for the Global Manager* on page 83.
- Saved consoles provide users with a defined view of the network. This defined view can equal multiple displays of information. For more information about creating a saved console, see *Creating a Saved Console* on page 28.

Configuring System Parameters

System configuration includes tasks that define the operation of the Global Manager, including:

- Specifying sources of topology and events and defining a smoothing interval and a minimum certainty for notifications received from each underlying domain.
- Setting parameters related to the archival of notifications.
- Defining the interaction between the Global Manager and Global Consoles
- Defining the interaction between the Global Manager and underlying domains.

For information about the system defaults, see *Defining System Defaults* on page 45.

Managing Topology and Notifications

Managing the topology and notifications maintained by the Global Manager requires an understanding of how the Global Manager synchronizes topology information and archives notifications. In addition, you may need to perform one or more of the following tasks:

- Uniquely identify elements managed by two or more underlying domains (tagging). For more information regarding tagging, see *Managing Overlapping Elements From Separate Underlying Domains* on page 48.
- Organize managed elements into groups. For information regarding topology configuration, see *Managing Topology with the Global Manager* on page 63.
- Customizing the display of maps with new icons or map backgrounds. For more information, see *Custom Icons for Map Nodes* on page 78.

Automating Responses to Notifications

Escalation policies provide the ability to automatically respond to notifications. For more information regarding escalation policy configuration, see *Escalation Configuration for the Global Manager* on page 103.

Completing Configuration Tasks

Configuration of the Global Manager is accomplished using several different methods:

- Global Manager Administration Console
- *ics.conf* configuration file
- **sm_config** utility

Privilege Requirements

To access the Global Manager Administration Console, you must have the following privileges:

- Your must have *All* privileges, as determined by the *serverConnect.conf* file used by the Global Manager.
- You must have access to the Configure Global Manager Administration Console console operation.

To edit the *ics.conf* file or to invoke the **sm_config** utility requires *All* privileges in the Global Manager's *serverConnect.conf* file as well as the necessary system privileges to edit installed files. Typically, root or Administrator privileges are required to edit InCharge files.

Global Manager Administration Console

The Global Manager Administration Console is a graphical interface for performing configuration tasks related to user profiles, escalation policies, and specifying custom map icons. The following wizards are provided to simplify configuration tasks:

- Notification List Wizard
- Automatic Tool Wizard
- Client Tool Wizard
- Server Tool Wizard
- User Profile Wizard
- User Wizard

When using the User Profile Wizard, you can also create new users and notification lists.

Note: The Global Manager Administration Console is used to configure elements that were previously configured through the *ics.conf* file.

Launching the Global Manager Administration Console

To open the Global Manager Administration Console, select *Configure* > *Global Manager Administration Console*.

The Global Administration Console is divided into two panes with a tree in the left panel and a configuration panel on the right. Selecting an element in the left panel displays content in the configuration panel. At the top level of the tree is the Global Manager you are configuring. Listed under the Global Manager are five elements configured through the console; Escalation Policies, Notification Lists, Tools, Users, and User Profiles. Listed under each element are any instances of these elements. For example, the users *default* and *maint* are listed under the Users element.

To modify the icons displayed in the map consoles, select *Configure* > *Edit Map Icons* from the Global Manager Administration Console.

The configuration of the Global Manager is stored in the server's repository file. You can also import and export this configuration information using the **sm_config** utility, described in *Importing and Exporting Configurations* on page 131.

ics.conf: the Global Manager's Configuration File

Additional configuration parameters for the Global Manager are specified in the *ics.conf* configuration file. This file is located in the ***BASEDIR****/smarts/conf/ics* directory and should be edited as described in *Modifying Configuration Files* on page 15.

The Global Manager reads this file at startup to determine its configuration. If you edit this file while the Global Manager is running, you need to reconfigure the Global Manager as described in *Reconfiguring the Global Manager* on page 52.

The *ics.conf* file is divided into the following sections:

- DomainSection specifies the underlying InCharge Managers from which the Global Manager receives event and topology information and associates each underlying InCharge Manager with the proper data exchange file. For more information, see *Defining InCharge Manager Parameters* on page 41.
- TagSection defines IP elements that are managed by two or more underlying InCharge Managers that the Global Manager must keep topologically distinct. For more information, see *Managing Overlapping Elements From Separate Underlying Domains* on page 48.
- SystemDefaultsSection describes system configuration options for the Global Manager. For more information, see *Defining System Defaults* on page 45.
- BusinessSection specifies data files used to import group information into the Global Manager. For more information, see *Managing Topology with the Global Manager* on page 63.

Syntax Conventions of the ics.conf File

The following section describes the syntax of the *ics.conf* configuration file. Configuration files for related Service Assurance components follow the same syntactical conventions. For more information on the configuration files utilized by the Service Assurance components, see *Service Assurance Configuration Files* on page 16.

Each section of the *ics.conf* file starts with a name, such as DomainSection, followed by a pair of curly braces (`{}`). The configuration information for the named section is enclosed by the curly braces. If a section is divided into one or more subsections, each subsection is also named and followed by a pair of curly braces. The configuration information for the subsection is enclosed by the curly braces.

The configuration fields are declared inside of a section or subsection. A configuration field is composed of a parameter, an equals sign (`=`), and the value assigned to the parameter. All non-numeric values must be enclosed in double quotes (`""`). Each configuration field must end with a semicolon (`;`). A line that starts with a pound sign (`#`) is considered a comment and is ignored.

The following example illustrates the syntax for one section of the *ics.conf* file. The DomainSection is composed of two DomainType subsections for InCharge Managers called INCHARGE-AM and BMC-SA.

Note: Field names and their values are case sensitive.

```
DomainSection
{
#   DomainType definition for INCHARGE-AM.
    DomainType
    {
        ConfFile          = "dxa-conn.conf";
        MinimumCertainty  = 0.24;
        SmoothingInterval = 65;
        HookScript        = "ics/dxa-sample-hook.asl";
        Name              = "INCHARGE-AM";
    }

#   DomainType definition for BMC-SA.
    DomainType
    {
        ConfFile          = "dxa-bmc.conf";
        MinimumCertainty  = 0.24;
        SmoothingInterval = 65;
        HookScript        = "ics/dxa-sample-hook.asl";
        Name              = "BMC-SA";
    }
}
```

Modifying Configuration Files

As part of the InCharge deployment and configuration process, you will need to modify certain files. User modifiable files include InCharge tool scripts, configuration files, rule set files, and templates. Original versions of these files are installed into appropriate subdirectories under the ***BASEDIR****/smarts/* hierarchy. For example, on UNIX operating systems the original versions of Global Manager configuration files are installed to */opt/InCharge6/SAM/smarts/conf/ics.*

To edit a user modifiable file, create a local copy of the file in ***BASEDIR****/smarts/local* or one of its subdirectories. For example, a modified *ics.conf* file should be saved to */opt/InCharge6/SAM/smarts/local/conf/ics.* InCharge software is designed to first search for user modifiable files in ***BASEDIR****/smarts/local* or one of its subdirectories. If a modified version of a file is not found in the local area, InCharge software then searches appropriate nonlocal directories.

Note: Original versions of files may be changed or updated as part of an InCharge software upgrade. However, files located in ***BASEDIR****/smarts/local* are always retained during an upgrade.

To facilitate proper file editing, SMARTS provides the *sm_edit* utility with every InCharge product suite. When used to modify an original version of a file, this utility automatically creates a local copy of the file and places it in the appropriate location under ***BASEDIR****/smarts/local*. This ensures that the original version of the file remains unchanged. In both UNIX and Windows environments, you can invoke *sm_edit* from the command line. Optionally, you can configure Windows so that *sm_edit* is automatically invoked when user-modifiable files are double-clicked in Windows Explorer.

To invoke the *sm_edit* utility from the command line, specify the path and the name of the file you want to edit under ***BASEDIR****/smarts*. If multiple InCharge products are running on the same host, you should ensure that you invoke *sm_edit* from the *bin* directory of the product suite whose files you wish to edit. For example, to edit the configuration file for the Global Manager, you invoke the *sm_edit* utility as follows:

```
# /opt/InCharge6/SAM/smarts/bin/sm_edit conf/ics/ics.conf
```

The *sm_edit* utility automatically creates a local copy of the *ics.conf* file in the ***BASEDIR****/smarts/local/conf/ics* directory, if necessary, and opens the file in a text editor. If a local version of the file already exists, the *sm_edit* utility opens the local version in a text editor. In addition, *sm_edit* creates any necessary directories.

For more information about how to properly edit user modifiable InCharge files and how to use the *sm_edit* utility, refer to the *InCharge System Administration Guide*.

Service Assurance Configuration Files

In addition to the *ics.conf*, there are a number of other configuration files for Service Assurance. Configuration files are located in directories that correspond to the component they configure. For example, the ***BASEDIR****/smarts/local/conf/ics* directory contains configuration files related to the Global Manager.

Table 4 lists the configuration files for Service Assurance components.

DIRECTORY	FILE(S)	DESCRIPTION
conf	*serverConnect.conf* *clientConnect.conf* *brokerConnect.conf* *sm_cloneDir.conf*	Security files for configuring access to Service Assurance. For information regarding security, see the *InCharge System Administration Guide*.
	runcmd_env.sh	Used to set environment variables for SMARTS software.
conf/ics	*ics.conf*	Configuration file for the Global Manager. This file contains parameters for domain naming, tagging, system default settings, and business topology.
	topology-group.data.template	Example data file for importing groups into the Global Manager.
	ics-default.xml	XML file defining default notification lists, user profiles, users, and console operations for the InCharge Global Manager. This XML file is read the first time the Global Manager is run.
	ics-config-sample.xml *nlconfig-sample.xml* *profileconfig-sample.xml* *userconfig-sample.xml* *mapgif-sample.xml*	XML files defining sample configurations for notification lists, user profiles, and users for the InCharge Service Assurance Manager.
conf/maps	*map-config.dtd* *map_gif.xml*	XML and definition files defining custom map icons.
conf/xml-if	*icim_xml.dtd* *sample.xml*	Example XML file and the document type definition (DTD) file to define the XML standard used to import ICIM objects into InCharge. Refer to the InCharge XML Adapter User's Guide for more information.

Table 4: Service Assurance Configuration Files

Table 5 lists additional files used for Service Assurance Configuration.

Note: The files listed in Table 5 *cannot* be modified by the user.

DIRECTORY	FILE(S)	DESCRIPTION
conf/ics	*ics-config.dtd* *console_operations.xml* *consoleoper_config.dtd*	XML and definition files defining default notification lists, user profiles, users, and console operations for the InCharge Global Manager. This XML file is read the first time the Global Manager is run.
conf/maps	*map-config.dtd* *map_gif.xml*	XML and definition files defining custom map icons.
conf/xml-if	*icim_xml.dtd*	The document type definition (DTD) file to define the XML standard used to import ICIM objects into InCharge. Refer to the InCharge XML Adapter User's Guide for more information.

Table 5: Non-modifiable Service Assurance Configuration Files

If your deployment involves InCharge Adapters that integrate with any third-party products, there may be additional configuration and XML definition files. For more information regarding adapter configurations, refer to the guide for that adapter.

3

User Configuration for the Global Manager

This chapter describes how to create and modify user profiles, including the following properties of user profiles: users, notification lists, saved consoles, and console operations. Tools, which can also be associated with a user profile are described in *Tool Configuration for the Global Manager* on page 83.

User Profiles

User profiles provide a means by which an administrator can create and control the access and display of users who attach to the Global Manager through the Global Console, including the InCharge Web Console and the InCharge Business Dashboard. For example, an administrator can create a user profile for operators who monitor network infrastructure and a second user profile for operators who monitor application performance. All user accounts for a Global Manager are associated with a user profile.

A user profile includes the following properties:

- A notification list determines what notifications the Global Manager sends to the user(s).
- One or more users defined for the Global Manager. You must also ensure that these users have access to the Global Manager through the *serverConnect.conf* security file. For information regarding security, see the *InCharge System Administration Guide*.

- Server or client tools that enable users to invoke programs in response to notifications. Server and client tools, which are optional, are separately assigned to user profiles.
- One or more saved consoles that open when the user attaches to the Global Manager.
- Console operations, which are user actions on the console and determine a user's access privileges to various functions and commands provided by the Global Console.

You can create user profiles with the Global Manager Administration Console or by using the **sm_config** utility. For information about using the **sm_config** utility, see *Importing and Exporting Configurations* on page 131.

Creating a User Profile

When using the User Profile Wizard, you are prompted to specify the user(s), notification list, tools, saved consoles, and console operations associated with the user profile. Note that you can create users and notification lists from within the User Profile Wizard.

As you proceed through the wizard, you may notice that certain elements, such as users or tools are listed in gray text. This indicates that these elements are not enabled. You can associate disabled elements with a user profile, however such elements are not available to the user profile. For example, if a tool is not enabled, it will not be available to a user until it is enabled. You can enable or disable elements by selecting them in the tree and selecting or deselecting the Enable box in the Configure User Profile panel.

In addition, you can click **Finish** at any point and the wizard will create the user profile. If you are creating a new user profile, any options after the point where you selected **Finish** will be blank. If you copied an existing user profile, the new profile will maintain the options of the copied profile.

Note: Users may have to detach and reattach for changes to their user profile to take effect.

1. From the Global Manager Administration Console, choose *Edit > New User Profile...* or click the **Launch User Profile Wizard** toolbar button.

   This displays the User Profile Creation Wizard.

2. Type the name of the user profile and choose whether to create a new profile or to copy an existing profile. If you choose to copy an existing profile, the remaining configuration options will be identical to the user profile that was copied.

   The name of a user profile cannot be the same as the name of a user. For example, you cannot create a user profile "Operator" and a user "Operator". A suggested naming convention is to name the user profile with a "–profile" extension. For example, you could have a user profile called "Operator–profile."

3. Choose a Notification List from the drop-down menu and click **Next**. Optionally, you could choose New Notification List from the menu to create a new notification list. For more information on creating a notification list, see *Notification List Parameters* on page 31.

4. Choose the user or users associated with this user profile. You can select multiple users by holding the **Ctrl** key when selecting users. Click the **Add** button to associate the user or users with the user profile.

   If the user does not exist, you can create a user by typing the user name in the Create and add new user field and clicking the **Add** button.

   When you are finished, click **Next**.

Note: A user can belong to only one user profile. If the user you select is already a member of another user profile, the user is associated with the new user profile.

5. Choose any appropriate server tools from the list of Available server tools and click the **Add** button to associate them with the user profile. You can select multiple tools by holding the **Ctrl** key while selecting. When finished, click **Next**. For more information about tools, including the differences between client and server tools, see *Tool Configuration for the Global Manager* on page 83.

6. Choose any appropriate client tool(s) from the list of Available client tools and click the **Add** button to associate them with the user profile. You can select multiple tools by holding the **Ctrl** key while selecting. When finished, click **Next**.

7. Choose the saved console(s) that are opened when users attach to the Global Manager. If a saved console is not selected, the default NotificationLog console is opened.

If the appropriate console is not listed, type the name of the console in the "Add new console:" field and click the **Add** button. You can specify the name of a saved console whether or not the saved console exists.

For more information about saving a console, see *Creating a Saved Console* on page 28.

8 Choose the console operations for users associated with this user profile. You can select Other to choose specific console operations or choose one of the default sets of console operations:

- Read Only
- Operator
- Administrator

The default sets of console operations cannot be modified.

You can assign a modified version of one the default sets of console operations by first selecting the set that provides most of the console operations you want to assign. Then, select Other. When you click **Next**, the console operations that correspond to the set you chose are selected but you can enable or disable individual console operations.

Click **Next**.

For more information about access control for console operations, see *Console Operations* on page 34.

9 The final screen of the User Profile Wizard shows all newly created elements, including the new user profile. By default, the user profile is enabled. You can disable it by unchecking the box next to the profile name. Clicking **Finish** displays a dialog confirming that the user profile elements were created.

Deleting a User Profile

To delete a user profile, select the user profile and choose *Edit > Delete*. Alternatively, you can right-click on the user profile in the tree and select **Delete** from the pop-up menu. Note that you can make a user profile inactive without removing it. For more information, see *Disabling a User Profile* on page 23.

Note: You must not delete the default-profile user profile.

When you log in to a Global Manager, if the specified user name (for example, admin or operator) does not correspond to a defined user profile, the default-profile user profile is used instead. If the default-profile user profile does not exist, an error message is displayed and the Global Console will not open.

Modifying a User Profile

From the tree, select the user profile that you wish to modify. The properties of the user profile are displayed in the Configure User Profile panel.

After you modify the user profile, click **Apply** at the bottom of the Configure User Profile panel to apply the changes to the Global Manager. Changes to the User Profile may not be available to console users until they restart the Global Console.

Disabling a User Profile

You can disable a user profile without having to delete it. When users of a disabled profile attach to the Global Manager, they are assigned the *default-profile*. If the *default-profile* is deleted, users associated with a disabled user profile cannot attach to the Global Manager. For more information regarding the *default-profile* user profile, see *Creating a Saved Console* on page 28.

To disable a user profile, un-check the *Enabled* check box and click **Apply**. The name of a disabled user profile is displayed in gray. To enable a user profile, check the *Enabled* check box and click **Apply.**

Changing the Notification List for a User Profile

A notification list determines what notifications a user sees in the Global Console. Changing the notification list associated with a user profile affects all the users of this profile.

To change the notification list for a user profile, select a new notification list from the Notification List menu. If you need to create a new notification list, use the Notification List Wizard, described in *Notification List Parameters* on page 31.

Adding or Removing a User

A user can only be associated with one user profile. If you add a user to a user profile, the user is automatically removed from the previous user profile.

To add a user to a user profile, select the user profile in the tree. From the Configure User Profile panel, click **Modify List** in the Users section. This displays the Modify Users dialog where you can add or remove users from the user profile. When you are finished, close the dialog and click **Apply** at the bottom of the Configure User Profile panel. For more information about creating a new user, see *User Accounts* on page 26.

To remove a user, select the user profile in the tree. From the Configure User Profile panel, select the user from the Users list and click **Remove Selected**. Click **Apply** at the bottom of the Configure User Profile panel.

When you remove a user from a user profile, the user is automatically associated with the *default-profile* until you assign the user to a different user profile. For more information on the default user profiles, see *Creating a Saved Console* on page 28.

Adding or Removing a Tool

Client and server tools are configured separately, however the process for adding or removing a client or server tool from a user profile is the same.

To add a tool to a user profile, select the user profile in the tree. From the Configure User Profile panel, click **Modify List** in the Server Tools or Client Tools section. This displays the Modify Server Tools or Modify Client Tools dialog where you can add or remove tools from the user profile. When you are finished, close the dialog and click **Apply** at the bottom of the Configure User Profile panel.

To remove a tool, select the user profile in the tree. From the Configure User Profile panel, select a tool and click **Remove Selected**. Click **Apply** at the bottom of the Configure User Profile panel.

For more information about creating or configuring tools, see *Tool Configuration for the Global Manager* on page 83.

Adding or Removing a Saved Console

A saved console provides users with a pre-configured view of the network, helping users focus on aspects of the managed system they are to monitor.

To add a saved console to a user profile, select the user profile in the tree. From the Configure User Profile panel, click **Modify List** in the Saved Consoles section. This displays the Modify Saved Consoles dialog where you can add or remove saved consoles from the user profile. When you are finished, close the dialog and click **Apply**.

To remove a saved console, select the user profile in the tree. From the Configure User Profile panel, select a saved console and click **Remove Selected**. Click **Apply** at the bottom of the Configure User Profile panel.

For more information about creating a saved console, see *Creating a Saved Console* on page 28.

Adding or Removing Console Operations

Console operations are the functions and commands a user can invoke through the Global Console. By adding or removing console operations, an administrator can determine the level of access a user has to console operations. For more information about console operations, see *Console Operations* on page 34.

To add or remove console operations for a user profile, select the user profile in the tree. From the Configure User Profile panel, click **Modify List** in the Console Operations section. This displays the Console Operations dialog where you can select or deselect console operations for the user profile. In addition, you can select one of the three default groups of console operations and apply it to the user profile. When you are finished, click **OK** to close the dialog. Click **Apply** at the bottom of the Configure User Profile panel.

About the Default User Profiles

Service Assurance includes four editable user profiles: *admin-profile*, *oper-profile*, *default-profile* and *maint-profile*. The *default-profile, admin-profile,* and *oper-profile* user profiles include the Default notification list and the Notification Log console.

The *default-profile* user profile provides read-only operations and no tools. The *default-profile* user profile is the "default" user. This profile ensures that any user that is not associated with a user profile or whose user profile is disabled, can attach to the Global Manager. The user must still authenticate with the Global Manager.

The *maint-profile* also serves a special purpose, which is to take ownership of notifications that are not relevant to operators. For example, if notifications are generated for interfaces that should not be managed, or for components that are known to be faulty but are not scheduled to be fixed, an administrator can log in as the user *maint* and take ownership of these notifications.

The *admin-profile* includes all console operations access groups, server tools, and client tools, assuming the user has been given "All" privileges on the server side via the *serverConnect.conf* configuration file.

The *oper-profile* also serves a special purpose. This user profile includes:

- all client and server tools
- details about notifications
- maps
- event management, such as acknowledge
- taking ownership of a notification
- groups configurations
- restricted access to configuration and management of user console and views

The Default notification list filters out notifications where the value of the Owner attribute is *maint*. When the user *maint* takes ownership of a notification, that notification is removed from the display of users assigned the Default notification list.

Note that there is a difference between taking ownership of a notification and acknowledging a notification. If a notification is owned and acknowledged and then recurs, the ownership and acknowledgement values are cleared in the new notification. If the notification is owned but not acknowledged and the notification recurs, the ownership is maintained and the recurring notification is also filtered from the operator's display.

For a description of the default notifications lists used by *oper-profile*, *admin-profile*, *default-profile* and *maint-profile*, see *Notification Lists* on page 30.

User Accounts

A user account provides a unique identifier for each person who attaches to the Global Manager with the Global Console. The user name provides a method for tracking who performs the following operations on notifications:

- When a user acknowledges or unacknowledges a notification, their user name is added to the notification's Owner attribute and this information is recorded in the notification's audit trail.

- When a user takes ownership, their user name is added to the notification's Owner attribute and this information is recorded in the notification's audit trail.
- When a user invokes a tool on a notification, the user's name is recorded to the notification's audit trail.

When a user attaches to a Global Manager, the console automatically tries to associate the user with a user profile. If the user is not assigned to a profile, the *default-profile* is used.

Note: User must authenticate with a matching user name and password in the *serverConnect.conf* file used by the Global Manager. For more information, see the *InCharge System Administration Guide*.

Creating a User Account

You can create new users in one of the following ways:

- Through the User Profile Wizard when you create a user profile as described in *User Profiles* on page 19.
- Through the User Wizard as described below.
- Through the **sm_config** utility as described in *Importing and Exporting Configurations* on page 131.

To create a new user with the User Wizard:

1. Select *Edit > New User* in the Global Manager Administration Console or click the **Launch User Wizard** toolbar button. This displays the User Creation Wizard.
2. Type the name of the new user. The user name must be unique. Click **Next**.
3. Choose a user profile for this user from the drop-down list.
4. Click **Next** to display a confirmation screen or **Finish** to create the user.
5. Add the user name and password in the *serverConnect.conf* file used by the Global Manager. For more information, see the *InCharge System Administration Guide*.

Deleting a User Account

To delete a user, select the user in the tree and choose *Edit > Delete*. Alternatively, you can right-click on the user in the tree and select **Delete** from the pop-up menu.

Creating a Saved Console

You can create a console layout for the Global Console, save it to the Global Manager, and associate it with a user profile so that it automatically opens for users associated with that profile. In addition, you can save the console to the Global Manager so that users can open it from the Global Console using *File > Open Remote*.

For information regarding the different ways you can modify the layout of the Global Console, see the *InCharge Service Assurance Manager Operator's Guide*.

There are two methods to set up access to a saved console:

- Specify the console name when you set up a new user profile, the saved console opens automatically when the user attaches to the Global Console.
- Save a console to a directory that users can access. Console users can use the **Open Remote As** command to open consoles in this directory.

Saving a Remote Console for User Access

You can save a console to one of the following locations on the host where the Global Manager is running:

- Use ***BASEDIR****/smarts/local/consoles* for all users.
- Use ***BASEDIR****/smarts/local/consoles/<user>* for a specific user.
- Use ***BASEDIR****/smarts/local/consoles/<user_profile>* for all users assigned a specific user profile.

Note: The ***BASEDIR****/smarts/local/consoles/<user_profile>* directory must be manually created by the administrator.

When a console is saved using *File > Save As > Save Remote As*, the console is saved to the ***BASEDIR****/smarts/local/consoles/<USER>* directory, where <USER> is the InCharge username. Users cannot see or open the consoles saved in another user's directory unless it is manually copied into their own console directory, the ***BASEDIR****/smarts/local/consoles* directory or the ***BASEDIR****/smarts/local/consoles/<user_profile>* directory that corresponds to the user profile associated with their user name.

Providing a Saved Console in a User Profile

To provide a saved console in a user profile, complete the following steps:

1. Configure the console layout and select *File > Save As > Save Remote As*. The console is saved to a directory that corresponds to your user name under ***BASEDIR****/smarts/local/consoles*. For example, if you are logged in as the user "admin", the console is saved to the ***BASEDIR****/smarts/local/consoles/admin* directory.

   When you invoke **Save Remote As**, you are prompted to select the InCharge Managers to which the saved console should automatically reattach in the event of a disconnect. The list includes those InCharge Managers to which you are currently attached. By choosing to automatically reattach to an InCharge Manager, you can omit the attach/detach console operations from the user profile.

2. To have the console available to all Global Console users, move the saved console from its current directory to the ***BASEDIR****/smarts/local/consoles* directory.

   When a saved console is located in this directory, a user can also open the console using *File > Open Remote*.

3. Add the console to the User Profile(s) as described in *Adding or Removing a Saved Console* on page 24.

4. To have the console available to all users of a particular user profile, move the saved console from its current directory to the ***BASEDIR****/smarts/local/consoles/<user_profile>* directory.

Note: The ***BASEDIR****/smarts/local/consoles/<user_profile>* directory must be manually created by the administrator.

Notification Lists

A notification list determines what notifications a client of the Global Manager receives. A notification list includes a filter that the Global Manager uses to process notifications before sending the notifications to the client. In addition to a filter, you can also use a notification list to change the name of column headings displayed in the Notification Log of the Global Console.

Note: Changes to a notification list are not available to console users until they reattach to the Global Manager.

Clients that use notification lists include:

- Global Consoles—The notification list is a property of the user profile. For information about how to associate a notification list with a user profile, see *Creating a User Profile* on page 20.
- Notification adapters—The notification list used by a notification adapter is specified in the NLSubscription section of the adapter's configuration file. For more information about notification adapters, see the *InCharge Service Assurance Manager Notification Adapters User's Guide*.
- Adapter Platform—A Global Manager uses a notification list when it receives events from an Adapter Platform server. The notification list used by Global Manager is specified in the *dxa-oi.conf* file.

Because a notification list controls what notifications a client receives, it affects more than the display of notifications in a Notification Log. A notification list also affects the display of maps in the Global Console. For example, if a notification list filters out all notifications for a router, the status of that router will appear normal, regardless of its actual condition.

The filter(s) specified as part of a notification list should not be confused with the filtering capabilities of the Global Console. The Global Manager performs the filtering defined in a notification list. The notifications that are filtered out by a notification list are not sent to a client. Filters in the Global Console can be used to further refine the filter of a notification list. This filtering is performed by the Global Console and described in the *InCharge Service Assurance Manager Operator's Guide*.

You can create or modify a notification list using the Global Manager Administration Console or the **sm_config** utility. This section describes using the Global Manager Administration Console. For information about the **sm_config** utility, see *Importing and Exporting Configurations* on page 131.

Notification List Parameters

A notification list includes the properties described in Table 6.

FIELD	DESCRIPTION
Name	Name of the notification list, must be unique among notification lists.
Filter and ASL Filter	Type of filter used to filter notifications. You can specify two types of filters: an expression filter or an ASL filter. • An expression filter matches a wildcard expression against the value of a notification attribute. You create an expression filter using the filter builder. For more information regarding expression filters, see *Building Expression Filters* on page 121. • An ASL filter is specified by checking the *ASL Filter* box and typing the name of the ASL file. The ASL filter file must be located in the ***BASEDIR****/smarts/local/rules/ics* directory. You must also specify the ics directory with the name of the filter. For example: *ics/myASLfilter.asl* If no filter is specified, the notification list matches all notifications.
Column Heading	Used to modify the column headings that are displayed in the Notification Log. For a list of notification attributes, see *Attributes for Matching Notification Properties* on page 146.

Table 6: Notification List Parameters

You can use expression filters and ASL filters in combination, as well as specify multiple expressions in an expression filter. However, each filter definition sheet can contain only expression filters or an ASL filter. You cannot have expression filters and ASL filters on the same sheet.

- When an expression filter and an ASL filter are used in combination (they are on two separate sheets), a notification must match either the expression filter or the ASL filter to be sent to the client.
- When multiple expression filters are specified on the same sheet, the notification must match each expression filter to be sent to the client.
- When multiple expression filters are specified on multiple sheets, the notification must match each expression filter on sheet 1 or each expression filter on sheet 2, and so on.

By default, there are two notification lists: Default and Maintenance. The Default notification list specifies an expression filter. The second notification list, Maintenance, does not specify a filter, meaning it matches all notifications.

For information regarding the default notification lists, see *About the Default Notification Lists* on page 33.

Creating a Notification List

To create a notification list using the Global Manager Administration Console:

1. Choose *Edit* > *New Notification List...* or click the **Launch Notification List Wizard** toolbar button. This displays the Notification List Creation Wizard.
2. Type a unique name for the Notification List.
3. Choose to create a new notification list or copy an existing notification list. If a new notification list is being created, all of the filter properties are empty. If you copy a notification list, the notification list properties contain the same values as the copied list.

   Click **Next**.
4. Create an expression filter or type the name of an ASL Filter prefaced by the name of the ***BASEDIR****/smarts/local/rules* directory in which it is located. The ASL filter must exist in the specified location, as the Global Manager loads the filter when you exit the wizard. The filter builder, used to create expression filters, is also used to configure tool and escalation policies and is described in *Working with Filters* on page 121.
5. Edit the column headings that are displayed in the notification log. The left column lists the attributes included in a notification. The right column lists the columns names as they are currently displayed. You can edit the values in the right column by double-clicking on a field.
6. Click **Next** to view the confirmation panel or **Finish** to create the notification list.

   The new notification list is displayed in the Global Manager Administration Console.

Modifying a Notification List

You can edit the filters and the column headings for a notification list.

1. Select the notification list in the tree. This displays the Configure Notification List panel.
2. Click **Edit Filter** to modify the filter. This launches the filter builder, which is described in *Working with Filters* on page 121.
3. Edit the Display Heading as needed. Double-click the display heading in the right column to edit the text. The new display heading will be visible when users reattach to the Global Manager.
4. Click **Apply**.

Note: To modify the value of a UserDefined attribute, you must configure the hook script to populate the field. For more information, see *Customizing User-Defined Notification Attributes* on page 62.

Disabling a Notification List

You can disable a notification list if you do not want it active but do not want to delete it. When a user attaches to a Global Manager and their notification is disabled, their console will not receive any notifications. The console will receive notifications when the notification list is enabled.

To disable a notification list, select the list from the tree in the Global Manager Administration Console. Un-check the *Enabled* check box and click **Apply**. The name of the disabled notification list is displayed in gray.

About the Default Notification Lists

Two notification lists, Default and Maintenance, are included with the Global Manager and defined in the *ics-default.xml* file. The Default notification list is used by the *default-profile, admin-profile, and oper-profile* user profiles. If a user profile is disabled, users of that profile are assigned the Default user profile, which includes the notification list. The Default notification list uses an expression filter that specifies the following conditions:

- Matches any notification in which the value of the Owner attribute is not “maint”.
- Matches any notification in which the value of Owner attribute is not “SYSTEM”.

A notification must match both conditions before it is sent to a client using this notification list.

The Maintenance notification list does not specify a filter. Because of this, it matches all notifications.

The purpose of the default notification lists, and how they are incorporated into the default user profiles is described in *About the Default User Profiles* on page 25.

Console Operations

Console operations are user actions on the console and determine a user's access privileges to various functions and commands provided by the Global Console. Examples of console operations include taking ownership or acknowledging a notification, adding a view to the console, or viewing the details of a notification. An administrator can assign one or more of these tasks to a user profile.

Interaction With InCharge User Name/Password Privileges

An administrator needs *All* privileges, conferred by the Global Manager's *serverConnect.conf* file and console operations access privileges to perform the following administrative tasks.

- Display the *Configure* menu in the Global Console
  - Open the Global Manager Administration Console
  - Open the Domain Manager Administration Console
  - Open the Group Definition Console
  - Open the Topology Builder Console
  - Edit Map Icons
- Save View as InCharge Viewlet
- Edit attributes in underlying domains
- Invoke **Correlate Now** or **Recompute Codebook** in underlying domains
- Manage or unmanage elements in underlying domains

With console operations, an administrator can assign one or more of these tasks to a user profile.

Note: Some features require a separate license, such as Business Services Maps. An administrator can assign any feature to a user, but the feature will not be enabled unless it is licensed.

Default User Profiles

The user profiles included with Service Assurance include the following console operations:

- The *admin-profile* includes all console operations groups, giving users assigned to this profile administrative access.
- The *maint-profile* includes the Detail Information, Map, Views, and Event Management console operations groups, giving users operator access.
- The *default-profile* includes the Detail Information, Map, and Views console operations groups, giving users read-only access.
- The *oper-profile* includes the Detail Information, Map, Views and Event Management console operations groups, giving users operator access.

Console Operations and Groups

Table 7 lists the console operations and their assigned group. To facilitate the configuration of user profiles, the following sets of groups are provided:

- Read Only includes the Detail Information, Map, and Views groups
- Operator includes the groups in Read Only and Event Management
- Administrator includes all groups

When one of these sets is selected, the associated groups and operations are automatically selected. The administrator can then customize the selected groups or operations as described in *Creating a User Profile* on page 20.

GROUP/CONSOLE OPERATION	DESCRIPTION
Detail Information	
Browse	Open a Topology Browser view from the Global Manager and displays the element as the root.
Browse Detail	Open a Topology Browser view from the underlying InCharge Manager and displays the element as the root.
Expand Map Node	Expand a map node to display connected nodes.
Show Containment	Display an element's components.
Map	
Physical Connectivity	Opens a map displaying physical connectivity.
IP Connectivity	Opens a map displaying IP connectivity.
VLAN Connectivity	Opens a map displaying VLAN connectivity.
Membership	Opens a map showing group elements.
Business Services	Opens a map displaying business elements.
Application	Opens a map displaying application elements.
BGP Connectivity	Opens a map displaying BGP connectivity.
OSPF Connectivity	Opens a map displaying OSPF connectivity.
Views	
Notification Log	Provides the ability to access a Notification Log view.
Topology Browser	Provides the ability to access a Topology Browser view.
Summary	Provides the ability to access a Summary View view.
Status Table	Provides the ability to access a Status Table view.
Map	Open a Map Console
View Configuration	
Notification View Filter	Create a filter for a Notification Log.
Summary View Filter	Create a filter for Summary Views.

GROUP/CONSOLE OPERATION	DESCRIPTION
Edit Summary Parameters	Edit a Summary View configuration.
Add Summary	Add a Summary to a Summary View.
Delete Summary	Remove a Summary from a Summary View.
Status Table Configuration	Edit Status Table configuration.
Console	
Save	Save the console to wherever it was originally loaded from; local or remote. If a console has not been previously saved, the system will prompt the user for information on where to save the console.
Save Remote As	Save a console to the Global Manager.
Open Local	Open a console from the local system.
Open Remote	Open a Console from the Global Manager.
InCharge Manager Attach	Attach to an InCharge Manager.
InCharge Manager Detach	Detach from an InCharge Manager.
Views Management	
Add View	Provides the ability to add a view to a console, and also open a new console (for example, Notification Log Console). NOTE: You can only add views or open a console for views for which you have permission to access.
Delete View	Remove a view from the console.
Copy View	Copy a view in the console.
Copy View as New Console	Copy a view in the console and open as a new console.
Set View Context Listening	Set listening context for active view.
Event Management	
Take Ownership	Take ownership of a notification.
Release Ownership	Release ownership of a notification.
Acknowledge	Acknowledge a notification.
Unacknowledge	Unacknowledge a notification.
Notification Properties	Open a notification properties dialog or add notification properties view to console.

GROUP/CONSOLE OPERATION	DESCRIPTION
Administration	
Configure Group	Open a Groups Definition Console.
Configure Global Manager Administration Console	Open a Global Manager Administration Console.
Configure Domain Manager Administration Console	Open a Domain Manager Administration Console.
Configure Map Icons	Specify custom icons for map nodes using the Global Manager Administration Console.
Launch Topology Builder	Open the Topology Builder Console.
Save View as Viewlet	Save a view as a viewlet for the InCharge Business Dashboard.

Table 7: Console Operations and Groups

Console Operations Allowed for All Users

There are additional console operations that are always available but which cannot be removed from the console. Table 8 contains a list of the more useful of these console operations.

Note: There are other operations that cannot be removed from the console that are not included in this table.

CONSOLE VIEW/CONSOLE OPERATION	DESCRIPTION
Notification Log View/Console	
Sort Columns	Sort notifications by selected column.
Select Columns	Select columns to display in the log.
Save log contents to file	Save the log to a file on the local system.
Freeze Display	Freeze the log to prevent display of new notifications.
Summary View/Console	
Layout	Set the number of summaries per row.
Move Summary Left	Move a summary one summary to the left.
Move Summary Right	Move a summary one summary to the right.
Topology Browser View/Console	

CONSOLE VIEW/CONSOLE OPERATION	DESCRIPTION
Select	Control the set of classes or instances displayed in the tree.
Refresh tree	Refresh the tree view.
Make Root	Make the selected instance the root of the tree.
Map View/Console	
Save map	Save a map to the Global Manager.
Edit map filter	Edit classes displayed in a map.
Set background	Load a background image for a map.

Table 8: Console Operations Allowed for All Users

4

System Configuration for the Global Manager

This chapter describes system configuration tasks for Service Assurance Manager. System configuration topics include configuring settings for the underlying domains, managing overlapping elements, and setting system parameters for the Global Manager.

Defining InCharge Manager Parameters

An InCharge Manager refers to an underlying InCharge analysis application, such as Service Assurance Adapter Platform, CiscoWorks 2000 Device Fault Manager or other Global Manager that serves as a source of event and topology data. You specify these domains and the parameters that control the import of event and topology data in the DomainSection of the *ics.conf* configuration file.

The DomainSection is divided into one or more DomainType subsections, each of which defines the configuration for one or more underlying domains. The following example shows the syntax of a DomainSection when the underlying domain is InCharge IP Availability Manager.

```
DomainSection
{
    DomainType
    {
        ConfFile          = "dxa-conn.conf";
        MinimumCertainty  = 0.0;
        SmoothingInterval = 65;
        HookScript        = "ics/dxa-sample-hook.asl";
        Name              = "INCHARGE-AM";
    }
}
```

Table 9 describes the fields of DomainType subsections.

FIELD	DESCRIPTION
ConfFile	The data exchange file that corresponds to the underlying domain. A data exchange file ensures that Global Manager receives the correct event and topology data. These files should not be edited. • *dxa-app-poller* for InCharge Application Connectivity Monitor • *dxa-asm.conf* for InCharge Application Services Manager for 1.1 • *dxa-asm10.conf* for InCharge Application Services Manager version 1.0 • *dxa-bgp.conf* for InCharge Protocol Services Manager • *dxa-bmc.conf* for InCharge Adapter Platform for BMC Patrol • *dxa-conn-perf.conf* for an InCharge application running both IP Availability Manager and IP Performance Manager • *dxa-conn.conf* for InCharge Availability Manager • *dxa-dfm.conf* for CiscoWorks Device Fault Manager • *dxa-oi.conf* for InCharge Adapter Platform server • *dxa-ospf.conf* for InCharge Protocol Services Manager • *dxa-perf.conf* for InCharge Performance Manager • *dxa-sam.conf* for another Global Manager • *dxa-sim.conf* for InCharge Security Infrastructure • *dxa-vhm.conf* for CiscoWorks Voice Health Monitor
MinimumCertainty	Minimum value the Certainty attribute must have before an event is sent to the Global Manager from the underlying domain. Events with a Certainty value below the threshold are discarded. This value must be a number between 0.0 and 0.99. The default value is 0.24.

FIELD	DESCRIPTION
SmoothingInterval	Time, in seconds, an event must be active before it is sent to the Global Manager. The default value is 65 seconds. Note that the smoothing interval does not apply to underlying Adapter Platform servers or other Global Managers.
HookScript	[Optional] Name of an ASL script that modifies a notification. Typically, this is used to add information to one of the user-defined fields of a notification. Hook scripts must be located in the ***BASEDIR***/*smarts/local/rules* directory. You must prefix the name of the script with the directory in which it is located, typically this is the *ics* directory. For example, the hook script *ics/dxa-sample-hook.asl* is located in the ***BASEDIR***/*smarts/local/rules/ics* directory
Name	Name of the underlying domain. You can specify multiple domains with the same configuration by adding Name fields to the DomainType subsection. However, the value of each Name field within a DomainSection must be unique.

Table 9: Fields Defining the DomainSection

If you edit the *ics.conf* file when the Global Manager is running, you must reconfigure the Global Manager to make the changes take effect. To reconfigure the Global Manager, invoke the following command from the ***BASEDIR***/*smarts/bin* directory:

```
% sm_adapter -s <global_manager> ics/ICS_RemoteConfig.asl
```

For more information regarding this command, see *Reconfiguring the Global Manager* on page 52.

Examples of DomainSection Configurations

This section provides an example of a DomainSection. As shown in the example, you can specify two or more underlying domains within a single DomainType by using additional Name fields. To do this, the underlying domains must be of the same type and use the same settings, such as MinimumCertainty. When you specify two or more underlying domains, the name of each domain must be unique within the DomainSection.

DomainSection with Multiple Underlying Domains

```
DomainSection
{
    DomainType
    {
        ConfFile          = "dxa-perf.conf";
        MinimumCertainty  = 0.24;
        SmoothingInterval = 65;
#        HookScript        = "ics/dxa-sample-hook.asl";
        Name              = "INCHARGE-PM";
    }

    DomainType
    {
        ConfFile          = "dxa-conn-perf.conf";
        MinimumCertainty  = 0.24;
        SmoothingInterval = 65;
#        HookScript        = "ics/dxa-sample-hook.asl";
        Name              = "INCHARGE-AM-PM";
    }

    DomainType
    {
        ConfFile          = "dxa-oi.conf";
        MinimumCertainty  = 0.24;
        SmoothingInterval = 65;
#        HookScript        = "ics/dxa-sample-hook.asl";
        Name              = "INCHARGE-OI";
    }

    DomainType
    {
        ConfFile          = "dxa-dfm.conf";
        MinimumCertainty  = 0.24;
        SmoothingInterval = 65;
#        HookScript        = "ics/dxa-sample-hook.asl";
        Name              = "DFM";
    }

    DomainType
    {
        ConfFile          = "dxa-sam.conf";
        MinimumCertainty  = 0.24;
        SmoothingInterval = 65;
#        HookScript        = "ics/dxa-sample-hook.asl";
        Name              = "INCHARGE-SA2";
    }
}
```

By default, the Hookscript fields are commented out. The default ASL hook scripts are placeholders for a script you must provide. A hook script is not required unless you need to perform additional processing on the notifications coming from the underlying domain.

Defining System Defaults

System defaults are system-wide settings that affect the Global Manager and its clients. The following example illustrates the syntax of the SystemDefaultsSection.

```
SystemDefaultsSection
{
   AutoAcknowledgementInterval = 300;

   InactiveAutoArchiveInterval = 14400;

   AuditTrailSizeLimit   = 100;

   SMTPServer = "localhost";

   FetchLocalNotificationProperties = false;

   RMITimeOut = 60;

   NumberOfWorkerThreads = 2;
}
```

Table 10 describes the fields of the SystemDefaultsSection.

FIELD	DESCRIPTION
AutoAcknowledgementInterval	Interval, in seconds, after which an inactive and unowned notification is acknowledged. Notifications that are acknowledged by the Global Manager are owned by the user SYSTEM. Default is 300 seconds.
InactiveAutoArchiveInterval	Interval, in seconds, after which an inactive and acknowledged notification is archived. Default is 14400 seconds (4 hours). If this value is set to zero, archiving is disabled and notifications will not be deleted, causing Global Manager to use more memory.
AuditTrailSizeLimit	Number of audit log entries for each notification that are saved and visible in the Global Console before the log contents are archived. When this limit is reached, half of the entries are written to the notification archive. Default is 100 entries.
SMTPServer	Name of the SMTP mail server through which mail messages sent by the Global Console's Mail tool are sent. This mail server must be reachable from the host where the Global Manager is running. The default is "localhost". This value can be overridden in the Global Console.
FetchLocalNotificationProperties	Controls how the properties of a particular notification are retrieved when requested by a client application. If set to "true", properties are retrieved from the Global Manager. If set to "false", properties are retrieved from the Global Manager and from the underlying domain. The default value is false. Client applications include the Global Console, the dmctl command-line utility, and adapters.
RMITimeOut	Maximum amount of time, in seconds, the Global Manager is allotted to fetch Notification Properties, Find System, or get System Containment information from an underlying InCharge domain. If a negative value is specified, no time-out value is set. The default value is 60 seconds.
NumberOfWorkerThreads	Number of worker threads used by the escalation mechanism in executing tools associated with escalation levels in escalation paths. The number of threads should be increased if the escalation mechanism falls behind in executing tools for escalation levels. Valid values are 1 - 10.

Table 10: Fields Defining SystemDefaultsSection

About Acknowledging Notifications

When a notification is acknowledged, an entry is appended to the notification's Audit Log. In addition, the values of two notification attributes are set: Owner and Acknowledged. The value of the Owner attribute is set to the InCharge user name of whomever acknowledged the event. The value of the Acknowledged attribute is set to TRUE.

If a notification is unacknowledged, another entry is appended to the notification's audit log. The value of the Owner attribute is set to the InCharge user name of whomever unacknowledged the notification and the value of the Acknowledged attribute is set to FALSE.

Auto acknowledgement is a facility designed for notifications that clear before an operator is able to acknowledge them. It is not uncommon for a notification to clear shortly after it appears. Instead of requiring operators to manually acknowledge such notifications, the Global Manager automatically acknowledges unowned notifications after they remain inactive (clear) for the time specified by the AutoAcknowledgementInterval. Notifications acknowledged by the Global Manager have their Owner attribute set to SYSTEM.

After a notification is acknowledged, it is eligible for archiving. The Global Manager will not archive an active or unacknowledged notification.

Archiving Notifications

Old notifications are periodically removed from the Global Manager's repository and archived to a flat file. The notification archive includes the values of the notification's attributes at the time it is archived, including the contents of the audit log. This archive is intended to provide a record of information before it is deleted.

If you wish to generate historical reports based on notification data, see the *InCharge Service Assurance Manager User's Guide for Report Manager*. InCharge Report Manager does not use the notification archive for reporting, but maintains the notification information in a relational database.

Archived notifications are written to the file *<global_manager>.archive*, where *<global_manager>* is the name of the Global Manager. By default, this is INCHARGE-SA so the resulting notification archive is named *INCHARGE-SA.archive*. This file is written to the ***BASEDIR****/smarts/local/logs* directory.

Note: Over a period of time, a busy Global Manager can generate a sizeable archive file. It may be necessary to periodically rotate the notification archive file.

Managing Overlapping Elements From Separate Underlying Domains

A Global Manager collects topology information from multiple underlying domains. In some cases, two or more of these domains can manage elements with the same name. Two such examples are IP networks, which are named using the network address, and partitions, which are assigned names by the underlying domain.

For example, a service provider might use a private IP network address, such as 10.0.0.0, to provide IP addresses to different customers. When the same range of IP addresses are assigned to multiple customers, different topology elements may have the same IP address, and thus the same name.

Partitions are created and named by InCharge IP Availability Manager. All Availability Manager domain managers use the same convention to name partitions. If the topologies of two domain managers each include partitions, it is possible that one or more of the partitions have the same name, as defined by the Name attribute. For more information about partitions, see the *InCharge IP Discovery Guide.*

By default, the Global Manager treats elements with the same name from different underlying domains as a single instance, consolidating the relationships and attributes of the two elements to a single topological instance. The values for the attributes are taken from the underlying domain that performs the most recent topology synchronization.

You can, however, configure the Global Manager to manage elements of the same name from different domains as distinct elements. By specifying a *tag* for one or both of the underlying domains, the Global Manager will create two separate topology elements, each containing the attributes and relationships of the respective objects in the underlying domains. The tag that you specify is appended to the name of the objects in the tagged domain and displayed in the Global Console.

To tag managed elements, you need to complete two tasks:

- Specify a tag in the DomainSection.
- Specify a matching pattern that will match the IP addresses and partitions to be tagged.

The following example adds the tagging syntax to the DomainSection where four underlying domains are specified. The four underlying domains use the same configuration values for the ConfFile, MinimumCertainty, and SmoothingInterval fields. However, no tags are applied to the INCHARGE_1 and INCHARGE_2 domains. A tag, "tag-3", is applied to instances of the INCHARGE_3 domain that match the specified matching pattern. In addition, a tag, "tag-4" is applied to the instances of the INCHARGE_4 domain that match the specified matching pattern. Examples of matching patterns for these tags follow.

```
DomainSection
{
    DomainType
    {
        ConfFile          = "dxa-conn.conf";
        MinimumCertainty  = 0.24;
        SmoothingInterval = 65;
#        HookScript        = "ics/dxa-sample-hook.asl";
        Name              = "INCHARGE_1";
        Name              = "INCHARGE_2";
        Tagging
        {
            Name    = "INCHARGE_3";
            TagType = "Private IP 192";
            Tag     = "tag-3";
        }
        Tagging
        {
            Name    = "INCHARGE_4";
            TagType = "Private IP 10";
            Tag     = "tag-4";
        }

    }
```

Table 11 describes the Tagging fields of the DomainSection.

FIELD	DESCRIPTION
Tagging	Identifies this as the tagging section of a DomainType.
Name	Name of the underlying domain to which the tag is applied. The values of Name fields within the DomainSection must be unique.
TagType	The TagType applied to this domain. This must match a Name field in the TagType subsection of the TagSection in the *ics.conf* file. The TagSection specifies the matching criteria against which instances of the underlying domain are compared.
Tag	The string of characters that are applied as the tag.

Table 11: Tagging Fields of DomainSection

The following example illustrates what the TagSection for the preceding example might look like. The name of TagType "Private IP 192" and "Private IP 10" must match a TagType field in the Tagging section of the DomainSection.

- Matching patterns that begin with "Partition*" match instances of the Partition class.
- Matching patterns that begin with "IP*" match instances of the IP Network class that belong to the 192.168.0.0 and 10.0.0.0 IP networks.
- Matching patterns with an IP address range match both IP addresses and system elements named with an IP address.

```
TagSection
{
  TagType
  {
     Name    = "Private IP 192";
     Pattern = "Partition*|IP*-192.168.*|192.168.<0-255>.
                <0-255>";
  }
  TagType
  {
     Name    = "Private IP 10";
     Pattern = "Partition*|IP*-10.*|10.<0-255>.<0-255>.
                <0-255>";
  }

}
```

Note: The tag pattern must be specified on a single line.

Table 12 describes the fields of the TagSection.

FIELD	DESCRIPTION
TagType	Defines a TagType subsection of the TagSection.
Name	The name of the appropriate TagType section.
Pattern	Matching pattern used to identify managed elements to which the tag is applied. The "*" and other wildcard patterns are described in *Wildcard Patterns* on page 151.

Table 12: Fields Defining the TagSection

How the Global Manager Applies Tags

The Global Manager can apply tags to both topology elements and notifications. The Global Manager finds matching elements by comparing the matching pattern against the Name and DisplayName attributes of both topology elements and notifications from the relevant underlying domain. For matching elements, any additional attributes that match the pattern are also tagged.

As a result, you should construct a matching pattern to match the values of attributes you want tagged. For topology elements, this could include attributes such as Name and DisplayName. For notifications, this could include attributes such as DisplayName, InstanceDisplayName, and InstanceName.

For example, the matching pattern "IP*-172.16.*|172.16.*" could match the following:

- Elements of the IPNetwork class, which are prefixed with "IPNET-". For IPNET-172.16.0.0, this would result in a DisplayName of `172.16.0.0 [tag-3]` and a Name of `IPNET-172.16.0.0_tag-3`.
- Elements of the system classes, such as Host, that are named using the system's IP address. For host 172.16.1.107, this would result in a DisplayName of `172.16.1.107 [tag-3]` and a Name of `172.16.1.107_tag-3`.

- Notifications generated for these elements would also be tagged. For a Host Down notification, this would result in a DisplayName of `Host Down 100%: 172.16.1.107 [tag-3]`, an InstanceDisplayName of `172.16.1.107 [tag-3]`, and an InstanceName of `172.16.1.107_tag-3`.

Note, however, that if the system was named using its host name, this matching pattern would not apply.

Reconfiguring the Global Manager

The Global Manager reads the *ics.conf* file during startup and configures itself accordingly. If you change the *ics.conf* file after the Global Manager is running, you need to invoke a command so that the Global Manager will reload its configuration file. We refer to this procedure as *reconfiguring* the Global Manager. Reconfiguring the Global Manager requires administrative privileges.

To reconfigure the Global Manager, invoke the following command from the ***BASEDIR****/smarts/bin* directory:

```
% sm_adapter -s <global_manager> ics/ICS_RemoteConfig.asl
```

Depending on your security configuration, you may be prompted for your InCharge user name and password.

When the Global Manager reloads the *ics.conf* file, it sends output to the terminal or its log file verifying that it was able to read each section of the configuration file. The following example shows the output when *ics.conf* is successfully reloaded.

```
% sm_adapter -s INCHARGE-SA ics/ICS_RemoteConfig.asl
Server INCHARGE-SA User: admin
admin's Password: XXXXX
ICNF-N-Processing configuration file
    '/opt/smarts/local/conf/ics/ics.conf'
ICNF-N-Successfully processed 'SystemDefaultsSection'
ICNF-N-Successfully processed 'TagSection'
ICNF-N-Successfully processed 'DomainSection'
ICNF-N-Successfully processed 'BusinessSection'
```

If the Global Manager encounters an error while reloading the *ics.conf* file, it sends an error to the terminal or log file and continues to function using its previous configuration. The following example shows the output if the Global Manager encounters a syntax error when reading its configuration file.

```
% sm_adapter -s INCHARGE-SA ics/ICS_RemoteConfig.asl
Server INCHARGE-SA User: admin
admin's Password: XXXXX
ICNF-N-Processing configuration file
    '/opt/smarts/local/conf/ics/ics.conf'
ICNF-E-Line 152: Syntax Error 2007: No matching right brace
'}' found for
    container 'DomainSection' with left brace on line 41
```

5

Managing Notifications with the Global Manager

This chapter describes notifications, the attributes that define a notification's state, and how a notification's state affects the acknowledgement and archival of notifications. This chapter also describes notification lists; what they are, how they are used, and how to create them.

Overview of Notifications

The Global Manager stores the topology and event information that it receives from the underlying domains in its repository. The topology and event information are stored as objects; instances of the classes defined in the InCharge Common Information Model (ICIM). The notifications that Global Manager sends to clients are themselves objects in the Global Manager's repository.

Notification objects, similar to other objects in the Global Manager's repository, have attributes that describe their properties. For notifications, these attributes include the time the event occurred, the type of event, the name of the object where the event occurred, and much more information. For the complete list of notification attributes, see *Attributes for Matching Notification Properties* on page 146.

Notification attributes are used throughout Service Assurance for a variety of purposes:

- Notification attributes correspond to the columns in the Notification Log of the Global Console. Note that attributes names are not exactly the same as the column headings of the Notification Log.
- You can specify a matching pattern against the values of notification attributes for filtering a notification list.
- Ten notification attributes are user-defined. You can write an ASL script that populates these attributes with additional information.

Understanding and Managing Notifications

This section describes the states of an InCharge notification. It also describes the attributes of a notification and how a change in the notification's state affects the values of these attributes. Finally, this section describes the acknowledgement and archival of notifications.

The following notification attributes are related to a notification's state:

- Event State
- First Notify Time
- Last Notify Time
- Last Change Time
- Last Clear Time
- Count

Uniquely Identifying Notifications

Before we describe the states of a notification, it is important to note that each InCharge notification has a unique name. A notification's name is created from the name of the class and instance where the event occurred and the name of the event itself. For example, the notification NOTIFICATION-Router_R1_Down identifies the event Down, which occurred in the instance R1 of the Router class. The Notification Log will not list more than one notification with this name. Instead, if this event occurs again, the Count field is increased accordingly.

States of a Notification

A notification's state is defined by the Event State attribute, which has five possible values. Table 13 describes each possible value of the Event State attribute and shows its relationship to the Active notification attribute.

EVENT STATE VALUE	DESCRIPTION	ACTIVE VALUE
ACTIVE	Event that causes the notification is occurring. The notification is in the "notify" state.	TRUE
WAS_ACTIVE	Global Manager has disconnected from the underlying domain that was the Source for the notification. For more information regarding WAS_ACTIVE, see *The WAS_ACTIVE and SUSPENDED Event States* on page 57.	TRUE
SUSPENDED	Global Manager can no longer retrieve information about the notification. For more information regarding SUSPENDED, see *The WAS_ACTIVE and SUSPENDED Event States* on page 57.	TRUE
INACTIVE	Event that causes the notification is no longer occurring. The notification is in the "clear" state.	FALSE
UNINITIALIZED	State of the notification object when it is first created and before any state is assigned to it.	FALSE

Table 13: Description of Event State Notification Attribute

The WAS_ACTIVE and SUSPENDED Event States

The WAS_ACTIVE event state is used to identify those notifications that are active under two conditions:

- A Global Manager disconnects from the underlying domain that is the event source. In this case, all active notifications from the disconnected domain are marked WAS_ACTIVE.
- A Global Manager is started from a saved repository file. In this case, all active notifications are marked WAS_ACTIVE.

Notifications marked as WAS_ACTIVE remain in the WAS_ACTIVE state until the Global Manager can verify their status.

If the Global Manager is not able to reconnect to the underlying domain(s) that generated these notifications in 1800 seconds (30 minutes), the value of the Severity attribute is set to 4, changing the color of the notification in a Notification Log to blue. The value of 1800 seconds is referred to as the detachTime.

When the connection between the Global Manager and the underlying domain(s) is re-established, the Global Manager does the following:

- Notifications that are still active in the underlying domain have their Event State changed to ACTIVE and their Severity value updated accordingly.
- Notifications that have cleared in the underlying domain remain in the WAS_ACTIVE state until the attachTime has elapsed. The default value of attachTime is 6000 seconds (100 minutes). However, the actual value that is used to calculate when to clear the notifications is determined as follows:
    - When the uptime for the underlying domain is greater than the attachTime, then the value is 240 seconds plus the smoothing interval. The smoothing interval is specified in the DomainType section of the *ics.conf* file. This typically occurs when the server was restarted.
    - When the uptime for the underlying servers is less than the attachTime, the value is attachTime minus uptime. For temporary disconnects, this value is typically used to determine the state of WAS_ACTIVE notifications. This typically occurs when the server was restarted and there was a disconnect.

The SUSPENDED state indicates that the Global Manager is no longer able to retrieve information about an active notification. When an underlying InCharge domain is not able to get to the source of a notification, it suspends the notification. The InCharge domain sends this message to the Global Manager, which in turn suspends the notification. An InCharge domain may not be able to get to the source of a notification because an SNMP agent is not responding or because it received unexpected error values in an SNMP request.

Notifications suspended by the Global Manager have the value of the Severity attribute set to 4, changing the color of the notification in a Notification Log blue.

A Notification's Life Cycle

Table 14 shows the state and the value of certain notification attributes for NOTIFICATION-Router_R1_Down at different time intervals. The columns of the table mirror the columns an operator might see in the Global Console, with two exceptions: Time and Archived. The Time and the Archived columns are provided for this example.

NOTIFICATION-Router_R1_Down notification becomes active (notified) at 2:00. Fifteen minutes later, at 2:15, the notification becomes inactive (clears). At 2:18, the notification returns to the active state and clears again at 2:35. After remaining in the clear state for five minutes, the notification is acknowledged by the Global Manager at 2:40. After remaining acknowledged and inactive for four hours, the notification is archived at 6:40. At 6:55 NOTIFICATION-Router_R1_Down becomes active. Because the previous instance of this notification was archived, the Global Manager resets the Count, First Notify, Last Notify, and Last Change fields.

Time	Active	First Notify	Last Notify	Last Change	Last Clear	Count	Acknowledged	Archived
2:00	TRUE	2:00	2:00	2:00	0	1	—	—
2:15	FALSE	2:00	2:00	2:15	2:15	1	—	—
2:18	TRUE	2:00	2:18	2:18	2:15	2	—	—
2:35	FALSE	2:00	2:18	2:35	2:35	2	—	—
2:40	FALSE	2:00	2:18	2:40	2:35	2	Acknowledged	—
6:40	FALSE	2:00	2:18	2:40	2:35	2	Acknowledged	Archived
6:55	TRUE	6:55	6:55	6:55	0	1	—	—

Table 14: States of NOTIFICATION-Router_R1_Down

Acknowledging and Archiving Notifications

Regardless of its state, a notification can be marked as Acknowledged. An operator can acknowledge a notification through the Global Console. When an operator acknowledges a notification, the operator becomes the owner of the notification. Acknowledging a notification does not change its state. If an inactive acknowledged notification is re-notified, the value of the Acknowledged attribute is set to FALSE and the value of the Owner attribute is cleared.

By default, the Global Manager automatically acknowledges cleared (inactive) and unowned notifications after five minutes. When Global Manager acknowledges a cleared notification, the owner is set to SYSTEM. You can configure this interval through the AutoAcknowledgementInterval.

Using a notification list, you can filter acknowledged notifications so that they do not appear in an operator's display.

After a notification has been acknowledged and is cleared (inactive), it is eligible to be archived. An archived notification is removed from the Global Manager's repository and written to an archive file. The notification archive file is named *<global_manager>.archive* and located in the ***BASEDIR****/smarts/local/logs* directory.

When a notification is archived, the Global Manager treats a recurrence of that notification as though it were the first occurrence. The value of notification attributes start over again: new First Notify time, Count starts at 1, and so on.

The Global Manager archives only notifications that are both inactive and acknowledged. You can configure the acknowledgement and archival of notifications through the AutoAcknowledgementInterval and InactiveAutoArchiveInterval settings.

Configuration Parameters for Acknowledging and Archiving Notifications

The following parameters, defined in the *ics.conf* file, control how and when the Global Manager acknowledges or archives notifications. Any changes made to the parameters of the *ics.conf* file require that you reload the *ics.conf* file to make those changes take effect.

For information about where to set these parameters, see *Defining InCharge Manager Parameters* on page 41.

AutoAcknowledgementInterval

AutoAcknowledgementInterval controls when a *cleared* (inactive) notification is automatically marked as acknowledged by the Global Manager. The interval is calculated from the time when a notification clears. If a notification recurs, or is unacknowledged, the notification is rescheduled for automatic acknowledgement. The default value is 300 seconds (5 minutes). When this interval is set to 0, the Global Manager will not automatically acknowledge cleared notifications.

InactiveAutoArchiveInterval

InactiveAutoArchiveInterval controls when a *cleared* (inactive) and *acknowledged* notification is archived by the Global Manager. The interval is calculated from either the time the notification is cleared or the time a notification is acknowledged (whichever is later). If a notification recurs, or is unacknowledged, the notification is rescheduled for archival. The default value is 14400 seconds (4 hours). When this interval is set to 0, the Global Manager will not automatically archive cleared and acknowledged notifications.

Notification Types and Incrementing OccurrenceCount

The ICIM defines two types of notifications: momentary and durable. A momentary notification has no duration, it describes an event that happened at a specific time. An authentication failure is an example of a momentary event. A durable notification describes an event that is active over a period of time. While the event is active, the problem it causes is still in effect. An example of a durable event is a link failure. You can determine a notification's type by checking the value of the EventType notification attribute.

A notification's type determines how the value of the OccurrenceCount attribute is increased by a Global Manager. In addition, the source of the notification and the conditions under which a notification is sent, also determine how a notification's count is increased.

For example, consider the scenario where notification N_1 is active in a Global Manager and there are two different sources (S1 and S2) for the notification.

- For durable notifications, the OccurrenceCount represents how many times the notification has become active (notified) during the notification's life cycle.

  Using the example above, when S1 notifies, the Global Manager increments the OccurrenceCount to 1. If S2 then notifies, the OccurrenceCount remains 1.

- For momentary notifications, the OccurrenceCount represents how many times the Global Manager has received a notify message regarding the notification. The current state of the notification has no effect.

  Using the example above, when S1 notifies, the Global Manager increments the OccurrenceCount to 1. If S2 then notifies, the OccurrenceCount increments to 2.

  The Global Manager would increment the OccurrenceCount on a re-notify by one, regardless of whether the notification was active. In fact, if notification N_1 occurred 50 times in the underlying domain, the Global Manager would increase the value of the OccurrenceCount by 50.

Customizing User-Defined Notification Attributes

A notification object also includes ten user-definable attributes. By default, these attributes have no value.

To populate the attributes with values, you must create an ASL program and specify it in the HookScript field of a DomainType subsection. See *Defining InCharge Manager Parameters* on page 41 for a description of the HookScript field.

A sample ASL program, *dxa-sample-hook.asl*, is included in the ***BASEDIR****/smarts/rules/ics* directory. This ASL program receives a handle to the notification object. You need to provide any additional functionality. Your ASL script should query the Global Manager or the underlying domain for additional information.

For example, you can populate one of the columns with the number of affected customers (instances of the ServiceSubscriber class) by counting the number of impacted subscribers.

To give the column names in the Global Console more meaningful names, use the method described in *Notification List Parameters* on page 31.

6

Managing Topology with the Global Manager

This chapter describes the topology management functions of the Global Manager. Topics include ensuring a consistent view of topology, organizing topology into groups, and specifying custom icons for map nodes.

For additional information about creating elements for your topology, see the *InCharge Application Services Manager Discovery Guide* or the *InCharge Service Assurance Manager User's Guide for Business Impact Manager*.

Topology Synchronization

A Global Manager imports topology information from the underlying domains specified in its *ics.conf* configuration file. To maintain an up-to-date representation topology, the Global Manager synchronizes its topology with an underlying domain when any one of the following occur:

- The underlying domain is reconnected after a connection loss or a restart.
- The Global Manager is started.
- The underlying domain performs a full or incremental discovery, rediscovers an object, or a manual discovery is initiated.

- A change is made to the DomainType section of the Global Manager's *ics.conf* configuration file. When the Global Manager is reconfigured, it automatically synchronizes its topology with the underlying domains in the changed sections.

Ensuring a Consistent Representation of Topology

A Global Manager imports topology information. Because the topology information comes from disparate sources, it is important that the Global Manager present a correct and consistent representation of the topology. This is especially true when two or more domains manage the same devices.

Newer versions of InCharge applications include an expanded and updated list of certified devices. This might mean that a device classified as Uncertified or Node by an older version of InCharge is classified as a Router or Switch by a newer version of InCharge.

For best results, SMARTS recommends that you upgrade existing InCharge applications to the most recent version. If that is not possible, use the same version of InCharge for all the underlying applications.

Two issues can arise when the Global Manager receives inconsistent topology information:

- The Global Manager imports information about two or more devices with the same name but the devices are classified differently in their respective underlying domains.
- The Global Manager imports the same device from two or more underlying domains but the device is named differently in each underlying domain.

Note: The Global Manager does not import instances of the Unsupported and Undiscovered classes. However, the topology of a Global Manager may include instances, such as Hosts, with a value of Undiscovered for their Certification attribute.

Same Device Classified Differently in Separate Underlying Domains

The Global Manager can receive conflicting topology information when two underlying domains discover the same device but classify it differently. The scenarios where this may occur are:

- A device has the same name but is an instance of different classes in two or more underlying domains. In addition, the device is classified as Host, Uncertified, or Node in one or more of the underlying domains. When this occurs, the Global Manager replaces an instance of a less specific class, Host, Uncertified or Node, with an instance of a more specific class. Incoming event information for the device is consolidated to the instance in the Global Manager's topology.

  For example, *device1.mydomain.com* is classified as Uncertified in one underlying domain and classified as a Router in a second underlying domain. The Global Manager classifies the device as a Router in its topology. All incoming notifications related to *device1.mydomain.com* are associated with the Router instance in the Global Manager's topology.

- A device has the same name but is an instance of different classes in two or more underlying domains. In this case, the device is not classified as Host, Uncertified, or Node in any underlying domain. The Global Manager classifies the device according to the first topology information it receives from an underlying domain. Relationship information for the device is updated during consecutive topology synchronizations. Incoming event information for the device is consolidated to the instance in the Global Manager's topology.

  For example, *device2.mydomain.com* is classified as a Probe in one underlying domain and a Router in a second underlying domain. If the first underlying domain synchronizes first with the Global Manager, the Global Manager classifies the device as a Probe in its topology. However, any relationship information for the router from the second underlying domain is preserved by the Global Manager and added to the Probe instance. All incoming notifications related to *device2.mydomain.com* are associated with the Probe instance in the Global Manager's topology.

Same Device Named Differently in Separate Underlying Domains

The Global Manager creates its topology based on the names of the devices it imports from the underlying domains. When the Global Manager imports the same device from two or more underlying domains with the same name, it creates a single corresponding device in its own topology. The Global Manager associates any incoming events from the underlying domains that are related to this device with the single device in its own topology.

When the Global Manager imports the same device from two or more underlying domains and each domain gives the device a different name, the Global Manager creates unique elements in its topology for each device.

For example, one underlying domain discovers a device and gives it the name *device3*. A second underlying domain discovers the same device but gives it the name *device3.mydomain.com*. When the Global Manager receives topology information from these underlying domains, it creates two instances in its topology—one named *device3* and one named *device3.smarts.com*.

For more information about InCharge discovery and the convention InCharge applications use to name devices, see the *InCharge IP Discovery Guide*.

Organizing Topology with Groups

Grouping provides a method by which you can organize topology elements. With Service Assurance, you can create groups and organize topology elements to help you more efficiently manage large numbers of elements.

Before you start, you may find it useful to devise a strategy around which you organize topology elements into groups. Common strategies include organizing by:

- Business units
- Geographical regions
- Resources

General Properties of Groups

A group is a user-defined collection of instances from the Global Manager's topology. A group consists of members or child groups. A *member* is a topological element such as a switch. A *child group* (or subgroup) is another group, which may be a collection of members or additional subgroups. A group that contains child groups is referred to as the *parent* group.

Parent and child groups are organized into a tree structure. At the root of each tree is a top-level group. Each top-level group is a distinct organization of groups and members—its configuration or removal does not effect other top-level groups. In the Map Console, top-level groups are displayed directly beneath the icon for the Global Manager.

- A member is also a member of the groups above it within the same group hierarchy.
- Within a single group hierarchy, an element cannot be a member of more than one group. An element can belong to two or more groups that descend from different top-level groups.
- You cannot create a circular group where a group is specified as a subgroup of itself or one of its subgroups.

Before we explain how to create groups, you may find it helpful to understand how groups are displayed in the Global Console.

How Groups Are Displayed in the Map Console

The Map Console only displays the members of a group when that group does not contain any child groups. If a group contains one or more child groups as well as members, only the child groups are displayed.

When you create groups to display them in the console maps, SMARTS recommends that you create a "catch-all" group within a parent group. The "catch-all" group should contain all the members that do not belong to any of the other subgroups. This technique prevents a parent group from containing both members and child groups.

Types of Groups

Service Assurance Manager supports two types of groups: *selective groups* and *hierarchical groups*. A selective group is a group whose members are determined by a matching pattern that you specify through the Global Console. Hierarchical groups are specified in a data file which is then imported by the Global Manager.

In addition to the methods by which they are created, there exist several other differences between selective and hierarchical groups. These differences are described in the following section.

Properties of Selective Groups

Selective groups contain three properties that distinguish them from hierarchical groups: matching criteria, priority, and target classes. These properties help to determine what elements become members of a selective group.

Matching Criteria

Matching criteria are attributes defined in the ICIM model that you use to determine what elements are eligible to become a member of a group. When you create a selective group, you specify a matching pattern that is compared against the attributes of the element. If the pattern matches the value of the specified attribute, the element becomes a member of the group. A matching pattern is comprised of one or more characters and wildcards. If you do not specify a matching pattern, all managed elements that pass the target class filter match the group; priority will determine if any elements become members.

Table 15 lists attributes against which you can apply a matching pattern. The group's target class determines which attributes are available to match against. For example, if the target class is IPNetwork, attributes that describe a managed system, such as Certification, are not listed.

ATTRIBUTE	DESCRIPTIONS
Certification	Level of certification assigned to this device during discovery. Possible values include: UNCERTIFIED, GENERIC, TEMPLATE, CERTIFIED, or VALIDATED.
CreationClassName	Name of the class of which the managed element is an instance. This is used as the ClassDisplayName attribute in notifications affecting this element.
Description	A brief description of the element.
DisplayClassName	Same as creation class name.
DisplayName	Name of the managed element. For systems, DisplayName and Name are usually the same.
IsManaged	Determines if the system is monitored by Global Manager. Note that unmanaged elements do not appear in the Global Manager topology. Value is TRUE or FALSE.

ATTRIBUTE	DESCRIPTIONS
Location	A brief textual description of the system's physical location.
Model	Vendor's name for the system.
Name	Name of the managed element. For systems, Name and DisplayName are usually the same.
PrimaryOwnerContact	Information on how to contact the system's owner.
PrimaryOwnerName	Name of the system's owner.
ServiceName	Name of external system used to import attributes and events.
SystemName	Name of the system that contains this managed element.
Type	Classifies the type of system. Possible values include: Bridge, Host, Hub, Node, Other, Probe, Router, RSFC, RSM, Switch, and TerminalServer.
Vendor	Name of the system's manufacturer.

Table 15: Attributes for Matching Criteria

For information regarding the wildcards you can use to build a matching pattern, see *Wildcard Patterns* on page 151.

Priority

Priority distinguishes between groups at the same level of the hierarchy with the same parent. Each such group is automatically assigned a different priority. When a topology element matches the pattern of two different groups, it becomes a member of the group with the higher priority. Because of this, you should assign a higher priority to a group with a stricter matching pattern. If a group has a high priority and it matches all the topology elements, it will contain all the available members.

Target Classes

A target class acts like a filter, allowing only those elements that are instances of the target class, or one of its subclasses, to become members of the group. Managed elements must pass the target class filter before they are compared against the matching criteria.

When you create a child group, the child group should have the same target class as its parent. The exception to this rule is when the new target class is a subclass of the parent group's target class. For example, the ICIM_ManagedSystemElement class is near the top of the ICIM hierarchy. All of the other target classes are a subclass of ICIM_ManagedSystemElement. If ICIM_ManagedSystemElement is the target class of a parent group, you can select a different target class for a child group.

Similarly, VLAN, NetworkConnection, and IPNetwork are subclasses of ICIM_LogicalLink. If ICIM_LogicalLink is the target class of the parent, you can choose one of these three classes as the target class for a child group.

Table 16 lists the target classes you can assign to a selective group when you create it. Classes are listed in the order that they appear in the ICIM class hierarchy. The description indicates when a class is a subclass of another target class.

TARGETCLASS	DESCRIPTION
ICIM_ManagedElement	Base class for the ICIM system element hierarchy. This is the broadest target class.
UnitaryComputerSystem	Represents a single computer system. This is the superclass for the Bridge, Host, Hub, Probe, MSFC, Router, RSFC, RSM, Switch, TerminalServer, and Node classes.
ApplicationService	Represents service provided by software. Examples include e-mail, Web server, and database applications.
LogicalLink	Represents a link between two endpoints. Examples include database transactions, IP networks, and cables.
VLAN	Virtual LAN typical in switched networks. VLAN is a subclass of LogicalLink.
NetworkConnection	A connection between two routers, typically a virtual circuit. NetworkConnection is a subclass of LogicalLink.
IPNetwork	Subnet of an IP network. IPNetwork is a subclass of LogicalLink.
ServiceSubscriber	Customer who receives services provided through a service offering.
ServiceOffering	A service provided to customers.

Table 16: Target Classes for Selective Groups

Creating Selective Groups

You create and edit selective groups using the Global Console. Creating or editing groups through the console requires administrator privileges from the Global Manager. Operators with monitoring privileges can view groups in the Topology Browser and Map Console but are not able to create or edit groups.

You should be aware of the following points when creating selective groups:

- Top-level selective groups cannot contain members, only child groups. As such, you need to create both a top-level group and one or more child groups to assign topology elements to a group.
- By default, each top-level group contains all the topology elements that match its target class and matching criteria.
- Each group is identified by a unique name. The name is displayed in group maps and in the Topology Browser. It is the value of the DisplayName attribute for the group instance.
- Hierarchical groups are visible in the Group Definition window but cannot be edited.

Layout of the Group Definition Window

You create and edit groups through the Group Definition window. You open this window by selecting **Groups** from the *Configure* menu of the Global Console.

The Group Definition window is divided into two panels. The left panel displays the Global Manager, groups, and the group members. When you choose a selective group in the left panel, the Properties, Priorities, and Matching Criteria tabs display in the right panel of the Group Definition window.

Note: When you select a top-level group, only the Properties tab is displayed. Top-level selective groups do not have priority or matching criteria.

Priority and matching criteria determine which topology elements are members of each group. If you have worked with configuration groups (Polling Groups and Threshold Groups) for InCharge applications, the priority and matching criteria for topology groups function similarly.

For more information about how priority and matching criteria, see *Properties of Selective Groups* on page 68.

The toolbar of the Group Definition window contains four buttons:

- **Delete** removes the specified group from the topology of the Global Manager. If the deleted group contains child groups, the child groups are also deleted.

Note: The **Delete** command does not remove hierarchical groups.

- **Regroup** tells the Global Manager to rebuild the selective groups *from the selected group down to the bottom of the group hierarchy.* If you select the Global Manager icon and invoke Regroup, the Global Manager rebuilds all of the selective groups. This, however, is not usually necessary. It is more efficient to regroup a section of the group hierarchy when there are large number of topology elements in each group.

  You need to invoke **Regroup** after you make changes to the priority or matching criteria of a group. The Global Manager automatically regroups the topology when it synchronizes its topology with the underlying domains.

Note: The **Regroup** command does not affect hierarchical groups.

- **Create Top Level Group** displays the New Group dialog where you specify the name, description, and target class for the group. This command is only available when the Global Manager is selected.
- **Create New Group** displays the New Group dialog where you specify the name, description, and target class for a child group. This command is available when a group is selected in the group tree hierarchy in the left panel of the Group Definition window.

You can also find these commands under the Group menu. In addition, the Group menu also contains the **Save Groups** command. When you invoke **Save Groups**, the Global Manager saves its in-memory database to the repository file.

Method for Creating Selective Groups

To create a selective group, use the Create Top Level Group or Create New Group command.

1. Select *Groups* from the *Configure* menu of the Global Console. This displays the Group Definition window.

   Note that this requires administrator privileges. The *Configure* menu is not displayed for users with monitoring privileges.

2. To create a top-level group, select the Global Manager icon and click the **Create Top Level Group** toolbar button. Alternatively, right-click on the Global Manager and select **Create Top Level Group** from the pop-up menu.

   To create a child group, select the parent group and click the **Create New Group** toolbar button. Alternatively, right-click on the parent group and select **Create New Group** from the pop-up menu.

   This displays the New Group dialog.

3. Specify a name, description, and target class for the group. After you finish specifying values for these three fields, click **OK**.

   The new group displays in the Group Definition window. You can also see the group as an instance of the SelectiveGroup class in the Topology Browser Console or as a group in the Groups tab of the Map Console.

4. Change the priority of the new group. If there are other groups at the same level of the group hierarchy, they are listed under the Priorities tab. By default, a new group is assigned the lowest priority.

   To change a group's priority, select the group whose priority you wish to change. Under the Priorities tab, click the up arrow to give the group a higher priority or click the down arrow to give the group a lower priority.

5. Click the Matching Criteria tab and specify a matching pattern for the group. By default, a new group does not contain matching criteria, meaning it matches all possible elements.

   For example, if you want to create a group that includes systems from a certain geographical area then you might match against the value of the Type and Location attributes. First, specify a pattern that matches the type of systems you want to group. To create a group of routers you would add Type as a matching criteria attribute and specify a matching pattern of Router. Next, specify a pattern that matches against the geographical locale specified in the system's Location attribute. To match against routers in New York, you could add Location as a matching criteria attribute and specify a pattern of "*NY*|*New York*". This pattern would match against "NY" and "New York" anywhere in the Location attribute. *Table 15* lists the classes whose values you can match against.

6. Select the parent for the newly created group, or the Global Manager, and click the **Regroup** toolbar button. You can also right-click on the parent and select **Regroup** from the pop-up menu.

Note: To create a "catch-all" group, create a group with no matching criteria and assign it the lowest priority.

Editing the Properties of a Selective Group

You can edit the description, target class, priority, and matching criteria of a selective group. Similar to creating a group, you must attach to Global Manager with administrative privileges and select *Groups* from the *Configure* menu.

You can edit all the properties of a group before applying the changes. However, if you select another group before clicking Apply, your changes are not applied. In this case, the console displays a dialog window asking if you want to abandon the changes that have not been applied.

1. Select the group whose properties you wish to edit. The properties of the group are displayed in the right panel of the Group Definition window.
2. The description and target class of the group are displayed under the Properties tab.
   - To change the target class, select a class from the pop-up menu.
   - To change the description, edit the text in the Description text box.
3. Select the Priority tab to change the priority of the group.
4. Select the Matching Criteria tab to change the matching pattern of the group.
5. Click **Apply**. If you change the target class, priority, or matching criteria, you also need to click **Regroup**.

Creating Hierarchical Groups

This section describes how to create hierarchical groups, including the syntax of the group data file and how to load the group information into the Global Manager.

To create hierarchical groups:

1. Create one or more hierarchical group data files that specify the groups and their members. You can also add new groups and their members to existing hierarchical group data files, including the two example group

data files (*topology-group.data.template* and *service.data.template*) that are provided with the system. To use one of the example group data files, make a copy of the example file and rename it appropriately.

These data files must be located in the ***BASEDIR***/*smarts/local/conf/ics* directory. For the syntax of the group data files, refer to *Syntax of Hierarchical Group Data Files* on page 76.

2. If necessary, edit the BusinessSection listing of the *ics.conf* file to add or delete the names of hierarchical group data files. Refer to *Syntax of the Group Data File in ics.conf* on page 77.

Note: The two example group data files provided with the system are initially listed in the *ics.conf* file.

3. If hierarchical group files were added or deleted from the *ics.conf* file, reconfigure the Global Manager. Refer to *Reconfiguring the Global Manager* on page 77.
4. Invoke importing of the group definitions by regrouping the group data files. Refer to *Regrouping Hierarchical Group Data* on page 78.

Administrative privileges for the Global Manager are required to reconfigure the Global Manager and regroup the group data files.

The Global Manager processes the group and service topology data specified in the BusinessSection. If the Global Manager does not encounter any errors, it adds any new group and service information to its topology, and deletes any group and service information that is in the current topology but which has been deleted in the new hierarchical group data files being imported.

Note: Both reconfiguring and regrouping update the service topology, if any. For more information about importing service topology, see the *InCharge Service Assurance Manager User's Guide for Business Impact Manager*.

After the hierarchical group data is loaded into the Global Manager, you can view hierarchical groups through the Global Console.

Syntax of Hierarchical Group Data Files

The syntax for group data files provide a means for specifying groups, children (subgroups), and members. The following example illustrates the syntax of a hierarchical group data file.

```
HierarchicalGroup NewYork children NY-Routers "NY Customers"
HierarchicalGroup NewYork children Queens Nassau
HierarchicalGroup NewYork children Bronx
HierarchicalGroup NewYork members Router::nyc1
HierarchicalGroup NY-Routers members Router::nyc1
HierarchicalGroup "NY Customers" members▼
file:/opt/dev/incharge-sa/smarts/local/conf/ics/ny-
▲customers.members
```

This example illustrates the following syntactic rules of hierarchical group data files:

- Each line that specifies a hierarchical group must start with the keyword *HierarchicalGroup*.
- Elements of a hierarchical group must be separated by one or more spaces.
- The name of the group follows the keyword *HierarchicalGroup*. The name of the group must be unique for all existing groups. If the same group name is used in multiple lines, each line is referring to the same instance. In the example above, all four lines starting with "HierarchicalGroup NewYork" are defining either children or members of the NewYork group.
- If the name of a group contains a space, it must be enclosed in double quotes.
- The keyword *children* indicates that this line specifies subgroups of the named group. For example, NY-Routers and "NY Customers" are child groups, or subgroups, of the group named NewYork.
- The keyword *members* indicates that this line specifies members of the named group. For example, the router nyc1 is a member of the group named NY-Routers. You must specify the class name and the instance name of the topology element, separating them with a double colon (::).
- You cannot specify child groups and members in the same line. You can, however, use multiple lines to specify members or children for the same group.

- You can specify a list of members in a member file. For example, the file *ny-customers.members* lists members of the "NY Customers" group. However, you must specify the full path to the file, using the correct syntax for the host operating system.

The following example shows the syntax of a file that lists the members of a group. Because the hierarchical group data file specifies the name of the group, you only need to specify the *<class>*::*<instance>* pairs, one per line, for each member.

```
Router::nycbrd1
Router::nycbrd3
Switch::nycs2
```

Syntax of the Group Data File in ics.conf

The name of the file or files that define the hierarchical group data must be specified in the *ics.conf* file, located in the ***BASEDIR****/smarts/local/conf/ics* directory. The following example shows the syntax of the BusinessSection of the *ics.conf* file.

```
BusinessSection
{
    Name = "topology-group.data.template";
    Name = "service.data.template";
}
```

The BusinessSection is used to specify the data files for hierarchical groups and service topology. You can use any number of files by specifying additional "Name" lines, as shown in the following example.

```
BusinessSection
{
    Name = "new-york-group.data";
    Name = "albany-group.data";
    Name = "san-francisco-group.data";
}
```

Reconfiguring the Global Manager

To reconfigure the Global Manager, see *Reconfiguring the Global Manager* on page 52.

Regrouping Hierarchical Group Data

To regroup the hierarchical group and service topology data files, invoke the following command from the ***BASEDIR****/smarts/bin* directory:

```
▼ % dmctl -s <global_manager> invoke GA_DaemonDriver::
  ICS-Group-Driver start ▲
```

Note: The command must be typed as one line.

Depending on your security configuration, you may be prompted for your InCharge user name and password.

When the Global Manager regroups the topology information in the data files, it first checks for errors. If the Global Manager encounters an error, it writes output to the terminal or its log file and does not process the data files.

Custom Icons for Map Nodes

An administrator with appropriate privileges can assign/unassign a custom icon to a class of elements or to individual elements. A custom icon can be used to customize maps for your organization and to meet the needs of your operators.

The Global Console includes a default set of icons for all ICIM elements. Icons can be associated with classes or instances.

If an icon is not associated specifically with a class, the icon for the parent class is used; this is the *derived* icon. In addition, the icon for a given class could be associated multiple levels up in the class hierarchy.

The set of icons shipped with the console does not include one icon for each class. The derived icon (from the parent class) is used for those classes that do not have a specific icon assigned to them.

You can use two methods to assign icons to class or instances:

- Use the *Edit Map Icons* operation in either the Global Manager Administration Console or the Map Console.
- Using the **sm_config** utility and specifying the name of the map icon specification file. For more information about using **sm_config** to assign map icons, see *XML Reference* on page 155.

Assigning Icons to Classes or Instances

Before assigning images to map nodes, you must create the images and save them to the proper directory on the host where the Global Manager is running. Note that the icons included in the console are not loaded over the network.

The requirements for custom images are:

- Images must be in GIF format
- Images must be 56 pixels square

After you obtain the custom images, copy them to the ***BASEDIR****/smarts/local/images/icons* directory. Images located in this directory are automatically displayed in the icon configuration dialog. The file name of the image, minus any file extension, is used to identify the image.

If an image file located on the Global Manager has the same name as an icon included with the console, the image located on the Global Manager takes precedence.

Assigning an Icon to a class

To assign an icon to a class, perform the following steps:

1. Select *Configure > Edit Map Icons* operation in the Global Manager Administration Console. This displays the Select Icons for: <Global Manager> dialog.
2. Select the class in the Classes list box.

   The Classes list box displays the classes of managed elements. By default, all classes are assigned an icon, however, it may be an icon associated with a parent class. Certain classes are assigned the same icon as their parent class. For example, the Redundancy Group icon is used for the System Redundancy Group, Network Connection Redundancy Group, and Network Adapter Redundancy Group. You can determine what icon is used for the System Redundancy Group by selecting this class in the Classes list and seeing what is displayed as the derived icon. The name of the class to which the derived icon is assigned is also displayed.
3. Click *Use the specific icon*.
4. Scroll list at the bottom of the dialog to choose an icon.
5. Click **Apply**.

6 When you are finished, click **OK**.

Assigning an Icon to an Instance

You can assign an icon to an instance without changing the class icon. To assign an icon to an instance, perform the following steps:

1. Select *Configure > Edit Map Icons* operation in the Global Manager Administration Console. This displays the Select Icons for: <Global Manager> dialog.
2. Select the appropriate class in the Classes list box.
3. Enable the Instances list box (check the *Select Instances* check box). The Instances list box displays the instances of the selected class in the topology.
4. Select the instance in the Instances list box.
5. Click *Use the specific icon*.
6. Scroll list at the bottom of the dialog to choose an icon.
7. Click **Apply**.
8. When you are finished, click **OK**.

Reassigning the Derived Icon to a Class

To reassign the derived icon to a class, perform the following steps:

1. Select *Configure > Edit Map Icons* operation in the Global Manager Administration Console. This displays the Select Icons for: <Global Manager> dialog.
2. Select the class in the Classes list.
3. Click *Use the derived icon.*
4. Click **Apply**.
5. When you are finished, click **OK**.

Reassigning the Derived Icon to an Instance

To reassign the derived icon to an instance, perform the following steps:

1. Select *Configure > Edit Map Icons* operation in the Global Manager Administration Console. This displays the Select Icons for: <Global Manager> dialog.

2. Select the class in the Classes list.
3. Enable the Instances list box (check the Select Instances check box) and then select the instance.
4. Click *Use the derived icon*.
5. Click **Apply**.
6. When you are finished, click **OK**.

7

Tool Configuration for the Global Manager

This chapter describes the server, client, and automated tools you can create and use with the Global Manager. Tools are programs that can be invoked automatically by the Global Manager or through the Global Console by an operator. Topics include how to create, configure, and invoke tools.

In addition, Service Assurance provides the following tools that are described in separate documents.

- Business Process Tools, see the *InCharge Service Assurance Manager User's Guide for Business Impact Manager*
- Report Manager tools, see the *InCharge Service Assurance Manager User's Guide for Report Manager*
- Remedy tools, see the *InCharge Service Assurance Manager User's Guide for Remedy Adapter*.
- Concord eHealth tools, see the *InCharge Service Assurance Manager User's Guide for Concord eHealth Adapter*l.
- InfoVista tools, see the *InCharge Service Assurance Manager User's Guide for InfoVista Adapter*.

Types of Program Tools

A tool is a script that is typically invoked in response to a notification. Such a response might be to ping an affected device or to open a trouble ticket. Tools can also be invoked on topology elements. Service Assurance supports three types of tools:

- Server tools
- Client tools
- Automated tools

Server and client tools are associated with a user profile and invoked by an operator using the Global Console. Two tool submenus, Server Tools and Client tools, are displayed in a pop-up menu when an operator right-clicks on a notification or a device. Access to server and client tools can be controlled by associating or not associating tools to user profiles.

- The Server Tools menu in the Global Console displays a list of available server tools. Server tools are invoked by the Global Manager and available to any console user, provided the user's profile includes the tools.
- Client tools are invoked on the host where the Global Console is running, which requires that the tool scripts are on the host where the console is running. The Client Tools menu displays a list of available client tools, provided the user's profile includes the tool.
- An automated tool is a type of server tool that is invoked by an escalation policy or an adapter. Automated tools are not visible to users of the Global Console.

How Tools are Invoked

This section describes how to invoke tools and where the output of server and client tools is displayed.

Invoking Server and Client Tools

Server and client tools are invoked through the Global Console. Such tools are always invoked on a particular target object by the console operator. The target object can be a notification or a topology element such as a router. When an operator right-clicks on the target, the pop-up menu lists the available tools.

For information about invoking server and client tools from the Global Console, see the *InCharge Service Assurance Manager Operator's Guide*.

Viewing the Output of a Server or Client Tool

Server and client tools, depending on their purpose, may produce output. For example, the output of a ping tool can tell whether the ping completed successfully.

For client and server tools, you can configure whether a tool displays any output. When a server tool is configured to display its output, the output is echoed to *stdout* and collected by the Global Manager. The Global Manager sends the output back to the console where the tool was invoked.

The output of server and client tools is displayed in the Tool Output window. The Tool Output window opens when the tool is invoked. Output is written to the window while the tool is running. If a tool produces an error, the error message is also displayed in the Tool Output window. If an error occurs for a server tool that is not configured to display output, the Tool Output window displays with the return code of the tool.

Invoking Automated Tools

An automated tool is run when it is triggered based on the settings of an escalation policy. An escalation policy is configured to invoke automatic tools to run on notifications that meet the escalation criteria for specified periods of time. For more information, see *Escalation Configuration for the Global Manager* on page 103.

Alternatively, an automated tool runs with the automatic action adapter on the same host as the Global Manager. Start this adapter by invoking the **sm_adapter** command and supplying a list of arguments. After starting the automatic action adapter, no further intervention is required. The adapter uses a notification list to listen for particular notifications. When it receives a matching notification, it automatically responds by invoking the specified tools through the Global Manager.

Once started, the automatic action adapter runs in the background and responds to notifications that match its notification list. For example, the automatic action adapter, configured with a trouble-ticket system tool could respond to a notification by opening a trouble ticket.

For more information about creating an automated tool, see *Modifying a Tool* on page 100.

Information Recorded to a Notification's Audit Log

When a server or automated tool is invoked on a notification, information about the tool is recorded to the notification's Audit Log. This information includes:

- Date and time when the tool was invoked.
- Name of user that invoked the tool.
- Whether the tool completed successfully.
- Name of the tool.

Security Considerations for Tools

If a tool invokes a SMARTS utility, such as dmctl, or otherwise initiates a connection to the Global Manager, you must configure security to enable that connection. An essential piece of information is knowing the user name under which the tool is invoked.

- Client tools are invoked under the user name of the operator who started the Global Console.
- Server tools are invoked under the user name that the Global Manager is running under. Remember, server tools are invoked on the same host as the Global Manager.
- Automated tools are invoked under the user name that started the **sm_adapter** command or by the name of the Global Manager.

If a client tool needs to connect to the Global Manager and perform an action that requires administrative privileges, you have two choices:

- Provide the operator with administrative privileges.
- Create a server tools that performs the action.

Because server tools run under the user name that invoked the Global Manager process, this user must have an authentication record in the *clientConnect.conf* and *serverConnect.conf* files on the host where the tool is to be executed. You must configure the authentication records for this account so that prompting is not required. Prompting should not be used for automated tools.

Sample Tool Scripts

SMARTS provides a number of sample tool scripts that, with minor modifications, work on most systems. The main purpose of these scripts, however, is to provide examples that you can examine when developing your own tools.

Note: The ping and Telnet tools are *sample* scripts. You need to customize them to meet the needs of your environment. For example, the sample scripts do not take firewalls into account.

Table 17 lists the sample server and automated tools, the display name for the tool as seen by users in the Global Console, and provides a brief description. Server and automated tools are located in the ***BASEDIR****/smarts/actions/server* directory.

To modify one of the sample tool scripts, open it with the **sm_edit** utility to create a local copy of the script.

SERVER TOOL SCRIPT	DISPLAY NAME FOR TOOL	DESCRIPTION
`ics-closetkt`	Sample - Close Trouble Ticket	This tool inserts a static value into the field of a notification. The context and Status Criteria specify that this tool is only active for notifications with the text OPEN in the TroubleTicketID field.
`ics-opentkt`	Sample - Open Trouble Ticket	This tool inserts a static value into the field of a notification. The context and Status Criteria specify that this tool is only active for notifications without the text OPEN in the TroubleTicketID field.
`ics-ping-interface`	Sample - Ping-Interface	This tool pings an IP interface. The Context Criteria specify that this tool is active for notifications with a value of “Interface” for the ClassName attribute.

SERVER TOOL SCRIPT	DISPLAY NAME FOR TOOL	DESCRIPTION
`ics-ping-IP`	Sample - Ping-IP	This tool pings an IP interface. The Context Criteria specify that this tool is active for notifications with a value of "IP" for the ClassName attribute.
`ics-ping-all`	Sample - Ping-all	This tool pings all the IP interfaces associated with a device. The Context Criteria specify that this tool is active when the target is a UnitaryComputerSystem. As such, valid targets include both notifications and instances. When invoked, this tool retrieves all of the IP addresses associated with the target instance from one or more underlying domains. Two variables that control the behavior of the tool: • PING_ALL_DOMAINS controls whether the tool retrieves IP addresses from each underlying domain that manages the instance. When set to 0 (zero), the default, the tool retrieves IP addresses from a single domain. When set to 1 (one), the tool retrieves IP addresses from all domains. • PING_ONCE controls whether the tool pings each retrieved IP address or whether it stops after the first successful ping. When set to 0 (zero), the tool stops after the first successful ping. When set to 1 (one), the default, the tool pings all available IP addresses.

SERVER TOOL SCRIPT	DISPLAY NAME FOR TOOL	DESCRIPTION
`ics-ping-device`	Sample - Ping-device	This tool pings the IP address of the SNMP agent for the affected element. The Context Criteria specify that this tool is active when the target is a UnitaryComputerSystem. As such, valid targets include both notifications and instances. When invoked, this tool retrieves the IP address of the SNMP agent associated with the target instance from one or more underlying domains. The tools script includes one variable that controls the behavior of the tool: • PING_ALL_DOMAINS controls whether the tool retrieves IP addresses from each underlying domain that manages the instance. When set to 0 (zero), the default, the tool retrieves IP addresses from a single domain. When set to 1 (one), the tool retrieves IP addresses from all domains.
`ics-telnet`	Sample - Telnet	This tool opens a telnet session with the affected device. The *.sh* version of this script pings an IP address of the affected device before attempting to telnet. If the ping fails, the script pings another IP address on the device, and so on until a ping succeeds or all the pings fail. When a ping succeeds, the script telnets to the IP address where the ping succeeded. If this tool is invoked from a console running over X Windows, the script opens a separate window on the user's display to invoke the telnet session.

Table 17: Sample Server Tool Scripts

Table 18 lists the sample client tools and provides a brief description of each. Client tools are located in the ***BASEDIR****/smarts/actions/client* directory.

CLIENT TOOL	DESCRIPTION
`SmGetEnv`	This tool parses the environment variables passed by the Global Manager to the tool script. It also prints all environment variables to the Tool Output window. A version of this tool script is also provided in Perl.
`SmLaunchPerlScript`	This tool launches the *SmGetEnv.pl* tool script to illustrate how you can launch Perl scripts from a tool. You must edit this script to specify the location where Service Assurance is installed.
`browser`	This tool opens a Web browser. You must edit this script to specify the location where your browser is installed.
`ciscoworks`	This tool opens a Web browser that loads the administration page for CiscoWorks. You must edit this script to specify the location of your browser and the host where CiscoWorks is running.
`pinger`	This tool pings the name of the specified element.
`Reporting`	This actions accesses reports available through the InCharge Service Assurance Report Manager. It opens a Web browser with the URL set to the location of the Crystal Enterprise EportFolio application. You must edit this script to specify both the Web browser and the URL.

Table 18: Sample Client Tools Scripts

Remember that tool scripts are system-specific. For UNIX systems, a tool is typically invoked by a shell script (*/bin/sh*) while on Windows systems, a tool is typically invoked by the Windows command interpreter (cmd.exe). You can also write a tool script in a language such as Perl, which runs on different platforms with little or no modification.

Creating Tools

This section provides the information necessary to create and configure a tool script.

Follow these steps to develop and use a tool script:

1. Determine the action to be performed by the tool and write the script.
2. Save the script to the proper location.
3. Test the tool script to ensure that it executes properly.

4. Assign the proper read/execute permissions to the tool script.
5. Configure the tool to invoke the script. In addition, client and server tools must be associated with the appropriate user profile. For automated tools, you must create an adapter or escalation policy to invoke the tool.

You can configure tools using one of the following tools:

- Global Manager Administration Console, as described in this section.
- **sm_config** utility, as described in *Importing and Exporting Configurations* on page 131.

How Data is Passed to a Tool Script

When a tool is invoked, the attributes of the tool target are automatically passed to the tool script. The attributes are passed to the tool script as environment variables in the form:

```
SM_OBJ_<NAME>=<value>
```

The `<NAME>` parameter identifies the attribute and the `<value>` parameter contains the attribute's value. For example, the ClassName attribute would be passed in the environment variable:

```
SM_OBJ_CLASSNAME=Router
```

Your tool script must parse these environment variables and extract the information required by the program invoked by the tool script. For example, if you want a tool that telnets to a device, you need to extract the environment variable that contains the name or IP address of the device, as shown in the example code (in Windows CMD script):

```
rem if SM_OBJ_Name isn't set in environment report an error
if not defined SM_OBJ_Name (
   echo Invalid SM_OBJ_NAME
   exit 0
)

rem Use the value of SM_OBJ_Name to connect to the device
start /wait cmd /c "telnet %SM_OBJ_Name% "
```

The tool target can either be a notification or an instance. For both types of targets, a tool script receives the attributes listed in Table 19 as environment variables.

ENVIRONMENT VARIABLE	DESCRIPTION
SM_REMOTE_USER_NAME	Specifies the InCharge user name of the person that invoked the tool.
SM_SERVER_NAME	Specifies the name of the Global Manager that executes the tool script.

Table 19: Attributes Passed To All Tool Scripts

Table 20 lists the attributes that are passed to the tool script when the target is an instance.

ELEMENT TARGET ATTRIBUTES	DESCRIPTION
SM_OBJ_CLASS_NAME	Class name of the system. For example, Router.
SM_OBJ_INSTANCE_NAME	Name of the system.
SM_OBJ_DOMAIN_NAME	Name of the Global Manager.

Table 20: Attributes for Element Targets

Table 21 lists the attributes that are passed to the tool script when the target is a notification.

NOTIFICATION TARGET ATTRIBUTES	DESCRIPTION
SM_OBJ_Acknowledged	Specifies whether the notification is acknowledged. TRUE if the notification has been acknowledged, FALSE if not.
SM_OBJ_Active	Specifies whether the notification is active. TRUE if the notification is active, FALSE if not.
SM_OBJ_Category	Type of notification sent by the Global Manager. Possible values include: • BackplaneUtilization • Error • Performance • PowerSupply • Resource • SystemConnectivity • Temperature • VLANConnectivity
SM_OBJ_Certainty	Confidence that this notification is the correct diagnosis. Value ranges from 0 to 100

NOTIFICATION TARGET ATTRIBUTES	DESCRIPTION
SM_OBJ_ClassDisplayName	Name of the ClassName that is displayed to the user.
SM_OBJ_ClassName	Name of the class where the event occurred. May not be the same as SM_OBJ_ClassDisplayName.
SM_OBJ_ElementClassName	Class name of the managed element most closely related to this event.
SM_OBJ_ElementName	Name of the managed element most closely related to this event.
SM_OBJ_EventDisplayName	Name of the EventName that is displayed to the user.
SM_OBJ_EventName	Name of the event. May not be the same as SM_OBJ_EventDisplayName.
SM_OBJ_EventText	A description of the notification.
SM_OBJ_EventType	MOMENTARY when the notification has no duration. DURABLE if the notification has a period for which it is active, such as a link failure.
SM_EventState	State of the event. Possible values include: • ACTIVE • WAS_ACTIVE • SUSPENDED • INACTIVE The UNITIALIZED state is not relevant here.
SM_OBJ_FirstNotifiedAt	Time, in seconds, when the notification first became active.
SM_OBJ_Impact	Numeric value that indicates the effect of this event on related elements.
SM_OBJ_InMaintenance	TRUE if the device is in maintenance mode, FALSE if not.
SM_OBJ_InstanceDisplayName	Name of the InstanceName that is displayed to the user.
SM_OBJ_InstanceName	Name of the instance where the event occurred. May not be the same as SM_OBJ_InstanceDisplayName.
SM_OBJ_IsRoot	TRUE if the notification is a root cause, FALSE if not.
SM_OBJ_LastChangedAt	Time, in seconds, when the status of the notification last changed.

NOTIFICATION TARGET ATTRIBUTES	DESCRIPTION
SM_OBJ_LastClearedAt	Time, in seconds, when the notification was last cleared.
SM_OBJ_LastNotifiedAt	Time, in seconds, when the notification was last notified
SM_OBJ_Name	InternalEventHandle for the notification.
SM_OBJ_OccurrenceCount	Number of times the notification has occurred.
SM_OBJ_Owner	Name of the person responsible for this notification. Value is SYSTEM when acknowledged by the Global Manager.
SM_OBJ_Severity	Level of severity for this notification. 1 = CRITICAL 2 = MAJOR 3 = MINOR 4 = UNKNOWN 5 = NORMAL
SM_OBJ_SourceDomainName	Name of the underlying domain that sent this notification. If more than one domain is listed, the names are separated by commas.
SM_OBJ_TroubleTicketID	Trouble-ticket number associated with this notification.
SM_OBJ_UserDefined1-10	Ten user-defined fields.You can populate these fields with data with an ASL hook script.

Table 21: Attributes for Notification Targets

Table 22 lists additional attributes that are passed to the tool script when the tool is invoked through an escalation policy.

ELEMENT TARGET ATTRIBUTES	DESCRIPTION
SM_POBJ_POLICY	Name of the policy that executed the tool.
SM_POBJ_PATH	Name of the path that executed the tool.
SM_POBJ_LEVEL	Name of the level that executed the tool.

Table 22: Attributes for Escalation Targets

In addition to the attributes listed above, a server tool script receives the DISPLAY environment variable. The value of this variable is used to determine the location of the user's X Window System display. For more information about invoking server tools through an X Windows System, see *Running Tools Over X Windows* on page 102.

Where to Save Tool Scripts

After you write a script, you need to save it to the proper location and ensure that it is executable. A server tool, which includes automated tools, must be located in the ***BASEDIR****/smarts/local/actions/server* directory on the host where the Global Manager is running. The Global Manager must be able to execute the script.

A client script must be located on the host where the Global Console is installed. The proper location is the ***BASEDIR****/smarts/local/actions/client* directory. The console user must be able to execute the script. Server and client tools are configured in the Global Manager Administration Console.

Configuring a Tool with the Global Manager Administration Console

New tools, or modifications to existing tools, are not available to console users until they restart the Global Console.

To configure a tool using the Global Manager Administration Console:

1. Select the type of tool you need to configure from the *Edit* menu or click the appropriate toolbar button. This displays one of the following wizards:
   - Automated Tool Creation Wizard
   - Client Tool Creation Wizard
   - Server Tool Creation Wizard
2. Type a name for the tool. This name is displayed in the Global Manager Administration Console, and, for server and client tools, in the Global Console.
3. Click the Create New or the Copy Existing button. When you copy an existing tool, the configuration parameters for the tool are the same as the copied tool. Click **Next**.
4. Define the following tool parameters and click **Next**. For more information on these parameters, see *Tool Parameters* on page 96.

- Program to run
- Time-out Interval
- Trace
- Display (client and server tools only)
- User Profiles (client and server tools only)

5. Use the Filter Builder to specify the Context Criteria for a server or client tool and click **Next**. For more information on Context and Status Criteria, see *Context and Status Criteria for Client and Server Tools* on page 97.
6. Use the Filter Builder to specify the Status Criteria for a server or client and click **Next** to advance to the confirmation screen.

   Click **Finish** to create the tool. You will receive an error message or a success message based on the result.

Tool Parameters

Table 23 describes the configuration parameters for tools. Only a subset of these parameters are available for automated tools.

PARAMETER	DESCRIPTION
Name	Name of the tool that is displayed in the console.
Program	Specifies the name of the program that is invoked when the tool runs. Type the program name or select a program from the pop-up menu. • For client tools, this is a list of tools in the ***BASEDIR***/*smarts/actions/client* and ***BASEDIR***/*smarts/local/actions* directories on the host where the Global Manager Administration Console is running. • For server or automatic tools, this is a list of tools in the ***BASEDIR***/*smarts/actions/server* and ***BASEDIR***/*smarts/local/actions/server* directories. This parameter cannot contain path separators. This is a drop-down list of existing server or client scripts. The name cannot contain path separators.
Timeout	Specifies the number of seconds to wait for a tool script to complete. If the tool script does not complete within the specified, the tool process is terminated. The default is 30 seconds.
Trace	Specifies whether trace output, used for debugging, should display in the tool output window when a tool is run.

PARAMETER	DESCRIPTION
DisplayOutput	Specifies whether a tool script should display output to the console. If this parameter is set to true, the console opens the Tool Output window immediately after invoking the tool. If false, the Tool Output window opens only if the tool returns an error. If this parameter is omitted, its defaults to true. This parameter is not applicable to automated tools.
User Profiles	Specifies the user profiles that include this tool. This parameter is not applicable to automated tools.
Context	Determines whether a tool displays in the console's pop-up menu when the user right-clicks on a tool target. A tool always displays when this parameter is omitted. This parameter is not applicable to automated tools.
Status	Determines whether a tool is enabled or disabled in the console's pop-up menu. A tool is always enabled when this parameter is omitted. This parameter is not applicable to automated tools.

Table 23: Server Tool Configuration Parameters

Context and Status Criteria for Client and Server Tools

Context Criteria and Status Criteria determine whether a tool is applicable to the selected target. If an operator selects an instance, then the target is an instance. If an operator selects a notification, the target can be either the notification or the instance where the notification occurred, as identified by the ElementClass and ElementInstance attributes.

Context Criteria determine whether a tool is displayed in the menu when a target is selected. The console checks the Context Criteria when the notification is received and does not re-check until the notification is archived and re-notified. Because of this, you should apply context checking to attributes whose values do not change:

- Category
- ClassDisplayName
- ClassName
- ElementClassName
- ElementName
- EventDisplayName
- EventName
- EventType
- InstanceDisplayName

- InstanceName

Although you can apply context checking to other attributes, the results are based on the initial value of the attribute.

Status Criteria determines whether a tool is active. The name of a disabled tool is gray in the menu and cannot be invoked. You should apply Status Criteria to attributes whose value might change.

- Acknowledged
- Active
- Certainty
- ClearOnAcknowledge
- EventState
- EventText
- Impact
- InMaintenance
- IsRoot
- OccurrenceCount
- Owner
- Priority
- Severity
- SourceDomainName
- TroubleTicketID

The tool target determines what attributes are passed to the tool to be matched against the Context Criteria. For example, when the target is an element, you can match against the attributes specified in Table 20. If the target is a notification, the matching criteria determine which attributes are passed to the Context Criteria. If the matching criteria specify a notification attribute, then the attributes of a notification, from Table 21, are passed. If the matching criteria specify an element, then the attributes of an element are passed.

Creating Context and Status Criteria

You specify Context and Status Criteria using the Filter Builder described in *Working with Filters* on page 121. You can use an expression filter or an ASL filter to specify context and Status Criteria. Unlike the filters you can create for notification lists, you can use either an expression filter or an ASL filter, but not both.

The following example illustrates creating Context Criteria using Filter Builder.

1. Select a class from the Context object drop-down menu. The filter expression is matched against one or more attributes of the selected class.

   For example, select the UnitaryComputerSystem class.

2. Add an attribute to the filter sheet to specify a wildcard expression against an attribute of UnitaryComputerSystem. This enables you to narrow the range of notifications for which the tool is displayed.

   If you do not add an attribute, the Context Criteria matches all instances of UnitaryComputerSystem or its subclasses.

3. Click **Next**. This displays the filter sheet for Status Criteria.

4. Select a class from the Status object drop-down menu. The filter expression is matched against one or more attributes of the selected class.

   For example, to match Routers, select the Router class.

5. Add an attribute to the filter sheet to specify a wildcard expression against an attribute of the Router class.

   If do not add an attribute, the Status Criteria matches all instances of the Router class.

6. For example, select the Vendor attribute and type a value of CISCO. The Status and Context Criteria for this tool specify that it displays in the Server Tools menu for any system element or any notification for which the failing element is a system. If that system is a router with a value of CISCO for its Vendor attribute, the tool is active and can be invoked.

7. Click **Next** to view the confirmation screen or **Finish** to create the tool configuration.

Modifying a Tool

To modify an existing server, client, or automated tool, select the tool in the tree of the Global Manager Administration Console. This displays the Configure Tool panel. Modify the appropriate parameters and click **Apply** to synchronize the changes in the Global Manager.

Disabling a Tool

You can disable server, client, and automated tools. This enables you to prevent the tool from being used. Disabled tools are not display in the Global Console and do not execute when launched through an escalation policy or through **sm_adapter**.

To disable a tool, select the appropriate client, server, or automated tool in the Global Manager Administration Console. On the configuration panel, un-check the box beside Enabled. The name of disabled tool displays in gray in the tree. Click **Apply**.

Changing the Order in Which Tools are Displayed

You can determine the order in which tools are displayed in the Server Tools and Client Tools menus of the Global Console. In the Global Manager Administration Console, select Automatic, Client, or Server under the Tools in the tree. This displays the Prioritize Tools panel. Use the up and down arrows to move a tool up or down in the menu list.

Running Automated Tools with sm_adapter

The configuration options for an automated tool are specified as arguments to the **sm_adapter** program when you invoke it at the command line. Table 24 lists the arguments you should specify when invoking an automated tool.

ARGUMENT	DEFAULT VALUE	DESCRIPTION
`--server=`	INCHARGE-SA	The name of the Global Manager from which the adapter receives its notification list and notifications.
`--subscribe=`	N/A	Name of the notification list received from the Global Manager.
`-DNotifyAction=`	N/A	Name of the tool as specified in the Global Manager Administration Console. This is the tool to invoke when a notification is active. This argument is optional.
`-DClearAction=`	N/A	Name of the tool as specified in the Global Manager Administration Console. This is the tool to invoke when the notification clears. This argument is optional.
`-DLogActions=`	FALSE	Specifies whether the adapter should log each action to *stdout*. This argument is optional.
`--output`	N/A	Redirects the output from the adapter to a log file in ***BASEDIR****/smarts/local/logs*. The `--DLogActions` argument must be true to enable output.
`ics/auto-action.asl`	N/A	This script is required to perform the automated action.

Table 24: Arguments to sm_adapter for Automated Action

The following example shows the arguments for the automatic action adapter that invokes the "Open Trouble Ticket" script when a notification becomes active (is notified) and invokes the "Close Trouble Ticket" tool when the notification becomes inactive (clears).

```
# BASEDIR/smarts/bin/sm_adapter--server=INCHARGE-SA \
                 --subscribe=TicketingNL/n  \
                 -DNotifyAction="Open Trouble Ticket"  \
                 -DClearAction="Close Trouble Ticket"  \
                 -DLogActions=TRUE  \
                 --output  \
                 ics/auto-action.asl
```

Running Tools Over X Windows

If necessary, you can configure a server tool to invoke an X Windows application. The console user that invokes such an action must run an X server in order to display the X window. When the Global Console attaches to the Global Manager, it passes the user's display information to the Global Manager.

The Global Console determines a user's display using the following rules:

- If the environment variable SM_DISPLAY is set, its value is passed to the Global Manager.
- If the environment variable DISPLAY is set, its value is passed to the Global Manager.
- The value is set to ***<host>***:0.0 where <host> is the name of the user's system.

Because X Windows applications are typically long running, you should take special care when writing the tool script to avoid unwanted interactions with the tool time-out specified in the ToolSection of the *ics.conf* file. You should write the tool script so that the X Window application runs in the background. This allows the tool script to exit and the Global Manager to stop the time-out. If the tools script runs in the foreground, the Global Manager will terminate the process when the time-out expires.

The *sm_xcmd* utility is provided by SMARTS for executing an X Windows application in the background. The utility performs a basic connectivity test of the user's X Windows display. The *ics-telnet.sh* server tool is an example of a tool script that uses *sm_xcmd* to run an application over X Windows.

8

Escalation Configuration for the Global Manager

Escalation provides a method for automatically responding to a notification:

- As soon as the notification is received by the Global Manager.
- When the notification satisfies certain conditions.

Automatic escalation can enhance your problem resolution process by:

- Dispatching maintenance staff with a description of hardware failures and the specific parts they will need
- Restoring service by restarting applications, initiating failover processes, or rerouting services around failed components
- Notifying customers of service outages and providing status updates
- Alerting successive layers of management of unresolved problems to ensure that they receive the proper level of attention.

This chapter describes how escalation works and provides step-by-step procedures for configuring escalation.

Overview of Escalation

Escalation helps prevent important notifications from being overlooked or ignored and can immediately start the process of remedying the situation that caused the notification.

For example, you can create an escalation policy that is activated when a specific router goes down. When this notification occurs, the Global Manager matches to the appropriate escalation policy and path. The path then begins invoking tools such as opening a trouble ticket and paging technicians.

A notification continues to escalate while it matches the filter criteria of the path. As time progresses, the notification may escalate to different levels in the path based on the duration, and the path may invoke additional tools to respond to the notification. If the status of the notification changes in such a way so that it no longer matches the path's filter criteria, the notification no longer escalates to the next path level.

For example, if a Router Down notification matches against the path filter for 15 minutes, the notification is escalated to the next level. At this level, the escalation path could invoke a tool that sends e-mail to a manager. But, if a technician acknowledges the notification after 10 minutes, and this acknowledgement causes the notification to no longer match the path filter criteria, the notification does not escalate to the next level.

Escalation Policy Structure

An escalation policy is a set of related escalation paths. An escalation policy includes a policy filter and one or more escalation paths, as shown in Figure 3. Notifications that match the policy filter are passed to the escalation paths. If a notification does not match the policy filter, it is ignored. If a policy filter is not defined, all notifications are passed to the escalation paths.

An Escalation Path also includes a filter and up to six escalation levels. A notification must match both the policy filter and the path filter to begin at the first escalation level, Level 0.

Each escalation level includes one or more automated tools to invoke and a time duration, after which a notification advances to the next level. An escalation level cannot have a duration of less than one minute. You can, if necessary, create an escalation level that does not invoke a tool.

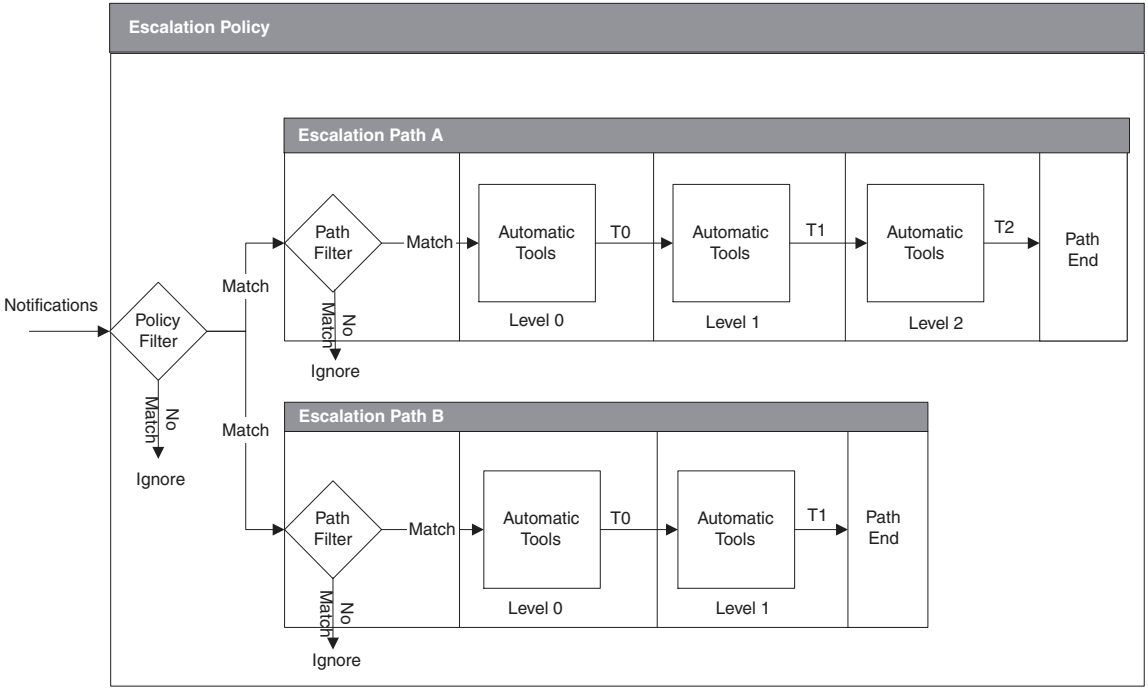

Figure 3: Escalation Policy Structure

When a notification matches both the policy and path filters, it starts at escalation level 0. When a notification reaches level 0, the tools associated with level 0 are invoked immediately. The notification remains at Level 0 for the specified duration, show as T0 in Figure 3, before escalating to Level 1. At this point, all Level 1 tools are invoked and, after the duration of T1, the notification escalates to Level 2. This process repeats through each level defined for the path. If the notification changes during escalation and no longer matches the path and policy filters, it leaves the escalation path without invoking any additional tools.

Viewing an Escalation Policy

You create, configure, and monitor escalation policies with the Global Manager Administration Console. Expanding Escalation Policies in the administration tree lists escalation policies. Selecting a policy in the administration tree displays the Escalation Policy view, as shown in Figure 4.

Policy names in the administration tree are black when any escalation path in the policy is enabled. When all paths in a policy are disabled, the policy name is gray.

The Configure Escalation Policy view allows you to create, modify, and delete escalation policies, paths, and levels. In addition, you can monitor the status of an escalation policy by right-clicking on an escalation path and selecting *Show Notifications* from the menu.

Escalation paths are displayed as horizontal bars. The name on the left of the escalation path is the path name. Within each bar are shaded blocks, each of which represents an escalation level within the path. The duration and automated tools are listed within each escalation level of the path.

An enabled escalation path is displayed with each level a different color. A disabled path is displayed in gray. All the escalation paths shown in Figure 4 are enabled.

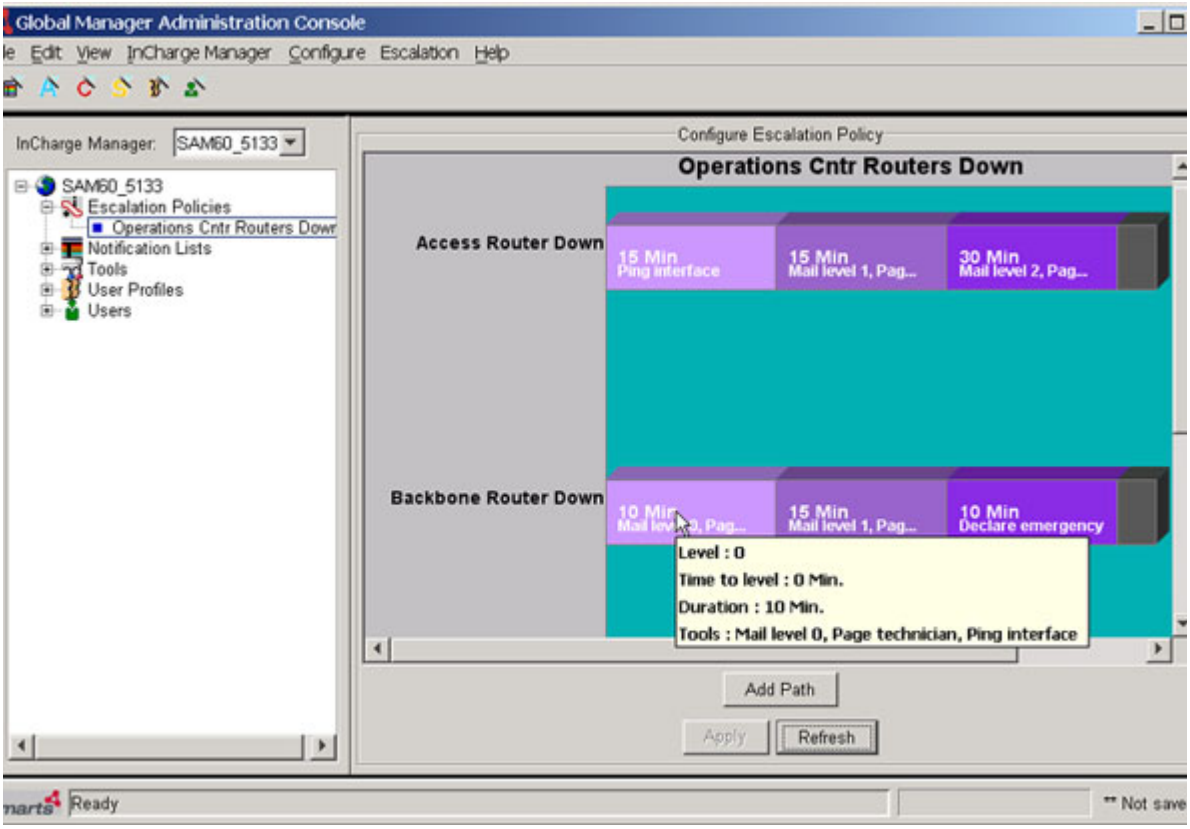

Figure 4: Escalation Policy View

When you hold the cursor over an escalation level, a tooltip window displays the following information:

- Level indicates the level (0–5) within the path.

- Time to level indicates the amount of time it takes a notification to reach this level within the path.
- Duration indicates the amount of time a notification spends at this level before advancing to the next level. However, Duration does not include the time it takes to invoke any tools. When a level is reached, actions are executed first. When all actions have finished executing, the notification is scheduled to enter the next level. The notification will enter the next level after the amount of time set by Duration.
- Tools lists the automated tools that are invoked at this level.

At the bottom of the Configure Escalation Policy window are three buttons:

- **Add Path** adds a new escalation path to the displayed policy. This is equivalent to selecting *Escalation > Path > New*.
- **Apply** saves any changes made to the escalation configuration to the Global Manager. Unapplied changes are not saved if you do not click **Apply** before leaving the Escalation Policy View. If you have unsaved changes, you are prompted to save before leaving the view.
- **Refresh** retrieves the configuration of current escalation policy from the Global Manager. This is important if another administrator might be modifying the policy that you are currently viewing. The Global Manager prevents two administrators from overwriting each other's changes without first viewing the changes. For example, if you make changes to an escalation policy and apply them while another administrator is also making changes, the other administrator will not be able to apply the changes. The Global Administration Manager Console prompts the other administrator to refresh their policy view.

  Refresh can also be used to cancel changes that have not been applied.

How Notifications are Scheduled for Escalation

When the Global Manager receives or creates a notification, it compares the notification against the filter for each policy and path. If the notification matches a policy and path filter, the Global Manager schedules the notification for escalation. When a level is reached, actions are executed first. When all actions have finished executing, the notification is scheduled to enter the next level. The notification will enter the next level after the amount of time set by the duration of the level.

The Last Notify time, or the `LastNotifiedAt` notification attribute, also determines whether a notification enters into an escalation policy. The Last Notify Time must be later than the time at which the path was enabled in order for the notification to escalate.

If a Global Manager is stopped and restarted, the Global Manager restores the state of escalated notifications. Notifications which are scheduled to escalate to a new level before the Global Manager stops are rescheduled based on the current time when the Global Manager is restarted.

The rescheduled escalation time accounts for the time that the Global Manager was not running. A rescheduled notification, however, will never escalate more than one level as a result of the time the Global Manager was not running.

Effect of Global Manager Restart on Escalation

The following examples illustrate the effect of a Global Manager restart on escalating notifications:

Example 1

1. A notification enters escalation level 0 at 1:00 PM. The duration of the level is 1 hour.
2. At 1:10 PM the Global Manager is shut down.
3. At 1:25 PM the Global Manager is restarted.

The notification was scheduled to escalate to level 1 at 2:00 PM. When the Global Manager restarts, this notification is rescheduled to escalate at 2:00 PM.

Example 2

1. A notification enters escalation level 1 at 2:00 PM. The duration of level 1 is 10 minutes and the duration of level 2 is 10 minutes.
2. The Global Manager is shutdown at 2:05 PM and restarted 20 minutes later at 2:25 PM.

Prior to the Global Manager shutting down at 2:05 PM, the notification would have escalated to level 2 at 2:10 PM and was scheduled to escalate to level 3 at 2:20 PM (the duration of level 2 is 10 minutes).

Since the notification was in level 1 when the Global Manager was shutdown, it needs to be rescheduled for the next level (level 2) when the Global Manager is restarted. Therefore, when the Global Manager is restarted, it reschedules the notification to escalate to level 2 immediately at 2:25 PM. Upon entering level 2 and performing actions, the notification is scheduled for level 3 at 2:35 PM.

Developing Escalation Policies

The first step in creating an escalation policy is to formalize your response plans to the notifications you are likely to encounter. You may need to assign responsibilities and determine how quickly your organization is required to respond to various types of system and network failures. These plans form the template for your escalation paths and levels.

You may also need to create automated tools to initiate your response to incoming notifications. For more information about automated tools, see *Tool Configuration for the Global Manager* on page 83.

The process of creating an escalation policy at the Global Manager Administration Console consists of the following steps:

- Create an escalation policy by copying an existing policy or creating a new policy.
- Add one or more escalation paths to the policy. Again, you can copy an existing path or create a new path.
- Add escalation levels, up to six, to each path.

Creating an Escalation Policy

To create an escalation policy, complete the following steps using the Global Manager Administration Console:

1. Select *Escalation* > *Policy* > *New*. This displays the Create a New Escalation Policy dialog.
2. Type the name for the Escalation Policy. This name is displayed in the administration tree and at the top of Configure Escalation Policy window
3. Create the policy by copying an existing policy or by creating a new policy. When you create a new policy, the parameters of the policy,

such as filters and paths, must be created. When you copy a policy, all of the properties of the existing policy are copied to the new policy.

4 Click **Edit Filter** to create or modify the filter that is matched against notifications to determine whether they qualify for the policy. This displays the Filter Builder dialog.

   You can specify an expression filter, an ASL filter, or both. When you combine filters, by adding filter sheets, a notification must match at least one of the filter sheets to pass the filter. For more information about the Filter Builder, see *Working with Filters* on page 121.

5 When you finish creating filters, click **OK** to close the Filtering Policy dialog and click **OK** to close the Create a new Escalation policy dialog.

The next step is add one or more escalation paths.

Creating an Escalation Path

You can create an unlimited number of paths within a policy, but it is usually easier to organize and manage smaller groups of paths in more policies.

1 Select the escalation policy in the administration tree. The Configure Escalation Policy view is displayed.

2 Create an escalation path by adding a new path or copying an existing path.

   - To create a new blank path, select *Escalation > Path > New* or click **Add Path** at the bottom of the Escalation Policy View panel. This displays the Create a new Escalation Path dialog.
   - To copy a path, select the path you wish to copy and select *Escalation > Path > Copy.* This displays the Copy Escalation Path dialog box. You can only copy a path from the current policy.

3 Type the name of the escalation path. If you are copying the path, modify it to suit your needs as described in *Modifying Disabled Escalation Paths or Policies* on page 114.

4 Click **Edit Filter** to create or modify the path filter. This displays the Filter Builder dialog.

   The filter determines what notifications follow this path of the policy. Similar to the policy filter, you can specify an expression filter, an ASL filter, or both. For more information on setting up a filter, see *Working with Filters* on page 121.

Note: For most escalation paths, you should add a filter that matches active notifications to ensure that you only escalate active notifications.

5. When you finish creating filters, click **OK** to close the Filtering Path dialog and click **OK** to close the Create a new Escalation Path dialog. Your path is displayed in the Configure Escalation Policy panel of the Global Manager Administration Console.

Repeat this procedure for each escalation path.

The next step is to add one or more levels to each escalation path.

Creating Escalation Levels

An escalation path requires at least one level. You can add up to six levels to each path.

1. Right-click on the path and select *Level > Insert* or select the path and choose *Escalation > Level > Insert*. This displays the Insert Level dialog.

   If the escalation path already contains a level, you can control where the new level is inserted:

   - To insert a new level at the end of the path, select the box at the end of the path to insert the level.
   - To insert a level before an existing level, select the level that you want to insert the new level in front of to insert the level.

2. Specify the duration of the level by typing a number and selecting the time unit from the drop-down list.
3. Select automated tools from the Available Tools list and move them to the Invoked Tools list. Tools are invoked immediately after a notification reaches this level.

   If you need to create a tool, see *Tool Configuration for the Global Manager* on page 83.

4. Click **OK** to close the Insert Level dialog box. The new level is displayed in the escalation path.
5. Repeat these steps for each level in the escalation path.
6. When the path is complete, click **Apply** to save the changes to the Global Manager.

Repeat this procedure for each escalation path in the escalation policy.

After you finish an escalation policy, you can enable one or more of its paths.

Modifying Escalation Policies and Paths

The types of modifications you can make to an escalation path are determined by whether the escalation path is enabled or disabled.

- When a path is enabled, you can edit the duration and tools for any escalation level in the path. You cannot add or remove escalation levels from the path or change the policy or path filters.
- When a path is disabled, you can edit the escalation path filter and change the duration or invoked tools at any escalation level. In addition, you can add or remove escalation levels.
- When all paths in a policy are disabled, you can edit the escalation policy filter.

Enabling and Disabling Escalation Paths

Escalation policies cannot be enabled or disabled. Instead, you can enable or disable the escalation paths within a policy. When all of the escalation paths within a policy are disabled, the policy is effectively disabled.

Disabling an escalation path cancels the escalation of all notifications currently scheduled within the path. New notifications are not added to a disabled path. To avoid losing any notifications currently scheduled on the path, but to prevent new notifications from being scheduled, you can retire the path instead of disabling it. For more information on retiring a path, see *Retiring an Escalation Path* on page 117.

Disable an Escalation Path

To disable an escalation path, do the following:

1. Select the escalation policy in the administration tree of the Global Manager Administration Console. This displays the Configure Escalation Policy panel.
2. Right-click on the path and select *Path > Disable*. You are prompted to confirm this choice.
3. Click **Yes** to disable the path. The path changes color to gray to indicate it is disabled.

4 Click **Apply**.

When all of the paths for an escalation policy are disabled, the name of the policy in the administration tree changes to gray to indicate that the policy has no active paths.

Enable an Escalation Path

When a path is enabled, the path begins to process any notifications with a Last Notify time that is later than when the path was enabled.

1. Select the policy in the administration tree. This displays the Configure Escalation Policy panel in the Global Manager Administration Console.
2. Right-click on the path and choose *Path > Enable* from the menu. The path changes color to indicate that it is enabled.
3. Click **Apply** to save the change to the Global Manager.

Modifying an Enabled Escalation Path

When you change the tools or duration of a level in an enabled path, the following is true:

- Notifications escalated beyond this level are unaffected.
- Notifications not yet escalated to this level are subject to the new duration or tools.
- Notifications currently at the modified level are rescheduled according to the new duration or use the new tools.

The following example illustrates what happens when the duration of a level is increased.

1. A notification enters Level 1 of an escalation path at 1:00 PM. The duration of the level is 20 minutes and the notification is scheduled to escalate to Level 2 at 1:20 PM.
2. At 1:10 PM, the duration of this level is increased to 30 minutes. As a result, the notification is rescheduled to escalate to Level 2 at 1:30 PM.

The following example illustrates what happens when the duration of a level is decreased.

1. A notification enters Level 1 of an escalation path at 1:00 PM. The duration of the level is 30 minutes and the notification is scheduled to escalate to Level 2 at 1:30 PM.

2. The notification is scheduled to enter Level 2 at 1:30 PM. The duration of Level 2 is 5 minutes and the notification is scheduled to enter Level 3 at 1:35 PM.
3. At 1:20 PM, the duration of Level 1 is decreased to 10 minutes. As a result, the notification escalates to Level 2 immediately. At Level 2, the notification is scheduled to escalate to level 3, at 1:25 PM because the duration of Level 2 is 5 minutes.

A change in duration will not cause a notification to escalate more than one level.

Changing the Tools or Duration of a Level

You can change the tools or duration of a level in an enabled escalation path.

1. Select the escalation policy name in the administration tree. The policy is displayed in the Configure Escalation Policy panel of the Global Manager Administration Console.
2. Right-click on the level you want to modify and select *Level > Edit Properties*. This displays the Edit Level dialog box displays.
3. Change the Duration or Invoked Tools:
   - Type a new value for Duration and select the units from the drop-down list.
   - Change the Invoked Tools list: select a tool in the Invoked Tools list and click → (right arrow) to remove it. Select a tool in the Available Tools list and click ← (the left arrow) to move the tool to the Invoked Tools list. If a tool is not available, you can create the tool by clicking **Tool Wizard**. For more information on creating automated tools, see *Tool Configuration for the Global Manager* on page 83.
4. Click **OK** to close the Edit Level dialog box.
   The changes to the Level display in the Path.
5. Click **Apply** to send the change to the Global Manager.

Modifying Disabled Escalation Paths or Policies

When an escalation path is disabled, you can modify the path filter and add or remove levels. If you disable all of the paths in an escalation policy, you can edit the escalation policy filter.

Disabling a path, cancels the escalation of any notifications along the path. If you want to modify a path or policy but do not want to cancel any scheduled notifications, you can retire the path or paths instead. This prevents new notifications from joining the path but allows scheduled notifications to continue to escalate. For more information on retiring a path, see *Retiring an Escalation Path* on page 117.

Modifying an Escalation Policy Filter

To modify the filter of an escalation policy, you must first disable all of the paths within the policy.

1. Select the escalation policy in the administration tree of the Global Manager Administration Console. This displays the policy in the Escalation Policy Configuration panel.
2. Select *Escalation Policy > Edit Properties*. to display the Edit Policy dialog.
3. Click **Edit Filters**, this displays the Filtering Policy dialog.

   Change the filters as needed. For more information on changing a filter, see *Working with Filters* on page 121.
4. When you finish editing the filters, click **OK** to close the Filtering Policy dialog box. Click **OK** to close the Edit Escalation Policy dialog box.
5. Click **Apply** to send the changes to the Global Manager. You can now enable the paths in the policy.

Modifying an Escalation Path Filter

To change a path filter, you must first disable the path.

1. Select the escalation policy in the administration tree of the Global Manager Administration Console. This displays the policy in the Escalation Policy Configuration panel.
2. Right-click in the path and select *Path > Edit Properties*. from the menu. This displays the Edit Path dialog.
3. Click **Edit Filters** to display the Filtering Path dialog.
4. Change the filters as needed. For more information on changing a filter, see *Working with Filters* on page 121.
5. When you finish editing the filters, click **OK** to close the Filtering Path dialog box. Click **OK** to close the Edit Escalation Path dialog box.
6. Click **Apply** to send the changes to the Global Manager. You can now enable the path.

Inserting Escalation Levels in a Path

1. For a disabled path, select the level (or Path End box) that will follow the new level.
   The selected level or box changes color to indicate that is selected.
2. Right click to display the Path menu. Choose *Level > Insert*.
   The Insert Level dialog box displays.
3. Type the Duration and select the units from the drop-down list.
4. Select any quantity or combination of tools from the Available Tools list. If a tool is not in list, you can create the tool by clicking **Tool Wizard**. For more information on creating automated tools, see *Tool Configuration for the Global Manager* on page 83.
5. Click ← (the left arrow) to move the tools that you selected to the Invoked Tools list.
6. Click **OK** to close the Insert Level dialog box.
   The Level displays in the Path.
7. Repeat Step 1 through Step 6 to add additional levels up to a total of six levels. Remember that new levels are always inserted in front of the selected level.
8. When your path is complete, click **Apply** to send the changes to the Global Manager. You can now enable the path.

Removing Escalation Levels from a Path

1. For a disabled path, select the level to remove.
   The selected level changes color to indicate that it is selected.
2. Right click to display the Path menu. Choose *Level > Remove*.
   The level disappears from the path.
3. Repeat Step 1 through Step 2 to remove additional levels. A valid path must have at least one level.
4. When your path is complete, click **Apply** to send the changes to the Global Manager. You can now enable the path.

Modifying Tools or Duration at an Escalation Level

This procedure is identical to the one performed when the path is enabled. See *Changing the Tools or Duration of a Level* on page 114.

Retiring an Escalation Path

To avoid losing escalations, consider retiring the path. Retiring an Escalation Path allows all notifications with a Last Notify time earlier than the retirement time to enter/continue escalation through the path. The path will not escalate any notifications with later Last Notify times.

1. Select the Policy name in the administration tree.
   The Policy appears in the right panel of the Administration Console.
2. Select the appropriate Path by clicking a Level in the Path.
   The Level changes color to indicate it is selected.
3. From the Escalation menu, choose *Path > Retire*.
   The Confirm Disable dialog box displays.
4. Click **Yes** to retire the path.
   The all levels in the path change color to one shade of blue to indicate the path is retired.
5. Click **Apply** to send the change to the Global Manager.

Later, after checking the retired path to determine whether any notifications are being escalated, you can disable or remove the path.

Modifying Escalation Paths Using the Retire Option

You may need to modify a path to add escalation levels or edit the filter. These changes require that the path is disabled. Disabling the path immediately terminates the escalation of any scheduled notifications.

The Retire option enables you to avoid this issue and provides a smooth transition to a new path. When a path is retired, notifications that are currently being escalated through the path continue to follow the escalation path. In addition, existing notifications that now meet the filtering criteria for the retired path will enter the path (from level 0). However, new notifications that meet the filtering criteria for the retired path are not added to the path.

Follow these steps to retire an existing path and replace it with a modified path:

1. Copy the enabled path you want to change.

Note: The copied path will be disabled.

2. Modify the path you copied.
3. Enable the new path, but do not Apply the changes.

4. Retire the existing path. See *Retiring an Escalation Path* on page 117.
5. Click **Apply** to simultaneously enable the new path and retire the original path. Notifications on the retired path continue to escalate while any new notifications are handled by the new path.
6. Remove the retired path when all notifications escalate through the entire path.

The new path only processes notifications that are notified after the enable time of the path. The existing path, because it is retired, processes any notifications with a notify time less than the time that the path was retired.

Removing or Deleting Escalation Paths and Policies

Enabled Paths and Policies can be removed or deleted. This terminates all scheduled escalations for the removed path or for all paths in a deleted policy.

Removing an Escalation Path

A Path can be removed when it is enabled, disabled, or retired.

1. Select the Policy name in the administration tree.
   The Policy appears in the right panel of the Administration Console.
2. Select the Path.
3. From the Escalation menu, choose *Path* > *Remove*.
   The Path is removed from Policy panel.
4. Click **Apply** to send the change to the Global Manager.

Deleting an Escalation Policy

A Policy can be removed with enabled, disabled, or retired paths. When an Escalation Policy is deleted, all escalation paths within the policy are deleted. All currently scheduled escalations will be unscheduled, and the policy will be removed from the system and no longer process any notifications.

1. Select the Policy name in the administration tree.
   The Policy appears in the right panel of the Administration Console.
2. From the Escalation menu, choose *Policy* > *Delete*.
   The Confirm Delete dialog box displays.

3 Click **Yes** to confirm the Policy deletion.
The Policy name is removed from the administration tree in the Administration Console.

4 Click **Apply** to send the change to the Global Manager.

Testing and Troubleshooting your Escalation Policies

Because Escalation Policies usually handle important notifications, it is a good practice to test your Escalation paths to ensure they function as desired.

InCharge allows you to quickly test and verify your Escalation Paths through the use of **sm_ems**, the Audit log, and the automated tools you intend to invoke.

After configuring and enabling an Escalation Policy and Paths, use **sm_ems** with the notify command option to issue a sample of the specific notification that you hope to escalate. For more information on using **sm_ems**, see the *InCharge Service Assurance Manager Adapter Platform User's Guide*.

Once your sample notification appears in Notification Log Console, double click the notification and select the Audit Log tab. Check the log for Escalation-related entries in Table 25. Escalation-related Audit Log entries always have the userid of SYSTEM.

Finally, ensure your Automated Tools are invoked and function as intended.

TYPE	AUDIT LOG DESCRIPTION	DESCRIPTION
Action Completed	<tool name>	The named Automated Tool was invoked and completed.
Action Failed	<tool name>	The named Automated Tool was invoked, but failed.
Action Invoked	<tool name>	The named Automated Tool was invoked.
ESCALATION	MATCHED:<policy name/path name>	Notification has characteristics that match the filter criteria for the specified Escalation Policy and Path.
ESCALATION	REACHED:<policy name>/<path name>, <level number>	Notification has reached the indicated escalation level in the specified path.
ESCALATION	SCHEDULED:<policy name>/<path name> for <level number> at <date> <time>	Notification is scheduled to reach the next escalation level at the specified date and time.

ESCALATION	TERMINATED ESCALATION: Unmatched <policy name>/<path name>	A change in the notification means it no longer matches the filter criteria for the Escalation Policy and Path and is removed from the policy.
ESCALATION	TERMINATED ESCALATION: Disabled <policy name>/<path name>	The Escalation policy or path was disabled and notifications are no longer escalated within the policy or path.
ESCALATION	TERMINATED ESCALATION: Notification Archived, <policy name>/<path name>	The notification was archived after it entered the Escalation Policy and was removed from the policy.
ESCALATION	RESCHEDULED: <policy name>/<path name> from <level> due at <datestamp> <timestamp>	An action within a level was rescheduled to a new date or time.
ESCALATION	END OF PATH: <policy name>/<path name>, <level number>	Notification has gone through all Escalation Levels and reached the end of the named Escalation path.

Table 25: Escalation-related Entries in the Audit Log

Viewing Notifications being Escalated

Once an escalation path is enabled, you can monitor any notifications that escalate through the path. You can select an escalation level within a path and view the notifications which are currently escalated to the level. You can then select any of the notifications to see the notification properties.

1. Select the Policy name in the administration tree.
   The Policy appears in the right panel of the Administration Console.
2. Select the appropriate Path by clicking a Level in the Path.
   The Level changes color to indicate it is selected.
3. From the Escalation menu, choose *Level > Show Notifications*.
   The Escalated Notifications dialog box displays any escalated notifications currently at the level.
4. Double click an escalated notification to display its Notification Properties.

Note: Notification data displayed is static. Changes are not automatically updated in the view, so click **Refresh** to update the view.

5. Click **Close** to close the Escalated Notifications dialog box.

9

Working with Filters

A filter screens incoming notifications by comparing a set of defined criteria against the attributes of a notification or ICIM class. Values that match the criteria pass through the filter. Filters are used to configure the following:

- Notification lists, described in *Notification Lists* on page 30
- Context and Status Criteria of client and server tools, described in *Context and Status Criteria for Client and Server Tools* on page 97
- Escalation paths and policies, described in *Developing Escalation Policies* on page 109

You can define two types of filters: an expression filter or an ASL filter. You create an expression filter using the Filter Builder, which is described in the following section. An ASL filter, specified in the ASL Filter check box of the Filter Builder, is defined in a separate ASL file. For more information about creating ASL filters, see *ASL Filters* on page 128.

Building Expression Filters

An expression filter uses matching criterion to filter against the value of an attribute of the specified class. Matching criterion are composed of a series of characters, and, optionally, wildcards.

- For the complete list of notification attributes, see *Attributes for Matching Notification Properties* on page 146.

- For a description of wildcards, see *Wildcard Patterns* on page 151.

Table 26 describes the parameters of an expression filter.

PARAMETER	DESCRIPTION
Attribute	Name of the attribute. The expression filter matches against the value of this attribute.
matches	Matching pattern that is compared against the value of of an attribute. • For Boolean attributes, a pull down menu lists the values. • For attributes whose values are known, the Helper button can be clicked to lists potential values, taken from the notifications in the Global Manager. You can use wildcards within the value field.
AND (green plus sign)	Add more attributes to the filter sheet to match against, creating a more narrow filter. A notification must match all of the attributes on a sheet to pass the filter.
OR (Sheet Tab)	Add more filter sheets to filter. A notification must match at least one of the filter sheets to pass the filter. You can add sheets by clicking **Add Sheet** or selecting *Edit > Add Sheet*.

Table 26: Expression Filter Fields

Layout of the Filter Builder

The Filter Builder interface differs, as described below, for notification filters, context and status criteria, and escalation policies.

Filter Builder for Notification Lists

The Filter Builder for notification lists opens with a blank "Sheet:1" tab. The sheet contains a "Use ASL Script" check box and a green plus sign, which is used to add notification attributes to the sheet. For notification lists, you can specify an ASL filter, an expression filter, or both. In addition to adding attributes to a sheet, you can also add, copy, or delete sheets using the buttons on the left side of the Filter Builder. If you use a sheet to specify an ASL filter, you cannot specify an expression filter on that same sheet.

Filter Builder for Context and Status Criteria

The Filter Builder for specifying Context Criteria and Status Criteria opens with a blank "Sheet:1" tab. At the top, the sheet contains a "Use ASL Script" check box and a drop-down menu that lists Context objects or Status objects. For expression filters, you select a class from the object menu and then click the plus sign to add attributes your expression matches against. When using the Filter Builder for Context and Status Criteria, you can use either an expression filter or an ASL filter, but not both. In addition, you cannot add more sheets.

Filter Builder for Escalation Policies

The Filter Builder is used to specify policy filters and path filters. In both cases, you are matching against notification attributes. The Filter Builder dialog provides an Edit menu from which you can use to add, copy, or delete sheets from your filter. Check the "Use ASL Script" check box to specify an ASL filter or click the green plus sign to create an expression filter.

At the top of the dialog, the Edit menu enables you to add, copy, or delete a Sheet tab. On the left, the green plus sign button enables you to add one filter criterion at a time. Pull-down menus for each filter criterion enable you to select a notification attribute and a match condition such as a specific value or expression. You can also specify wildcard characters in a match condition. Values are case-sensitive.

Using the Filter Builder

An expression filter is composed of one or more sheets. Each sheet specifies an ASL filter or is composed of one or more attribute/value pairs. Each pair includes the attribute name and an expression to match against that attribute's value. To match a given sheet, a notification or topological element must match every attribute/value pair specified in the sheet.

When you specify multiple sheets, a notification or topological element, passes the filter if it matches at least one sheet.

Using the Value Helper

When defining the expression for a filter you can use the Helper button to launch a dialog that enables you to add values to an expression. The type of help the dialog provides depends on the value type of the attribute for which you defining an expression. For example, the Value Expression Helper for the Certainty notification attribute allows to specify a range. An expression for a string field can include wildcard patterns as well characters. For example, the Element Class helper displays a list of known Element Classes. You can select multiple names and the helper dialog automatically creates an OR expression with the selected values.

You can also use wildcards in the Value field. Type one or more characters in the Value field and specify a wildcard. After you type in the Value field, the Value Helper is disabled. For more information on wildcards, see *Wildcard Patterns* on page 151.

Figure 5 illustrates the Value Helper dialog with two Element Class elements chosen.

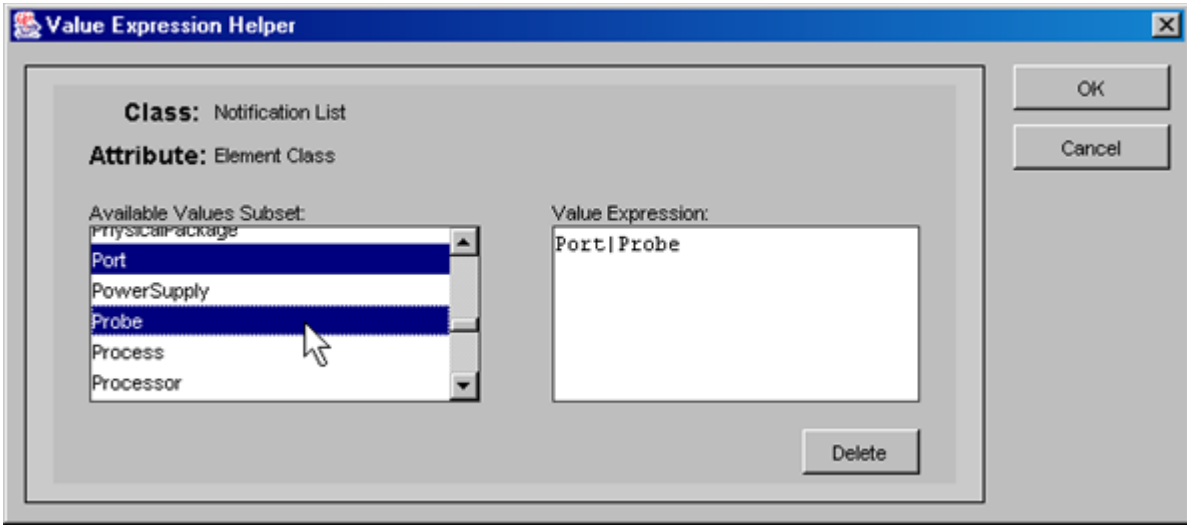

Figure 5: Filter Value Helper Dialog

After clicking **OK**, the dialog is closed and the expression is listed to the right of the attribute in the filter builder where you can manually edit it.

Filter Examples

The following examples illustrate different expression filters. The filter in the first example matches all notifications with a severity of one or two.

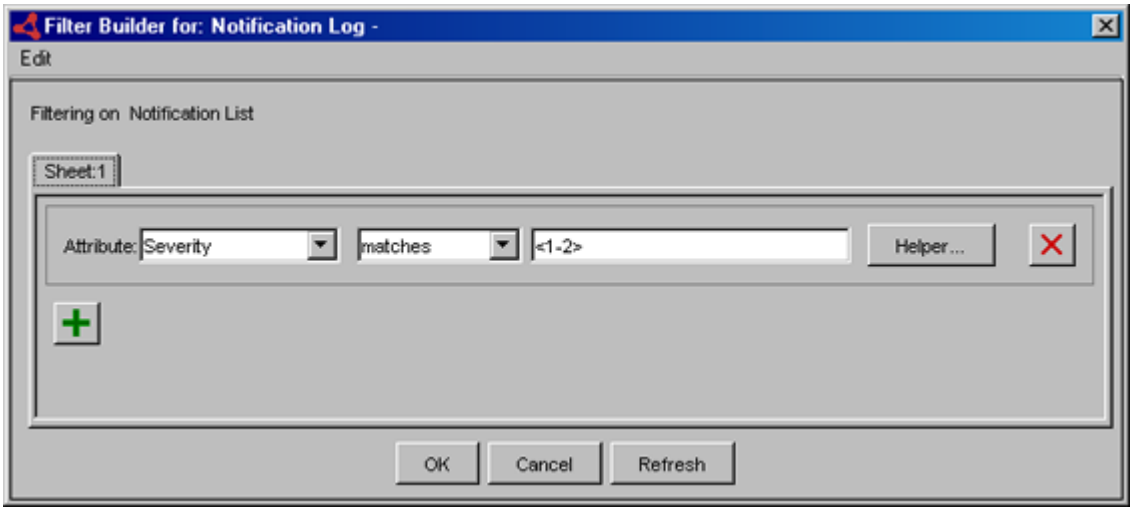

Figure 6: Filter Notifications with Severity of 1or 2

The filter in the following example matches all notifications with a value of Router or Switch in the Element Class attribute.

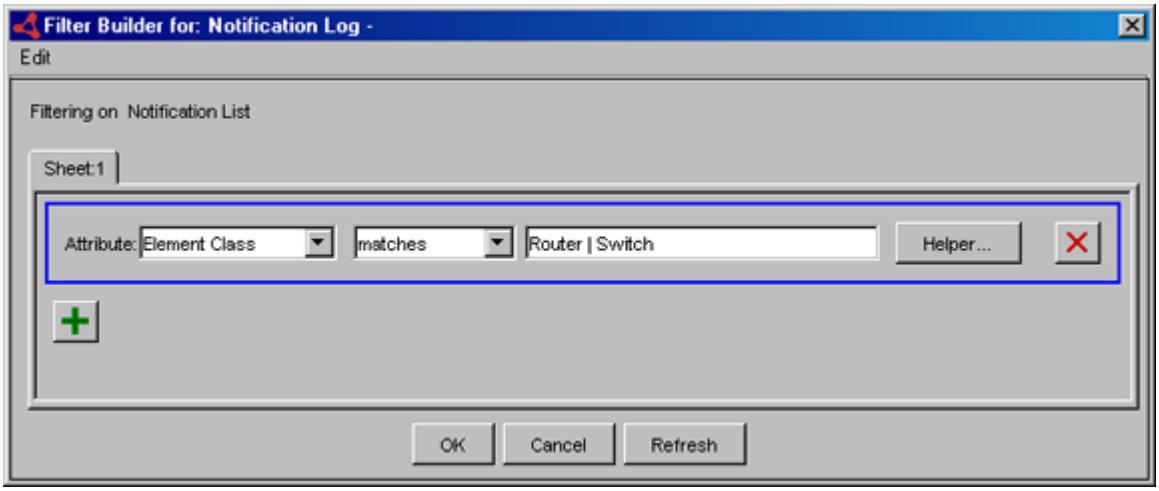

Figure 7: Filter Notifications of Element Class Router

The filter in the following example uses an implicit "or" condition by adding a second sheet. When you specify multiple sets of criteria as shown in the example, a notification passes the filter if it matches all of the conditions in one sheet. In this example, a notification passes the filter if it has a severity of one or two *OR* an impact between 5-100.

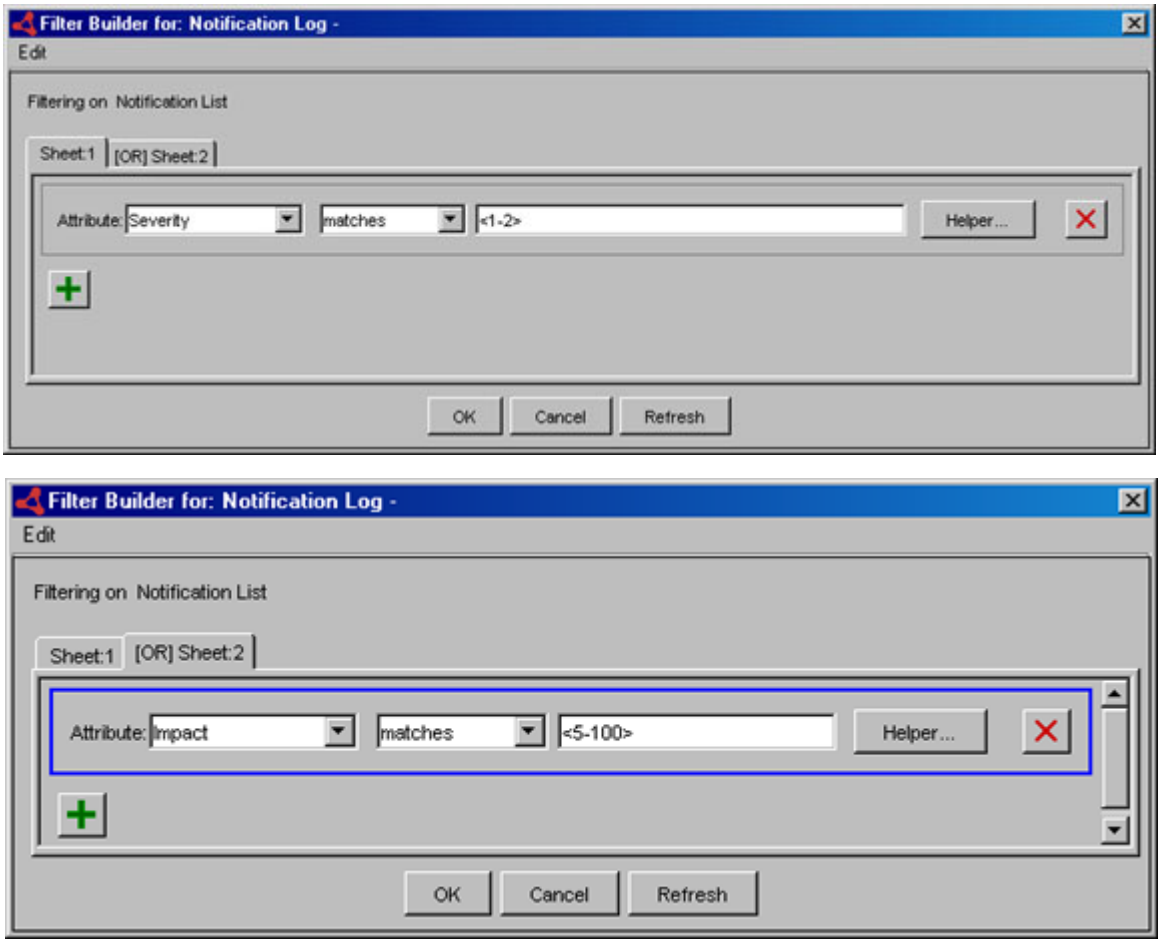

Figure 8: Filter Notifications with Severity of 1-2 OR an Impact of 5 or More

The filter example in Figure 9 matches all notifications that have an owner assigned to them unless that owner is "SYSTEM" or "maint."

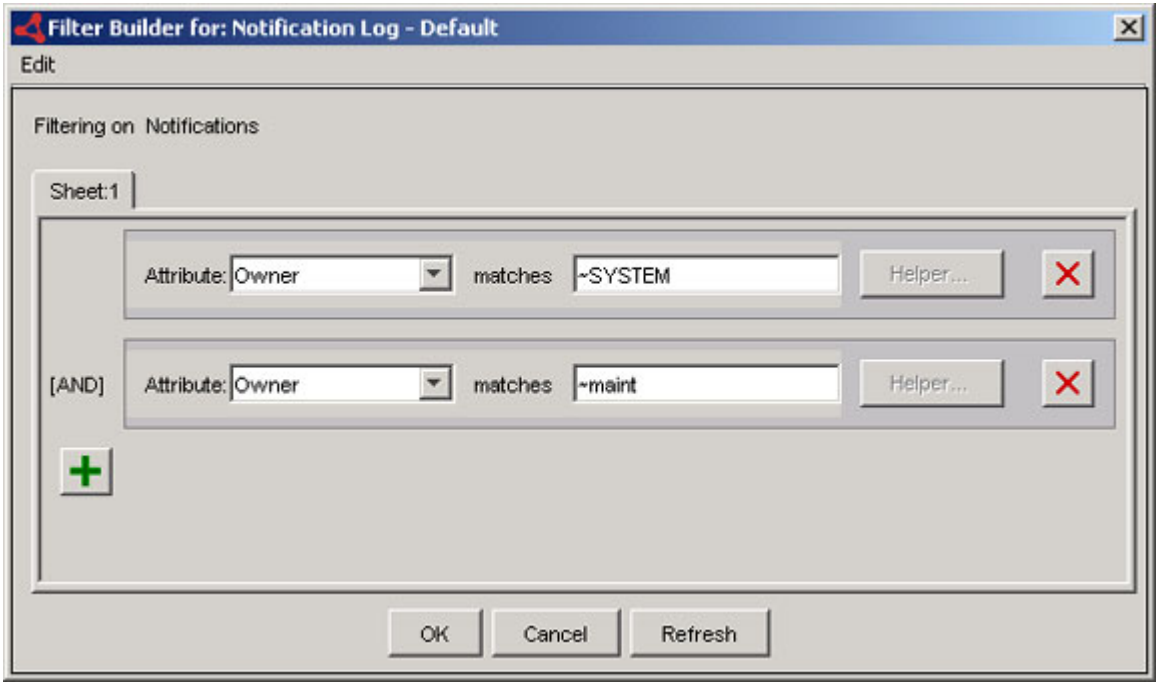

Figure 9: Filter Showing Notifications Owned by All Users Other Than "SYSTEM" or "maint"

The filter example in Figure 10 matches all notifications for routers with a value of one in the severity field.

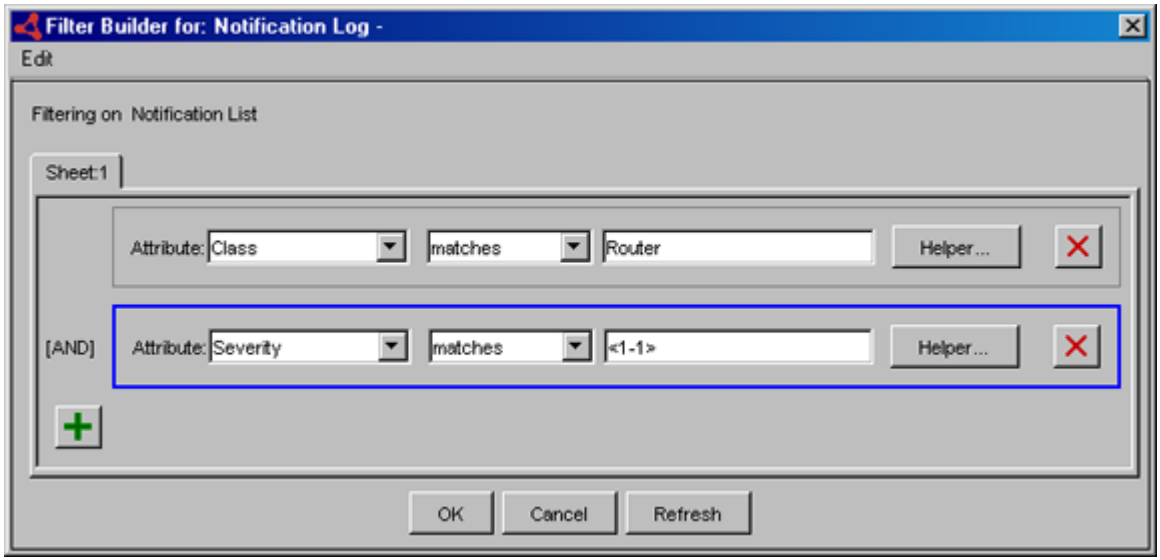

Figure 10: Filter Showing Routers with Severity 1

ASL Filters

If an expression filter does not provide enough flexibility to define filtering criteria, you can use an ASL filter. An ASL filter is an ASL program that processes each notification or other type of object.

An example ASL filter script, *nl-sample-filter.asl*, is included in the ***BASEDIR****/smarts/rules/ics* directory.

Notification Lists and Escalation Policies

The following example illustrates the format of an ASL filter used to filter Notification List and Escalation Policies. It filters for notifications with an EventName attribute "Failure."

The name of the notification object is assigned to the variable NotificationName. If the value of EventName is Failure, the value of Result is set to TRUE and the notification passes the filter.

```
START do {
      notification = object(NotificationName);
      if (notification->EventName == "Failure")
      {
         Result = TRUE;
      }
```

```
        else
        {
           Result = FALSE;
        }
}
```

To apply the ASL filter, access the filter builder. Check the "Use ASL Script" box and type the name of the appropriate ASL script.

Context and Status Criteria for Client and Server Tools

The following example illustrates the format of an ASL filter used to filter context and status criteria for client and server tools. It filters for notifications with an EventName attribute "Failure."

The name of the object is assigned to the variable ObjectName. If the value of EventName is Failure, the value of Result is set to TRUE and the notification passes the filter.

```
START do {
    timeNow = object(ObjectName)->nowString
    if (glob("*1<0-1>:*:* AM*|*<1-5>:*:* PM*|*12:*:* PM*",
timeNow))
    {
       Result = "TRUE";
    }
    else
    {
       Result = "FALSE";
    }
}
```

To apply the ASL filter, access the filter builder. Check the "Use ASL Script" box and type the name of the appropriate ASL script.

Saving ASL Filters

The ASL filter file must be located in the appropriate directory under the ***BASEDIR****/smarts/local/rules* directory. The files are located in different subdirectories for different products. For example, Service Assurance ASL files should be saved in ***BASEDIR****/smarts/local/rules/ics* and Adapter Platform (previously known as Open Integration) files should be saved in ***BASEDIR****/smarts/local/rules/icoi*.

10

Importing and Exporting Configurations

This chapter describes how to use **sm_config** to import and export InCharge repository objects such as notification lists, users, user profiles, tools, and escalation policies. It also describes how to use **sm_config** for configuring map icons.

The **sm_config** utility imports configurations, written in XML, for the Service Assurance Manager and the InCharge Adapter Platform (Adapter Platform, formerly referred to as Open Integration). The **sm_config** utility is used for the following purposes:

- As an alternative to creating notification lists, users and user profiles in the Global Manager Administration Console.
- To batch process multiple configurations.
- To duplicate configurations between Global Managers.
- To import tools for third-party applications.

You should have a strong understanding of XML if you intend to create and manage repository objects using XML. A guide for creating notification lists, user profiles, users, tools and escalation policies according to the *ics-config.dtd* XML specification can be found in the appendix *XML Reference* on page 155.

About the Default XML Files

Default Service Assurance settings are shipped in ***BASEDIR****/smarts/conf/ics* in a file called *ics-default.xml*. This file requires the DTD which must be in the same directory and is called *ics-config.dtd*.The *ics-default.xml* file also reads the *console_operations.xml* and *map_gif.xml* files.

The *ics-default.xml* file is only used when the Global Manager for Service Assurance is started without an repository (**.rps*) file. It creates the default repository objects if none are already defined in the repository. For example, the *ics-default.xml* file creates the default notification lists if there are no notification lists found in the repository.

Note: If you are currently using Service Assurance 5.0 or 5.0.1, you must run the **sm_migrate** utility to transfer your existing configurations to the current 6.2 environment.

The *console_operations.xml* file resides in ***BASEDIR****/smarts/conf/ics*. This file requires a DTD which must be in the same directory and is called *consoleoper-config.dtd*.

The *map_gif.xml* file resides in ***BASEDIR****/smarts/conf/maps*. The file also requires a DTD which must be in the same directory and is called *map-config.dtd*.

If you start the Global Manager with no objects in the repository, the *ics-default.xml* files is read to instantiate the Global Manager with default configurations. The objects added to the Global Manager are:

ics-default.xml

- Notification Lists:
  - "Default"
  - "Maintenance"
- Server Tools:
  - "Sample - Ping-Interface"
  - "Sample - Ping-IP"
  - "Sample - Ping-device"
  - "Sample - Ping-all"
  - "Sample - Telnet"

    - "Sample - Open Trouble Ticket"
    - "Sample - Close Trouble Ticket"
    - "Notify Business Process Event"
    - "Update Business Process Event"
    - "Clear Business Process Event"
- Client Tools:
    - "Report Manager"
    - "Launch Web Browser"
    - "Dump Properties"
    - "Ping"
- User Profiles:
    - "admin-profile"
    - "default-profile"
    - "maint-profile"
    - "oper-profile"
- Users
    - "admin"
    - "default"
    - "maint"
    - "oper"

console_operations.xml

The *console_operations.xml* file contains configuration information related to console operations. This file cannot be modified.

map_gif.xml

The *map_gif.xml* file sets the default image file that is used as a map icon for ICIM classes. This file cannot be modified.

Running sm_config

Using the **sm_config** command you can import objects into and export objects from the repository. In addition, the command allows you to enable, disable, and delete objects.

The options for importing, exporting, and enabling configurations are specified as commands to **sm_config** when you invoke it at the command line. Table 27 lists the arguments you should specify when invoking **sm_config**, which is run from ***BASEDIR***/*smarts/bin*.

COMMAND OPTION / ARGUMENT	DESCRIPTION
COMMANDS	
`import [command-options] <input-xmlfile>`	Imports the specified XML file into the Global Manager.
`export --server= [<output-xmlfile>]`	Exports the current object configurations to XML. If no output XML file name is specified, the XML is exported as standard output.
`delete --classname=<class> --instancename=<instance> [--type=<action_type>]`	Removes the specified instance from the Global Manager repository.
`enable --classname=<class> --instancename=<instance> [--type=<action_type>]`	Enables the specified instance in the Global Manager repository.
`disable --classname=<class> --instancename=<instance> [--type=<action_type>]`	Disables, but does not delete, the specified instance in the Global Manager repository.
COMMAND OPTIONS	
`--force`	To be used with the `import` command. Imports repository objects even existing objects were modified since the XML was last exported. This overwrites existing repository objects that may have been changed. Also `-f` can be used.
`--replace`	To be used with the `import` command. Deletes all of the XML-configured repository objects that are currently existing in the Global Manager (specifically notification lists, user profiles, users, tools, escalation policies, and actions). New repository objects, as specified in the XML file, are created in their place. Also `-r` can be used.

COMMAND OPTION / ARGUMENT	DESCRIPTION
`--addmodify`	This is the default option for the `import` command. Adds new and modifies the existing repository objects. If an object is not specified in the XML file, then no change is made to it in the repository. If `--replace` is used with the `import` command, it overrides `--addmodify`. Also `-a` can be used.
`--classname=<class>`	To be used with the `delete`, `enable`, or `disable` command. Specifies the class name of the object you want to modify. Also `-c <class>` can be used.
`--instancename=<instance>`	To be used with the `delete`, `enable`, or `disable` command. Specifies the instance name of the object you want to modify. Also `-i <instance>` can be used.
`--type=<action_type>`	To be used with the `delete`, `enable`, or `disable` command when you are modifying an action or tool. Specifies the action type. The possible values are `server`, `client`, or `auto`. Also `-t <action_type>`. This option is required if `--classname=Action` can be used.
ARGUMENTS	
`<input-xmlfile>`	Name of the Input XML file. Input XML files for Service Assurance Manager should be located in ***BASEDIR****/smarts/local/conf/ics*. Only the file name (including the file extension) must be specified, not the path to the file.
`<output-xmlfile>`	Name of the file that is generated when exporting configurations. Output XML files for Service Assurance Manager are created in ***BASEDIR****/smarts/local/conf/ics*. Output XML files for the Adapter Platform are created in ***BASEDIR****/smarts/local/conf/icoi*. If an output XML File is not specified, the XML is written to standard output.
`<class>`	Supported Values: • `Action` (To be used for client, server, and automatic tools.) • `AutoActionPolicy` (To be used for Escalation Policies) • `NotificationList` • `UserProfile` • `User` NOTE: There is no map icon support for <class>. You cannot import an individual instance of a map icon.

COMMAND OPTION / ARGUMENT	DESCRIPTION
`<instance>`	Instance value is the logical name of the repository object.
`<action_type>`	Used with the `--type` command option to specify the type of action to delete, enable, or disable. Possible values are: • client • server • auto
STANDARD OPTIONS	
–broker=<broker_location>	Alternate location of the broker as `<host>:<port>`. This value can be verified in the SM_BROKER environment variable. This argument is optional. Also `-b <broker_location>` can be used.
–server=<server_name>	Name of InCharge Global Manager or Adapter Platform. This argument is required. Also `-s <server_name>` can be used.

Table 27: Arguments to sm_config for Managing Configurations

Importing Configurations

Adding new objects and modifying existing objects requires the following steps:

1. Define the objects in an XML file.
2. Import the XML file.

Defining Objects in XML

You have several options for creating XML files for import with **sm_config**, including:

- Export the current configurations to a file and modify the XML as needed. For information on exporting, see *Exporting Configurations* on page 139.
- Copy and modify an existing object from one of the XML sample files located in ***BASEDIR***/*smarts/conf/ics*.
- Write the XML for the new object.

When making any modifications to the XML, be sure to adhere to the standards of the following DTDs:

- *ics-config.dtd* located in ***BASEDIR****/smarts/conf/ics*
- *map_config.dtd* located in ***BASEDIR****/smarts/conf/maps.*

For Service Assurance XML standards, see the *XML Reference* on page 155.

Sample XML Configuration Files

Sample XML documents are provided for use as templates and can be edited to suit your needs. The following sample files are located in ***BASEDIR****/smarts/conf/ics*:

- Tools: *actionconfig-sample.xml*
- Notification Lists: *nlconfig-sample.xml*
- Escalation Policy: *policyconfig-sample.xml*
- User Profile: *profileconfig-sample.xml*
- User: *userconfig-sample.xml*
- Multiple types of repository objects: *ics-config-sample.xml*
- Map Icons: *mapgif-sample.xml*

Working with XML Sample Files

The simplest way to create new repository objects in XML is to use a template. Sample XML files are located in ***BASEDIR****/smarts/conf/ics* are provided for this purpose. Use **sm_edit** to open and save a copy of this file in your ***BASEDIR****/smarts/local* directory. For example, to create a new notification list, open the *nlconfig-sample.xml* file using **sm_edit**:

```
# BASEDIR/smarts/bin>sm_edit /conf/ics/nlconfig-sample.xml
```

Modify the current notification list example to create a new one that suits your needs.

For example, change the `name="Default"` attribute to name="NotificationList_1". Remove everything up to the `ColumnHeading` declaration and then change one of the column headings to display the "Certainty" percentage of the event. The revised portion of the sample files looks like:

```
<ics_config>

    <nlconfig name="NotificationList_1" enable="True"
timestamp="0">
        <filterconfig type="Expression">
                <criterion
attribute="Owner">~maint</criterion>
```

```
                <criterion
attribute="Owner">~SYSTEM</criterion>
        </filterconfig>

       <columnheading
                column="InstanceDisplayName">"Name"
</columnheading>
       <columnheading
                column="EventDisplayName">"Event"
</columnheading>
       <columnheading
    column="Certainty" type="Percentage">"Certainty"
       </columnheading>

    </nlconfig>

</ics_config>
```

Note: When creating new users, the *serverConnect.conf* file and the *clientConnect.conf* file must be modified to authenticate them with usernames and passwords. For more information on modifying these files, see the *InCharge System Administration Guide*.

Importing XML Files to the Global Manager

When you have prepared an XML file with the appropriate additions or changes, you can import it into the Global Manager. By default, the **sm_config** import command will add new objects and will modify those objects that already exist in the repository. For example, to import a file called NewUsers.xml located in ***BASEDIR****/smarts/local/conf/ics*, use the following procedure:

1. Change directory to ***BASEDIR****/smarts/bin*
2. Run the command:
   `# sm_config -s INCHARGE-SA import NewUsers.xml`
3. Verify that the new configurations were imported by viewing them in the Global Manager Administration Console.

If you want to replace all of the current repository objects with new objects in your configuration file, replace the `--addmodify` option with the `--replace` option.

If you want to override any configuration changes that might have been made since you last exported the XML, use the `--force` option.

Importing Options

There are three options for the **sm_config** import command: addmodify, replace, and force.

- `--addmodify`: Use this option (without the `--force` option) and the timestamp of the repository object is monitored. If an object you configured was modified since the last time you exported, the object will not be imported in order to protect the modified configurations.
- `--replace`: This option should be used when you want to remove all of the XML-configurable objects that already exist in the repository and start fresh. This option is typically used when copying configurations from one Global Manager to another in order to duplicate the repository objects.
- `--force`: Use this option with the `--addmodify` option to override any values of existing XML-configurable objects.

As an example, you would use the following command to import configurations from a file called *ics-conf.xml* into a Global Manager called INCHARGE-SA and to ensure that your configurations do *not* override any changes made from the Global Manager Administration Console:

```
# sm_config -s INCHARGE-SA import --addmodify ics-conf.xml
```

Exporting Configurations

Exporting the repository objects to XML enables you to view the current configurations of your Global Manager. Exporting objects from one Global Manager in XML and then importing them to a second Global Manager allows you to duplicate configurations between Global Managers.

Before exporting, you need to know the name of the appropriate InCharge Manager and the location (host and port) of the broker. Exporting produces an XML representation of the current configurations for your notification lists, users, user profiles, tools, escalation policies, and map icons. By default, the XML file is created in the ***BASEDIR****/smarts/local/conf/ics* directory.

To export the repository object configurations, use the following command syntax:

```
# sm_config -s <Global_Manager_Name> ▼
▲--broker=<host:port> export <file_name>.xml
```

For example, to create an XML file called *Current_Configuration.xml* containing the repository objects in the INCHARGE-SA Global Manager on the host and port MyHost:426, the following command is used:

```
# sm_config -s INCHARGE-SA ▼
▲--broker=MyHost:426 export Current_Configuration.xml
```

Modifying Individual Repository Objects

Using **sm_config** you can also enable, disable, and delete individual repository objects. To modify these configurations, you need to know:

- Class name
- Instance name
- Action type (if the class name is Action)

Enabling or Disabling Repository Objects

Disabling an object deactivates it and prevents it from being used. To disable or enable an object from the repository, use the following command syntax:

```
# sm_config [options...] disable --classname=<class> ▼
▲--instancename=<instance>
```

To enable the object, replace the `disable` command with the `enable` command. For example, to enable a user profile called UserProfile1 in a Global Manager named INCHARGE-SA, use the following command:

```
# sm_config -s INCHARGE-SA enable --classname=UserProfile ▼
▲--instancename=UserProfile1
```

Deleting Repository Objects

Deleting a repository object completely removes it from the Global Manager. If you are not completely sure you want to remove it, consider disabling the object instead as described in *Enabling or Disabling Repository Objects* on page 140.

Note: Map objects cannot be deleted.

To delete an object from the repository, use the following command syntax:

```
# sm_config [options...] delete --classname=<class> ▼
▲--instancename=<instance>
```

For example, to delete a server action named Telnet from a Global Manager named INCHARGE-SA, use the following command:

```
# sm_config -s INCHARGE-SA delete --classname=Action ▼
▲--instancename=Telnet --type=server
```

Importing Tools Implemented By InCharge Adapters

If you are deploying an InCharge adapter that integrates with a third-party application, you may need to import an XML definition file for the tools implemented by that adapter.

The XML definition files for these adapters are imported into the Global Manager using **sm_config**. For more information, refer to *Importing XML Files to the Global Manager* on page 138. For more information about configuring InCharge Adapters, refer to the user's guide for that adapter.

For example, to import the tools definition for the Concord eHealth tool from a file called *configureEHealthTools.xml* into a Global Manager called INCHARGE-SA, execute the following command:

```
# sm_config -s INCHARGE-SA import configureEHealthTools.xml
```

A

ICIM Classes Used with Matching Criteria

To configure the Global Manager, you may need to specify a matching pattern or matching criteria. The matching pattern or criteria are compared to the name and/or value of system and notification attributes. Tasks where you might specify a matching pattern include:

- Creating notification lists.
- Specifying tagging patterns.
- Determining the status or criteria of tools.
- Creating selective groups with the Global Console

Matching criteria are a string that is matched against the value of the specified attribute. The criteria can contain any combination of text, integers, and wildcards.

Classes and Attributes for Matching Managed Systems

Table 28 lists the attributes of the system class hierarchy. The ICIM_System class is at the top of the hierarchy. As you move down the table, the classes become increasingly more specialized until you reach classes that represent specific systems such as routers and switches. Note that classes lower in the hierarchy inherit the attributes of classes higher in the hierarchy.

System Class	Attributes	Type	Description
ICIM_System	CreationClassName	String	Name of the class from which the element was instantiated.
	DisplayClassName	String	Name of the system's class that is displayed in the console.
	DisplayName	String	Name of the system that is displayed in the console.
	Description	String	Description of the system.
	IsManaged	Boolean	Determines if the system is monitored by Global Manager. Note that unmanaged elements do not appear in the Global Manager topology.
	Name	String	Name of the system.
	PrimaryOwnerContact	String	Contact information for the primary owner of this system.
	PrimaryOwnerName	String	Name of the primary owner of this system.
	SystemName	String	Name of the system of which this component is a part of or this service is hosted by.
ICIM_ComputerSystem			ICIM_ComputerSystem does not add attributes to the system class hierarchy. A computer system is a collection of managed system elements such as file systems, processors, and memory.

System Class	Attributes	Type	Description
UnitaryComputerSystem	Certification	String	Level of certification assigned to this device during discovery. Possible values are: • CERTIFIED • VALIDATED • TEMPLATE • GENERIC • UNCERTIFIED • UNDISCOVERED • UNSUPPORTED
	Location	String	Description of the physical location of this system.
	Model	String	Vendor name for the system.
	Type	String	Type of the computer system. Possible values are: • BRIDGE • HOST • HUB • MSFC • NODE • PROBE • ROUTER • RSFC • RSM • SWITCH • TERMINALSERVER • UNCERTIFIED
	Vendor	String	Name of the system's manufacturer.
Bridge	Type	String	Unitary computer system that bridges packets between separate segments. Value is BRIDGE.
Host	Type	String	Unitary computer system that represents a workstation or server. Value is HOST.
Hub	Type	String	Unitary computer system that connects multiple segments. Value is HUB.
MSFC			Unitary computer system that represents a Multi Layer Switch Feature Card. An MSFC is a card installed into a switch to perform routing between VLANs. Value is MSFC.
Node	Type	String	Unitary computer system that has not yet been certified by discovery.
Probe	Type	String	Unitary computer system that monitors networks or systems. Value PROBE.

System Class	Attributes	Type	Description
Router	Type	String	Unitary computer system that routes packets between computer networks. Value is ROUTER.
RSFC	Type	String	Unitary computer system that represents a Router Switch Feature Card. An RSFC runs Cisco IOS router software and directly interfaces with Catalyst switches to provide inter-VLAN routing. Value is RSFC.
RSM	Type	String	Unitary computer system that represents a Router Switch Module. Often installed in switches to route packets between VLANs. Value is RSM.
Switch	Type	String	Unitary computer system that switches packets between separate segments. Value is Switch.
TerminalServer	Type	String	Unitary computer system that represents a terminal server or similar access device. Value is TERMINALSERVER.
Uncertified	Type	String	[Deprecated, replaced by Node] Unitary computer system that represents a system that has not yet been certified by discovery. Value is HOST.

Table 28: System Classes and their Attributes

Attributes for Matching Notification Properties

Table 29 lists the notification attributes that are contained in notifications generated by the Global Manager. When you create a matching pattern, use the name of the attribute, not the column heading from the Notification Log.

Attribute	Default Column Name	Value Type	Description
Acknowledged	Acknowledged	Boolean	TRUE if the notification has been acknowledged, FALSE if not.
Active	Active	Boolean	TRUE if the notification is active, FALSE if not.
AuditTrail	n/a	String	Audit trail includes five fields separated by spaces. The fields include: • Serial number is a unique number that identifies the audit trail entry. • Timestamp identifies when this audit trail entry was written. • User is the InCharge user name associated with this audit trial entry. • Action type identifies the reason for the audit trail entry. • Text is a brief description of the audit trail entry.
Category	Category	String	Type of notification sent by the Global Manager. Possible values include: • Availability • Discovery • Error • IMPACT • Operational • Performance • PowerSupply • Resource • Temperature
Certainty	Certainty	Float	Confidence that this notification is the correct diagnosis. Value ranges from 0 to 100.
ClassDisplayName	Class	String	Name of the class that is displayed to the user.
ClassName	ClassName	String	Class name of the instance where this event occurred. This attribute, with InstanceName and EventName uniquely identifies this notification.
ClearOnAcknowledge	n/a	Boolean	Indicates that this event should be cleared when it is acknowledged. Default is FALSE.
ElementClassName	Element Class	String	Class name of the managed element most closely related to this event.
ElementName	Element Name	String	Name of the managed element most closely related to this event.

ATTRIBUTE	DEFAULT COLUMN NAME	VALUE TYPE	DESCRIPTION
EventDisplayName	Event	String	Name of the notification that is displayed to the user.
EventName	EventName	String	Name of this notification. This attribute, with ClassName and InstanceName, uniquely identify this notification.
EventState	Event State	String	Describes the state of the event. Value can be one of: • ACTIVE • WAS_ACTIVE • SUSPENDED • INACTIVE • UNITIALIZED
EventText	Event Text	String	A description of the notification.
EventType	Event Type	String	MOMENTARY when the notification has no duration. DURABLE if the notification has a period for which it is active, such as a link failure.
FirstNotifiedAt	First Notify	Integer	Time, in seconds, when the notification first became active.
Impact	Impact	Integer	Numeric value that indicates the effect of this event on related elements.
InMaintenance	In Maintenance	Boolean	TRUE if the device is in maintenance mode, FALSE if not.
InstanceDisplayName	Name	String	Name of the instance that is displayed to the user.
InstanceName	InstanceName	String	Name of the object where this notification occurred. This attribute, with ClassName and EventName, uniquely identify this notification.
IsRoot	IsRoot	Boolean	TRUE if the notification is an authentic problem (root cause), FALSE if not.
LastChangedAt	Last Change	Integer	Time, in seconds, when the status of the notification last changed.
LastClearedAt	Last Clear	Integer	Time, in seconds, when the notification was last cleared.
LastNotifiedAt	Last Notify	Integer	Time, in seconds, when the notification was last notified.
OccurrenceCount	Count	Integer	Number of times the notification has occurred.

ATTRIBUTE	DEFAULT COLUMN NAME	VALUE TYPE	DESCRIPTION
Owner	Owner	String	Name of the person responsible for this notification. Value is SYSTEM when acknowledged by the Global Manager.
Severity	Severity	Integer	Level of severity for this notification. 1 = CRITICAL 2 = MAJOR 3 = MINOR 4 = UNKNOWN 5 = NORMAL Note that only the numbers, not the text descriptions, are passed by the Global Manager.
SourceDomainName	Source	String	Name of the underlying domain that sent this notification. If more than one domain is listed, the names are separated by commas.
TroubleTicketID	Ticket ID	String	Trouble-ticket number associated with this notification.
UserDefined1-10	User Defined 1-10	String	Ten notification attributes can be defined by the user. You can set a value with a hook script specified in DomainType section of *ics.conf*.

Table 29: Notification Attributes

Note: For attributes that contain a time value, time is counted from Midnight, January 1st, 1970 (GMT). In the Global Console, these values are converted to a date and time.

B

Wildcard Patterns

A wildcard pattern is a series of characters that are matched against incoming character strings. You can use these patterns when you define pattern matching criteria.

Matching is done strictly from left to right, one character or basic wildcard pattern at a time. Basic wildcard patterns are defined in Table 30. Characters that are not part of match constructs match themselves. The pattern and the incoming string must match completely. For example, the pattern *abcd* does not match the input *abcde* or *abc*.

A compound wildcard pattern consists of one or more basic wildcard patterns separated by ampersand (&) or tilde (~) characters. A compound wildcard pattern is matched by attempting to match each of its component basic wildcard patterns against the entire input string. For compound wildcard patterns, see Table 31.

If the first character of a compound wildcard pattern is an ampersand (&) or tilde (~) character, the compound is interpreted as if an asterisk (*) appeared at the beginning of the pattern. For example, the pattern ~*[0-9]* matches any string not containing any digits. A trailing instance of an ampersand character (&) can only match the empty string. A trailing instance of a tilde character (~) can be read as “except for the empty string.”

Note: Spaces are interpreted as characters and are subject to matching even if they are adjacent to operators like “&”.

CHARACTER	DESCRIPTION
Note: Spaces specified before or after wildcard operators are interpreted as characters and are subject to matching.	
?	Matches any single character. For example, *server?.smarts.com* matches *server3.smarts.com* and *serverB.smarts.com*, but not *server10.smarts.com*.
*	Matches an arbitrary string of characters. The string can be empty. For example, *server*.smarts.com* matches *server-ny.smarts.com* and *server.smarts.com* (an empty match).
[set]	Matches any single character that appears within [set]; or, if the first character of [set] is (^), any single character that is *not* in the set. A hyphen (-) within [set] indicates a range, so that [*a-d*] is equivalent to [*abcd*]. The character before the hyphen (-) must precede the character after it or the range will be empty. The character (^) in any position except the first, or a hyphen (-) at the first or last position, has no special meaning. For example, *server[789-].smarts.com* matches *server7.smarts.com* through *server9.smarts.com*, but not *server6.smarts.com*. It also matches *server-.smarts.com*. Example: *server[^12].smarts.com* does not match *server1.smarts.com* or *server2.smarts.com*, but will match *server8.smarts.com*.
<n1-n2>	Matches numbers in a given range. Both *n1* and *n2* must be strings of digits, which represent non-negative integer values. The matching characters are a non-empty string of digits whose value, as a non-negative integer, is greater than or equal to *n1* and less than or equal to *n2*. If either end of the range is omitted, no limitation is placed on the accepted number. For example, *98.49.<1-100>.10* matches a range of IP addresses from *98.49.1.10* through *98.49.100.10*. Example of an omitted high end of the range: *<50->* matches any string of digits with a value greater than or equal to 50. Example of an omitted low end of the range: *<-150>* matches any value between zero and 150. A more subtle example: The pattern *<1-10>** matches 1, 2, up through 10, with * matching no characters. Similarly, it matches strings like 9x, with * matching the trailing x. However, it does not match 11, because <1-10> always extracts the longest possible string of digits (11) and then matches only if the number it represents is in range.
|	Matches alternatives. For example,"*ab|bc|cd*" without spaces matches exactly the three following strings: "*ab*", "*bc*", and "*cd*". A | as the first or last character of a pattern accepts an empty string as a match. Example with spaces "*ab | bc*" matches the strings "*ab*" and "*bc*".
\	Removes the special status, if any, of the following character. Backslash (\) has no special meaning within a set ([set]) or range (<n1-n2>) construct.

Table 30: Basic Wildcard Patterns

Special characters for compound wildcard patterns are summarized below.

&	"And Also" for a compound wildcard pattern. If a component basic wildcard pattern is preceded by & (or is the first basic wildcard pattern in the compound wildcard pattern), it *must* successfully match. Example: *NY*&*Router* matches all strings which contain NY and also contain Router. Example: <1-100>&*[02468] matches even numbers between 1 and 100 inclusive. The <1-100> component only passes numbers in the correct range and the *[02468] component only passes numbers that end in an even digit. Example: *A*|*B*&*C* matches strings that contain either an A or a B, and also contain a C.
~	"Except" for a compound wildcard pattern (opposite function of &).If a component basic wildcard pattern is preceded by ~, it *must not* match. Example: 10.20.30.*~10.20.30.50 matches all devices on network 10.20.30 except 10.20.30.50. Example: *Router*~*Cisco*&*10.20.30.*~10.20.30.<10-20>* matches a Router, except a Cisco router, with an address on network 10.20.30, except not 10.20.30.10 through 10.20.30.20.

Table 31: Compound Wildcard Patterns

C

XML Reference

This chapter describes the XML needed to create and modify InCharge repository objects such as notification lists, users, user profiles, tools, escalation policies, console operations, and map icons. The XML files you create are imported into the InCharge repository using the **sm_config** command as described in *Importing and Exporting Configurations* on page 131. You must have a strong understanding of XML if you intend to create and manage repository objects using XML.

Sample XML documents are provided to use as a template and can be edited to suit your needs. The following sample files are located in ***BASEDIR****/smarts/conf/ics*:

- Tools: *actionconfig-sample.xml*
- Notification Lists: *nlconfig-sample.xml*
- Escalation Policy: *policyconfig-sample.xml*
- User Profile: *profileconfig-sample.xml*
- User: *userconfig-sample.xml*
- Multiple types of repository objects: *ics-config-sample.xml*
- Map Icons: *mapgif-sample.xml*

A useful way of viewing valid, complex XML is to export the configurations you make in the Global Manager Administration Console. For more information on exporting configurations, see *Exporting Configurations* on page 139.

Structure of the XML Document

The basic building blocks of an XML document are elements, attributes, and values. "Tag" is a general term that can be used for elements and attributes. A relevant sample of the Service Assurance configuration XML DTD is provided with each of the structural descriptions.

Element

An element is a tag that is a container for subelements and attributes. For example, the `nlconfig` element contains defining information for the notification list you are adding or editing in the repository. If an element is comprised of any subelements, these are defined in parentheses after the element declaration in the DTD. For example, the notification list element, nlconfig, is comprised of a filter, column heading, and user profile as seen in this sample of the DTD:

```
<!ELEMENT  nlconfig  (filterconfig | columnheading |
userprofile)*>
```

Attribute Declaration (Attribute List)

Attribute declarations, as defined in the ATTList tag of the DTD, provide a detailed definition for an element. Many elements utilize a `name` declaration to identify it as well as an `enable` declaration that instructs the Global Manager to create and enable the object in repository.

An attribute is a source of additional information about an element, used to define the element.

- The value of an attribute must match one of the values listed. Values must appear in parentheses and separated by OR (|) symbols.
- Attribute may be required (#REQUIRED), optional (#IMPLIED), or have a default value in quotes.
- All attribute values must be quoted.

The following is an example of an attribute declaration in the DTD for the notification (nlconfig) element.

```
<!ATTLIST nlconfig
    name CDATA #REQUIRED
    enable (TRUE | FALSE | true | false | True | False)
"true">
timestamp CDATA "0"
```

This indicates that the notification list element has three attributes:

- name: a character string that is required
- enable: determines whether the repository object is enabled when it is imported. This attribute can take one of the values in parentheses. The default value is “true”.
- timestamp: character string that identifies the date and time the object was last modified. The “0” indicates that, by default, there is no timestamp.

Value

The value setting of the attribute declaration is paired with the declaration in “name=value” groupings. The values define the properties of the element you are adding, modifying, or deleting in the repository. In the following example, the notification list element is made up of a notification list element tag (“nlconfig”), attributes (“name” and “enable”), values (“Hosts_and_Routers” and “false”), and a close tag for the element (/>):

```
<nlconfig name="Hosts_and_Routers" enable="false">
</nlconfig>
```

An attribute value is either required or implied. If the value is an attribute that you must define such as the name of the object, this is indicated by the phrase “#REQUIRED” after the attribute in the DTD. If the value is implied, then a default value is indicated in quotes. For example, “`TRUE`” is the default value for the `enable` attribute of the `actionconfig` element.

```
<!ATTLIST actionconfig
    name CDATA #REQUIRED
    enable (TRUE | FALSE | true | false | True | False) "TRUE"
    type ( server | client | auto ) #REQUIRED >
```

Value Indicators

Table 32 lists the terms and symbols used to build the XML expressions with proper syntax structure.

VALUE INDICATOR	DESCRIPTION
`CDATA`	Character data (CDATA) type attributes may contain only character data.
`#PCDATA`	Parsed character data is text that will be examined by the parser for entities and proper syntax.
`<element>*`	The asterisk (*) indicates that the content is optional (may occur zero or more times).
`<element>?`	The question mark (?) after an element makes it optional, but only one may appear.

Table 32: XML Value Indicators

Configuration Elements

This section describes the elements of the *ics-config.dtd*. For each element, an explanation of the attributes and values is provided as defined by the DTD. An XML example for most elements is provided. Additional requirements and guidelines for the XML representations are defined in the Semantics section of each element.

ics_config

This is the root element of any Service Assurance configuration XML document. In any XML document that you create to be imported into the Global Manager, this root element must be the opening and closing tag of the document. There are no attributes for the `ics_config` element. All configuration elements are contained within the `ics_config` element tags.

ics_config DTD Description

Within the DTD, ics_config is defined as the root element. The `nlconfig`, `actionconfig`, `userprofileconfig`, `userconfig`, and `policyconfig` elements are listed as subelements in parentheses.

```
<!ELEMENT ics_config (nlconfig | actionconfig |
userprofileconfig | userconfig | policyconfig |
consoleoperationconfig | map_gifconfig )*>
```

ics_config XML Example

For example, to define a new user profile, it must be nested within the root ics_config element:

```
<ics_config>
    <userprofileconfig name="default" enable="True"
timestamp="0">
        <nl>Default</nl>
        <user>user-default</user>
        <console>NotificationLog</console>
    </userprofileconfig>
</ics_config>
```

Notification List (nlconfig)

This declares the notification list object and indicates that the subelements that can be nested are filters, column headings, and user profiles.

For more information on the purpose and standard configuration of notification lists, see *Managing Notifications with the Global Manager* on page 55.

nlconfig DTD Description

The nlconfig element defines the notification list repository object. In the *ics-config.dtd*, the element is defined as:

<!ELEMENT nlconfig (filterconfig | columnheading | userprofile)>

```
 <!ATTLIST nlconfig
    name      CDATA #REQUIRED
    enable    (TRUE | FALSE | true | false | True | False)
"TRUE"
    timestamp     CDATA "0" >
```

Table 33 identifies the attribute declarations and possible values for the `nlconfig` element.

XML Attribute	Description	Possible Value	Value Required/ Default Value
name	A unique name for the notification list.	CDATA	Required
enable	Indicates whether the notification list should be enabled when imported into the Global Manager or whether it was enabled at the time of export.	TRUE True true FALSE False false	Not Required. Default is “True” so the notification list is enabled when imported.
timestamp	When exporting, the value is the last time the object was modified by any user through the Administration Console or through **sm_config**. When importing, the timestamp defined in the XML file is compared with the timestamp of the object in the repository. Modifications to an existing object when the import timestamp is later than the existing repository timestamp (unless the `--force` option is used).	CDATA	Not Required. Default is “0”

Table 33: Attribute-list Declarations for the nlconfig Element

nlconfig XML Example

For example, a notification list named "NL1" that is enabled without a specific timestamp would be defined as:

```
<nlconfig name="NL1" enable="True" timestamp="0">
</nlconfig>
```

Filter (filterconfig)

The `filterconfig` element defines filter criteria that determines which notifications or topology elements may pass through it. The filterconfig element is a subelement to the following elements:

- `nlconfig`
- `policyconfig`
- `pathconfig`

- `status_criteria` (for actions)
- `context_criteria` (for actions)

filterconfig DTD Description

In the *ics-config.dtd*, the element is defined as:

```
<!ELEMENT  filterconfig (isa?, (criterion* | filename))>
```

This declares the filter object and indicates that the subelements that can be nested are isa, criterion, and filenames.

```
<!ATTLIST filterconfig
    type(ASL | EXPRESSION | asl | expression | Expression)
#REQUIRED
    enable        (TRUE | FALSE | true | false | True | False)
"TRUE">
```

Table 34 identifies the attribute declarations for the filterconfig element.

XML Attribute	Description	Possible Value	Value Required/ Default Value
type	One of two types of notification filters. For more information on expression and ASL filter types, see *Working with Filters* on page 121.	EXPRESSION Expression expression ASL asl	Required
enable	Indicates whether the filter should be enabled when imported into the Global Manager or whether it was enabled at the time of export.	TRUE True true FALSE False false	Not required Default is "True" so the filter is enabled when imported.

Table 34: Attribute-list Declarations for the filterconfig Element

filterconfig XML Example

For example, to indicate that an ASL filter called "*nl_notify.asl*" is to be utilized for a notification list called "NL1", use the following syntax:

```
<nlconfig name="NL1" enable="True" timestamp="0">
    <filterconfig type="ASL">
        <filename>nl_notify.asl</filename>
    </filterconfig>
</nlconfig>
```

In another example, the ics-default.xml file defines a filter for the "default" notification list that allows all notifications not owned by "maint" or "system." The XML representation of this is:

```
<filterconfig type="EXPRESSION" enable="TRUE">
    <criterion attribute="Owner">~maint</criterion>
    <criterion attribute="Owner">~SYSTEM</criterion>
</filterconfig>
```

filterconfig Semantics

You can have multiple criterion defined in a filter; these criteria are "AND"ed together. If you have multiple filterconfig definitions, these are "OR"ed together.

filename

This subelement is used when you are using an ASL filter. The `filename` refers to the ASL file to be used as the filter for the notification list. In the *ics-config.dtd*, the element is defined as:

```
<!ELEMENT filename (#PCDATA)>
```

filename XML Example

For example, to define an ASL filter using a file called nl_notify.asl, use the following syntax:

```
<filterconfig type="ASL" enable="TRUE">
    <filename>nl_notify.asl</filename>
</filterconfig>
```

isa

This `isa` element is used to define a filter for a particular ICIM class. There are no attributes associated with the `isa` element. In the *ics-config.dtd*, the element is defined as:

```
<!ELEMENT isa  (#PCDATA)>
```

Any ICIM class name is a valid #PCDATA value for this element.

isa XML Example

For example, to define an expression filter that allows all instances of the UnitaryComputerSystem class to pass through, use the following syntax:

```
<filterconfig type="Expression" enable="TRUE">
    <isa>UnitaryComputerSystem</isa>
</filterconfig>
```

criterion

The criterion element is used to populate an expression filter with the notification attributes against which the notifications or topology elements are matched.

Note: The `criterion` element utilized wildcards to build expressions. For more information on using wildcards, see *Wildcard Patterns* on page 151.

criterion DTD Description

In the *ics-config.dtd*, the element is defined as:

```
<!ELEMENT criterion (#PCDATA)>

<!ATTLIST criterion
    attribute     CDATA    #REQUIRED
```

Table 35 identifies the attribute declarations for the `criterion` element.

XML Attribute	Description	Possible Value	Value Required/ Default Value
attribute	Name of the notification attribute. Must be one of the attributes listed in *Attributes for Matching Notification Properties* on page 146.	#PCDATA	Required

Table 35: Attribute-list Declarations for the criterion Element

criterion XML Example

For example, to define an expression filter that will only pass on notifications that do not have an open trouble ticket ID (~*OPEN*), use the following syntax:

```
<filterconfig type="Expression" enable="TRUE">
    <criterion
attribute="TroubleTicketID">~*OPEN*</criterion>
</filterconfig>
```

Column Heading (columnheading)

The `columnheading` element defines the names of the columns in the notification log in the Global Console.

columnheading DTD Description

In the *ics-config.dtd*, the element is defined as:

<!ELEMENT columnheading (#PCDATA)>

```
<!ATTLIST columnheading
    column        CDATA   #REQUIRED
    column        (String | Boolean | Time | Integer | Float |
Percentage )      "String"
```

Table 36 identifies the attribute declarations for the columnheading element.

XML Attribute	Description	Possible Value	Value Required/ Default Value
column	The name of the notification attribute.	CDATA (string?)	Required
column	The text you want to display as the column heading in the Global Console for users of this notification list.	String Boolean Time Integer Float Percentage	Not required Default is "String" so the column heading text must be defined.

Table 36: Attribute-list Declarations for the columnheading Element

columnheading XML Example

For example, this sample represents a notification list called “NL1” with three columns. The ClassDisplayName attribute will be displayed in the “NL1 Class Name” column. The InstanceDisplayName attribute will be displayed in the “NL1 Display Name” column. Finally, the EventDisplayName attribute will be displayed in the “NL1 Event Name” column.

```
<nlconfig name="NL1" enable="True" timestamp="0">

    <columnheading column="ClassDisplayName">NL1 Class Name
       </columnheading>

    <columnheading column="InstanceDisplayName">NL1 Instance
Name
       </columnheading>

    <columnheading column="EventDisplayName">NL1 Event Name
       </columnheading>

</nlconfig>
```

Column Heading Values

The column headings for a notification list correspond to notification attributes. Table 37 describes the notification attributes, data types, and default display names of the column headings.

NOTIFICATION ATTRIBUTES	DATA TYPE	DEFAULT COLUMN NAME
"Name"	STRING	"InternalEventHandle"
"ClassName"	STRING	"ClassName"
"InstanceName"	STRING,	"InstanceName"
"EventName"	STRING	"EventName"
"ClassDisplayName"	STRING	"Class"
"InstanceDisplayName"	STRING	"Name"
"EventDisplayName"	STRING	"Event"
"ElementClassName"	STRING	"Element Class"
"ElementName"	STRING	"Element Name"
"SourceDomainName"	STRING	"Source"
"Active"	BOOLEAN	"Active"
"OccurrenceCount"	INTEGER	"Count"
"FirstNotifiedAt"	TIME	"First Notify"
"LastNotifiedAt"	TIME	"Last Notify"
"LastClearedAt"	TIME	"Last Clear"
"LastChangedAt"	TIME	"Last Change"
"IsRoot"	BOOLEAN	"IsRoot"
"Acknowledged"	BOOLEAN	"Acknowledged"
"Owner"	STRING	"Owner"
"EventType"	STRING	"Event Type"
"EventState"	STRING	"Event State"
"Category"	STRING	"Category"
"EventText"	STRING	"Event Text"
"Severity"	INTEGER	"Severity"

Table 37: Column Heading Attributes and Default Display Names

NOTIFICATION ATTRIBUTES	DATA TYPE	DEFAULT COLUMN NAME
"Impact"	INTEGER	"Impact"
"Certainty"	PERCENTAGE	"Certainty"
"InMaintenance", ,);	BOOLEAN	"In Maintenance"
"TroubleTicketID"	STRING	"Ticket ID"
"UserDefinedN	STRING	"User Defined N"

Table 37: Column Heading Attributes and Default Display Names

Tool/Action (actionconfig)

The actionconfig element identifies the tool repository object. For more information on the purpose and standard configuration of tools, see *Tool Configuration for the Global Manager* on page 83.

actionconfig DTD Description

In the *ics-config.dtd*, the element is defined as:

```
<!ELEMENT  actionconfig (program_name | timeout? | trace |
display | status_criteria? | context_criteria | userprofile
)*>

    <!ATTLIST actionconfig
     name          CDATA  #REQUIRED
     enable        (TRUE | FALSE | true | false | True | False)
"TRUE"
     type         (SERVER | CLIENT | AUTO | server | client |
auto | Server |
                  Client | Auto)#REQUIRED
     timestamp    CDATA          "0">
```

Note: The userprofile subelement is used by the Global Console to reference the user profile in the display. This is not used for configuration. To configure a user profile, use the userprofileconfig element as described in *User Profile (userprofileconfig)* on page 175.

This declares the tool object and indicates that the allowable subelements to be nested are:

- Program Name (see *program_name DTD Description* on page 169)
- Timeout (see *timeout DTD Description (for actionconfig)* on page 169)
- Trace (see *trace DTD Description* on page 171)
- Display (see *display DTD Description* on page 170)
- Status Criteria (see *status_criteria DTD Description* on page 172)
- Context Criteria (see *context_criteria DTD Description* on page 172)

Table 38 identifies the attribute declarations for the `actionconfig` element.

XML ATTRIBUTE	DESCRIPTION	POSSIBLE VALUE	VALUE REQUIRED/ DEFAULT VALUE
name	A unique name for the tool.	CDATA (string?)	Required
enable	Indicates whether the tool should be enabled when imported into the Global Manager or whether it was enabled at the time of export.	TRUE True true FALSE False false	Not required Default is "True" so the tool is enabled when imported.
type	Indicates the classification of the tool: • Server: tools executed on the same host as the Global Manager. These are displayed in the Tools menu on the Global Console. • Client: tools executed locally by the console. These are displayed in the Tools menu on the Global Console. • Auto: tools executed automatically through escalation policies or the automatic action adapter (**sm_adapter**). These are not visible in the tools menu on the Global Console.	SERVER Server server CLIENT Client client AUTO Auto auto	Required
timestamp	When exporting, the value is the last time the object was modified by any user through the Global Manager Administration Console or through **sm_config**. When importing, the timestamp defined in the XML file is compared with the timestamp of the object in the repository. Modifications to an existing object when the import timestamp is later than the existing repository timestamp (unless the `--force` option is used).	CDATA	Not required Default is "0"

Table 38: Attribute-list Declarations for the actionconfig Element

actionconfig XML Example

For example, a server tool named "Action1" that is enabled, running a server type program called "server-action.sh" without a specific timestamp would be defined as:

```
<actionconfig name="Action1" type="Server" enable="True"
timestamp="0">
    <program_name>server-action.sh</program_name>
</actionconfig>
```

Program Name (program_name)

This is the name of the program that the Global Manager (for server and automatic tools) or the host running the Global Console (for client tools) executes when the tool is invoked. This subelement is implicitly required when defining an action in XML.

program_name DTD Description

There are no attributes associated with the `program_name` element. In the *ics-config.dtd*, the element is defined as:

```
<!ELEMENT program_name (#PCDATA)>
```

program_name XML Example

For example, if the server tool called "Action1" is designed to execute a program called "server-action.sh" use the following syntax:

```
<actionconfig name="Action1" type="Server" enable="True"
timestamp="0">
    <program_name>server-action.sh</program_name>
</actionconfig>
```

Timeout (actionconfig)

This specifies the maximum number of seconds to wait for the tool script to complete. If the tool script does not complete within the specified timeout interval, the Global Manager terminates the tool.

timeout DTD Description (for actionconfig)

There are no attributes associated with the `timeout` element. In the *ics-config.dtd*, the element is defined as:

```
<!ELEMENT timeout (#PCDATA)>
```

timeout XML Example (for actionconfig)

For example, if the server tool called "Action1" is designed to execute a program called "*server-action.sh*" with a timeout value of 60 seconds, then the syntax to define that would look like this:

```
<actionconfig name="Action1" type="Server" enable="True"
timestamp="0">
    <timeout>60</timeout>
    <program_name>server-action.sh</program_name>
</actionconfig>
```

Display (display)

This specifies whether the tool should display output to the console. If this parameter is set to true, the console will open the Tool Output window immediately after running the tool. If false, the console only opens the Tool Output window if there is an error. The default is true.

display DTD Description

In the *ics-config.dtd*, the element is defined as:

```
<!ELEMENT display EMPTY>
<!ATTLIST display
    display_value (FALSE | TRUE | true | false | True | False)
"TRUE">
```

Table 39 identifies the attribute declarations for the display element.

XML Attribute	Description	Possible Value	Value Required/ Default Value
display_value	Indicates whether or not to open the Tool Output window when running the tool. If false, the Tool Output window will only open if an error occurs when the tool runs.	TRUE True true FALSE False false	Not required Default is "TRUE"

Table 39: Attribute-list Declarations for the display Element

display XML Example

For example, if the server tool called "Action1" is designed to execute a program called "*server-action.sh*" with a timeout value of 60 seconds, and you want to see the display output, then the syntax to define that would look like this:

```
<actionconfig name="Action1" type="Server" enable="True"
timestamp="0">
    <timeout>60</timeout>
    <program_name>server-action.sh</program_name>
    <display display_value="True" />
</actionconfig>
```

Trace (trace)

Trace indicates that additional debugging information will be included in the display output of the tool. This element is only valid if the display element is declared as well.

trace DTD Description

There are no attributes associated with the `trace` element. In the *ics-config.dtd*, the element is defined as:

```
<!ELEMENT trace EMPTY>
<!ATTLIST trace
    trace_value (FALSE | TRUE | true | false | True | False)
"FALSE">
```

The "EMPTY" declaration indicates that the element has no sub-elements or character data associated with it.

trace XML Example

For example, to turn on the trace utility when defining an action, use the following syntax:

```
<actionconfig name="Action1" type="Server" enable="True"
timestamp="0">
    <timeout>60</timeout>
    <program_name>server-action.sh</program_name>
    <display display_value="True" /diplay>
    <trace trace_value="True"/>
</actionconfig>
```

Status Criteria (status_criteria)

This element determines whether a client or server tool is accessible from the popup menu in the Global Console. The tool is always accessible when the `status_criteria` element is omitted. This parameter is not available for automated tools.

status_criteria DTD Description

There are no attributes associated with the `status_criteria` element. But, there is a `filterconfig` subelement that determines which notifications utilize the tool. In the *ics-config.dtd*, the element is defined as:

```
<!ELEMENT status_criteria (filterconfig?)>
```

status_criteria XML Example

For example, there is a server tool called "Action1" is designed to execute a program called "*server-action.sh*" with a timeout value of 60 seconds, with a display output. To set the status criteria to ensure that the tool only appears in the popup menu if there is an open trouble ticket for that notification, use the following syntax:

```
<actionconfig name="Action1" type="Server" enable="True"
timestamp="0">
    <timeout>60</timeout>
    <program_name>server-action.sh</program_name>
    <display display_value="True" /diplay>
    <status_criteria>
       <filterconfig type="Expression">
                 <criterion
attribute="TroubleTicketID">~*OPEN*</criterion>
                 </filterconfig>
    </status_criteria>
</actionconfig>
```

Context Criteria (context_criteria)

This determines whether the tool displays on the tool menu in the Global Console when the user right-clicks on a notification. The tool automatically displays in the tool menu when this parameter is omitted. This parameter is not available for automated tools.

context_criteria DTD Description

There are no attributes associated with the `context_criteria` element. But, there is a `filterconfig` subelement that determines which notifications utilize the tool. In the *ics-config.dtd*, the element is defined as:

<!ELEMENT context_criteria (filterconfig?)>

For example, there is a server tool called "Action1" is designed to execute a program called "*server-action.sh*" with a timeout value of 60 seconds, with a display output. To make sure that the tool is enabled in the popup menu for notifications on the ICIM class "Router," the syntax is:

```
<actionconfig name="Action1" type="Server" enable="True"
timestamp="0">
    <timeout>60</timeout>
    <program_name>server-action.sh</program_name>
    <display display_value="True" /diplay>
    <context_criteria>
        <filterconfig type="Expression">
                  <isa>ICIM_Notification</isa>
                  <criterion
attribute="ClassName">Router</criterion>
        </filterconfig>
    </context_criteria>
</actionconfig>
```

User (userconfig)

A user is an individual member of a user profile. If no user profile is specified, then the users are assigned to the "Default" user profile.

Note: The userconfig element contains a subelement called userprofile. This subelement is used by the Global Console to reference the user profile in the display. This is not used for configuration. To configure a user profile, use the userprofileconfig element as described in *User Profile (userprofileconfig)* on page 175.

userconfig DTD Description

In the *ics-config.dtd*, the element is defined as:

```
<!ELEMENT  userconfig (userprofile)*>
<!ATTLIST userconfig
    name          CDATA     #REQUIRED
    enable(TRUE | FALSE | true | false | True | False)"TRUE"
    timestamp   CDATA          "0">
```

Table 39 identifies the attribute declarations and possible values for the `userconfig` element.

XML Attribute	Description	Possible Value	Value Required/ Default Value
name	A unique name for the user.	CDATA (string?)	Required
enable	Indicates whether the user should be enabled when imported into the Global Manager or whether it was enabled at the time of export.	TRUE True true FALSE False false	Not required Default is "True" so the notification list is enabled when imported.
timestamp	When exporting, the value is the last time the object was modified by any user through the Administration Console or through **sm_config**. When importing, the timestamp defined in the XML file is compared with the timestamp of the object in the repository. Modifications to an existing object when the import timestamp is later than the existing repository timestamp (unless the `--force` option is used).	CDATA	Not required Default is "0"

Table 40: Attribute-list Declarations for the userconfig Element

userconfig XML Example

For example, if two users called "User1" and "User2" are designed to be enabled with no specific timestamp, then the syntax to define that would look like this:

```
<userconfig name="User1" enable="True" timestamp="0">
</userconfig>
<userconfig name="User2" enable="True" timestamp="0">
</userconfig>
```

Note: If you are creating new users, be sure they are authenticated with a username and password in the *serverConnect.conf* and *clientConnect.conf* files.

User Profile (userprofileconfig)

A user profile defines what a set of users can see and do when they log in to the Global Console. The user profile determines what notification lists, consoles, and actions a set users can access.

userprofileconfig DTD Description

In the *ics-config.dtd*, the element is defined as:

```
<!ELEMENT  userprofileconfig (nl | console | user | action |
consoleoperation)*> <!ATTLIST userprofileconfig
    name CDATA                #REQUIRED
    enable(TRUE | FALSE | true | false | True | False)"TRUE"
    timestamp   CDATA          "0">
```

This declaration states that the userprofileconfig element has four required subelements: `nl`, `console`, `user`, and `action`.

Table 41 identifies the attribute declarations and possible values for the `userprofileconfig` element.

XML Attribute	Description	Possible Value	Value Required/ Default Value
name	A unique name for the user profile.	CDATA (string?)	Required
enable	Indicates whether the user profile should be enabled when imported into the Global Manager or whether it was enabled at the time of export.	TRUE True true FALSE False false	Not required Default is "True" so the notification list is enabled when imported.
timestamp	When exporting, the value is the last time the object was modified by any user through the Administration Console or through **sm_config**. When importing, the timestamp defined in the XML file is compared with the timestamp of the object in the repository. Modifications to an existing object when the import timestamp is later than the existing repository timestamp (unless the `--force` option is used).	CDATA	Not required Default is "0"

Table 41: Attribute-list Declarations for the userprofileconfig Element

userprofileconfig XML Example

For example, in the *ics-default.xml* file, the "`maint-profile`" user profile is defined. It has the following attributes:

- one notification list ("`Maintenance`").
- one console ("`NotificationLog`").
- one user associated with it ("`maint`").

The "maint-profile" user profile also has several actions and console operations defined.

```
<userprofileconfig name="maint-profile" enable="True"
timestamp="0">
    <nl>Maintenance</nl>
    <console>NotificationLog</console>
    <user>maint</user>
    <action type="server">Sample - Telnet</action>
    <action type="server">Sample - Ping-Interface</action>
    <action type="server">Sample - Ping-IP</action>
    .
    .
    .
    <consoleoperation>Browse</consoleoperation>
    <consoleoperation>BrowseDetail</consoleoperation>
    <consoleoperation>ExpandMapNode</consoleoperation>
    <consoleoperation>ShowContainment</consoleoperation>
    .
    .
    .
</userprofileconfig>
```

Notification List (nl)

The `nl` is the name of notification list defined by the nlconfig element as seen in the section *Notification List (nlconfig)* on page 159.

nl DTD Description

There are no attributes associated with the `timeout` element. In the *ics-config.dtd*, the element is defined as:

```
<!ELEMENT nl (#PCDATA)>
```

nl XML Example

For example, in this user profile called "UserProfile1" there is a notification list called "NL1."

```
<userprofileconfig name="UserProfile1" enable="True"
timestamp="0">
    <nl>NL1</nl>
    <console>Console1</console>
    <console>Console2</console>
    <user>User1</user>
    <user>User2</user>
    <user>User3</user>
    <action type="server">Action1</action>
    <action type="client">Action1</action>
</userprofileconfig>
```

nl Semantics

When defining a user profile, you are required to provide a notification list. The `nl` subelement is required and only one is allowed.

Console (console)

The `console` element is the name of an InCharge console file (by default, the file extension is **.iccon* and does not need to be typed in the declaration). The console files are located in ***BASEDIR****/smarts/consoles* or ***BASEDIR****/smarts/local/consoles*.

console DTD Description

There are no attributes associated with the `console` element. In the *ics-config.dtd*, the element is defined as:

```
<!ELEMENT console (#PCDATA)>
```

console XML Example

For example, in this user profile called "UserProfile1" there are two consoles called "Console1" and "Console2."

```
<userprofileconfig name="UserProfile1" enable="True"
timestamp="0">
    <nl>NL1</nl>
    <console>Console1</console>
    <console>Console2</console>
    <user>User1</user>
    <user>User2</user>
    <user>User3</user>
    <action type="server">Action1</action>
    <action type="client">Action1</action>
</userprofileconfig>
```

console Semantics

The console subelement is required and multiple consoles are allowed.

User (user)

The `user` element is the name of user defined by the userconfig element, as seen in *User (userconfig)* on page 173.

user DTD Description

There are no attributes associated with the `user` element. In the *ics-config.dtd*, the element is defined as:

```
<!ELEMENT user (#PCDATA)>
```

user XML Example

For example, in this user profile called "UserProfile1" there are three users called "User1" and "User2" and "User3."

```
<userprofileconfig name="UserProfile1" enable="True"
timestamp="0">
    <nl>NL1</nl>
    <console>Console1</console>
    <console>Console2</console>
    <user>User1</user>
    <user>User2</user>
    <user>User3</user>
    <action type="server">Action1</action>
    <action type="client">Action1</action>
</userprofileconfig>
```

user Semantics

This subelement is not required but if user(s) are not defined no user will access the profile.

Tool/Action (action)

The action subelement refers to the name of a tool defined by the actionconfig element as seen in *Tool/Action (actionconfig)* on page 166. In addition to the name of the tool, you are required to define the type of tool (server, client, or automatic).

action DTD Description

In the *ics-config.dtd*, the element is defined as:

```
<!ELEMENT action (#PCDATA)>
<!ATTLIST action
```

```
type (SERVER | CLIENT | AUTO | server | client | auto |
      Server | Client | Auto) #REQUIRED>
```

Table 38 identifies the attribute declarations for the `actionconfig` element.

XML Attribute	Description	Possible Value	Value Required/ Default Value
type	Indicates the classification of the tool: • Server: tools executed on the same host as the Global Manager. These are displayed in the Tools menu on the Global Console. • Client: tools executed locally by the console. These are displayed in the Tools menu on the Global Console. • Auto: tools executed automatically through escalation policies or the automatic action adapter (**sm_adapter**). These are not visible in the tools menu on the Global Console.	SERVER Server server CLIENT Client client AUTO Auto auto	Required

Table 42: Attribute-list Declarations for the actionconfig Element

action XML Example

For example, in this user profile called "UserProfile1" there are two tools defined. The first is a server tool called "Action1" and the second is a client tool called "Action1."

```
<userprofileconfig name="UserProfile1" enable="True"
timestamp="0">
    <nl>NL1</nl>
    <console>Console1</console>
    <console>Console2</console>
    <user>User1</user>
    <user>User2</user>
    <user>User3</user>
    <action type="server">Action1</action>
    <action type="client">Action1</action>
</userprofileconfig>
```

action Semantics

If actions(s) are not defined the user will have no tools available in the tools menu of the Global Console. Multiple actions may be listed.

Console Operation (consoleoperation)

The `consoleoperation` element is the list of console operations associated with a user profile.

consoleoperation DTD Description

There are no attributes associated with the `consoleoperation` element. In the *consoleoper-config.dtd*, the element is defined as:

```
<!ELEMENT consoleoperation (#PCDATA)>
```

consoleoperation XML Example

The following example shows the console operations associated with the "training-profile" user profile.

```
<userprofileconfig name="training-profile" enable="True">
  <nl>Maintenance</nl>
   <console>NotificationLog</console>
   <user>User1</user>

  <consoleoperation>Browse</consoleoperation>
  <consoleoperation>BrowseDetail</consoleoperation>
  <consoleoperation>ExpandMapNode</consoleoperation>
  <consoleoperation>ShowContainment</consoleoperation>

</userprofileconfig>
```

Escalation Policy (policyconfig)

An Escalation Policy is composed of a policy filter and one or more escalation paths. If a policy filter is not defined, all notifications will be passed on to the paths for escalation.

policyconfig DTD Description

In the *ics-config.dtd*, the element is defined as:

```
<!ELEMENT  policyconfig  (filterconfig | pathconfig)*>
<!ATTLIST policyconfig
    name         CDATA              #REQUIRED
    timestamp   CDATA          "0"
```

Table 45 identifies the attribute declarations and possible values for the `policyconfig` element.

XML ATTRIBUTE	DESCRIPTION	POSSIBLE VALUE	VALUE REQUIRED/ DEFAULT VALUE
name	A unique name for the escalation policy.	CDATA	Required
timestamp	When exporting, the value is the last time the object was modified by any user through the Administration Console or through **sm_config**. When importing, the timestamp defined in the XML file is compared with the timestamp of the object in the repository. Modifications to an existing object when the import timestamp is later than the existing repository timestamp (unless the `--force` option is used).	CDATA	Not required Default is "0"

Table 43: Attribute-list Declarations for the policyconfig Element

policyconfig XML Example

This XML sample describes a policy called "Policy-Host." The expression filter indicates that all notifications pertaining to the ICIM class of Host will be entered into the policy.

There is one escalation path defined called "path1." An expression filter at the path level determines that only notifications with a Trouble Ticket ID are escalated."

The escalation actions open a trouble ticket for the notification after 60 minutes, then follows up with an E-mail after another 30 minutes.

```
<policyconfig name="Policy-Host" timestamp="0">

    <filterconfig type="EXPRESSION">
       <isa>ICIM_Notification</isa>
       <criterion attribute="ClassName">Host</criterion>
    </filterconfig>

    <pathconfig enable="TRUE" name="path1" retire="FALSE">
       <retireTime>0</retireTime> <enableTime>0</enableTime>
       <filterconfig type="EXPRESSION">
                 <isa>ICIM_Notification</isa>
                 <criterion attribute="TroubleTicketID">
</criterion>
       </filterconfig> <escalationlevel interval="1800"
level="0">
```

```
        </escalationlevel> <escalationlevel interval="900"
level="1">
        <action type="auto">Open Trouble Ticket</action>
        </escalationlevel> <escalationlevel interval="900"
level="2">
        <action type="auto">Email Manager</action>
</escalationlevel>
    </pathconfig>

</policyconfig>
```

Filter (filterconfig)

The `filterconfig` element within the `policyocnfig` element is the filter on the entire policy. This filter determines which notifications enter into the policy.

The `filterconfig` element within the `pathconfig` element is a filter on the escalation path. Notifications that enter the policy are filtered again to determine the path on which they will escalate.

For details of the `filter` element see *Filter (filterconfig)* on page 160.

Escalation Path (pathconfig)

You may have multiple escalation paths within an escalation policy.

pathconfig DTD Description

In the *ics-config.dtd*, the element is defined as:

```
<!ELEMENT  pathconfig  (filterconfig | retireTime |
enableTime | escalationlevel)*> <!ATTLIST pathconfig
    name  CDATA  #REQUIRED
    enable (TRUE | FALSE | true | false | True | False)
"False"
    retire    (TRUE | FALSE | true | false | True | False)
"False" >
```

Table 44 identifies the attribute declarations and possible values for the pathconfig element.

XML Attribute	Description	Possible Value	Value Required/ Default Value
name	A unique name for the escalation path.	CDATA	Required
enable	Indicates whether the escalation path should be enabled when imported into the Global Manager or whether it was enabled at the time of export.	TRUE True true FALSE False false	Not Required. Default is “False”
retire	Indicates whether the escalation path should be retired when imported into the Global Manager or whether it was retired at the time of export	TRUE True true FALSE False false	Not Required. Default is “False”

Table 44: Attribute-list Declarations for the pathconfig Element

pathconfig XML Example

For example, this XML sample defines an escalation path called “Path1.”

```
<pathconfig enable="TRUE" name="path1" retire="FALSE">
    <retireTime>0</retireTime>
    <enableTime>0</enableTime>
    <filterconfig type="EXPRESSION">
       <isa>ICIM_Notification</isa>
       <criterion attribute="TroubleTicketID"> </criterion>
    </filterconfig> <escalationlevel interval="1800"
    level="0">
</pathconfig>
```

Retire Time (retiretime)

This is the time interval, in seconds, for which an escalation path is retired.

retireTime DTD Description

There are no attributes associated with the `user` element. In the *ics-config.dtd*, the element is defined as:

```
<!ELEMENT retireTime (#PCDATA)>
```

retireTime XML Example

For example, this XML sample defines an escalation path called "Path1" with a retireTime of 5 minutes (300 seconds):

```
<pathconfig enable="TRUE" name="path1" retire="FALSE">
    <retireTime>300</retireTime>
    <enableTime>0</enableTime>
    <filterconfig type="EXPRESSION">
       <isa>ICIM_Notification</isa>
       <criterion attribute="TroubleTicketID"> </criterion>
    </filterconfig>
<escalationlevel interval="1800" level="0">
```

Enable Time (inabilities)

This is the duration far an escalation path that is retired.

enableTime DTD Description

There are no attributes associated with the `user` element. In the *ics-config.dtd*, the element is defined as:

```
<!ELEMENT enableTime (#PCDATA)>
```

enabletime XML Example

For example, this XML sample defines an escalation path called "Path1" with an enable interval of 15 minutes (900 seconds):

```
<pathconfig enable="TRUE" name="path1" retire="FALSE">
    <retireTime>0</retireTime>
    <enableTime>900</enableTime>
    <filterconfig type="EXPRESSION">
       <isa>ICIM_Notification</isa>
       <criterion attribute="TroubleTicketID"> </criterion>
    </filterconfig>
<escalationlevel interval="1800" level="0">
```

Escalation Level (escalationlevel)

Each Escalation Level includes a set of automatic tools to invoke and a time duration before a notification advances to the next level. As time progresses, a notification escalates to different levels in the path based on the level duration, and invokes tools to handle the notification.

escalationlevel DTD Description

In the *ics-config.dtd*, the element is defined as:

```
<!ELEMENT  escalationlevel (action)*>
```

```
<!ATTLIST escalationlevel
    level       CDATA    #REQUIRED
    interval    CDATA    #REQUIRED >
```

Table 45 identifies the attribute declarations and possible values for the `escalationlevel` element.

XML ATTRIBUTE	DESCRIPTION	POSSIBLE VALUE	VALUE REQUIRED/ DEFAULT VALUE
level	Level indicates the order of the level. Levels must be consecutive numbers beginning with "0".	CDATA (0-5)	Required
interval	The duration of time the notification spends in a level before going on to next level. This includes the time it takes to invoke the action at this level.	CDATA (in seconds)	Required

Table 45: Attribute-list Declarations for the escalationlevel Element

escalationlevel XML Example

For example, this sample defines two escalation levels:

- Level 0 will open a trouble ticket after a duration of 15 minutes (900 seconds)
- Level 1 will email a manager after a second duration of 15 minutes (900 seconds)

```
      </escalationlevel> <escalationlevel interval="900"
level="0">
                <action type="auto">Open Trouble
Ticket</action>
      </escalationlevel> <escalationlevel interval="900"
level="1">
                <action type="auto">Email Manager</action>
      </escalationlevel>
```

escalationlevel Semantics

You can specify that no tools are invoked at a level, but the duration cannot be less than one minute.

Map Icon Configuration

Custom map icons can also be configured using the XML interface. The specification for map icons has its own DTD, *map-config.dtd*, which is located in the ***BASEDIR****/smarts/conf/maps* directory.

The *map-config.dtd* defines the `mapgifconfig` element.

mapgifconfig Element

The `mapgifconfig` element is the root element of the *map-config.dtd*. It defines the subelements that specify the configuration for a map icon.

mapgifconfig DTD Description

In the *map-config.dtd*, the element is defined as:

```
<!ELEMENT  map_gifconfig  (map | repos | image)*>
```

The `map` element is not currently used.

repos Element

The `repos` element specifies a map configuration.

```
<!ELEMENT repos (#PCDATA)>
<!ATTLIST repos
  class  CDATA   #REQUIRED
  class (String | Boolean | Time | Integer | Float |
Percentage )  "String"
>
```

Table 46 identifies the attribute declarations for the map element.

XML ATTRIBUTE	DESCRIPTION	POSSIBLE VALUE	VALUE REQUIRED/ DEFAULT VALUE
class	The name of the ICIM class that the icon will represent.	CDATA (string?)	Required
class	The instance name of the ICIM class. If instance name is specified, the image/gif will be applied for that particular instance, otherwise "*" indicates that the image/gif is applied to all instances of that class.	String Boolean Time Integer Float Percentage	Not required Default is "String" so the column heading text must be defined.

Table 46: Attribute-list Declarations for the map Element

repos XML Example

The following example specifies a map icon for the DatabaseServer ICIM class.

```
<map_gifconfig>
    <repos class="DatabaseServer">*</repos>
    <image>DatabaseServer</image>
</map_gifconfig>
```

image Element

The `image` element specifies the name of the icon file, it has no attributes. The icon file must be of GIF format. It is not necessary to specify a filename extension such as ".gif". The icon file must be located in the ***BASEDIR****/smarts/local/conf/icons* directory.

image XML Example

The following example specifies the DatabaseServer image for the specified class.

```
<map_gifconfig>
    <repos class="DatabaseServer">*</repos>
    <image>DatabaseServer</image>
</map_gifconfig>
```

Service Assurance Configuration DTDs

The following DTDs identify the XML elements, attributes, and values for Service Assurance Manager configuration.

ics-config.dtd

The *ics-config.dtd* is located in the ***BASEDIR****/smarts/conf/ics* directory.

```
<?xml encoding="ISO-8859-1"?>

<!ELEMENT ics_config (nlconfig | actionconfig |
userprofileconfig | userconfig | policyconfig |
consoleoperationconfig | consoleoperationgroupconfig |
map_gifconfig)*>

<!ELEMENT  nlconfig  (filterconfig | columnheading |
userprofile)*>
<!ATTLIST nlconfig
    name         CDATA               #REQUIRED
    enable(TRUE | FALSE | true | false | True | False)"TRUE"
    timestamp   CDATA          "0"
>

<!ELEMENT columnheading (#PCDATA)>
<!ATTLIST columnheading
    column       CDATA   #REQUIRED
    column       (String | Boolean | Time | Integer | Float |
Percentage )      "String"
>
<!ELEMENT userprofile (#PCDATA)>

<!ELEMENT  userprofileconfig (nl | console | user | action |
consoleoperation)*>
<!ATTLIST userprofileconfig
    name         CDATA               #REQUIRED
    enable(TRUE | FALSE | true | false | True | False)"TRUE"
    timestamp   CDATA          "0"
>

<!ELEMENT nl (#PCDATA)>
<!ELEMENT user (#PCDATA)>
<!ELEMENT action (#PCDATA)>
<!ATTLIST action
    type (SERVER | CLIENT | AUTO | server | client | auto |
Server | Client | Auto) #REQUIRED
>
<!ELEMENT console (#PCDATA)>
```

```
<!ELEMENT  userconfig (userprofile)*>
<!ATTLIST userconfig
    name         CDATA               #REQUIRED
    enable(TRUE | FALSE | true | false | True | False)"TRUE"
    timestamp   CDATA          "0"
>

<!ELEMENT  actionconfig (program_name | timeout? | trace |
display | status_criteria? | context_criteria | user_prompts?
| userprofile)*>
<!ATTLIST actionconfig
    name         CDATA                #REQUIRED
    enable(TRUE | FALSE | true | false | True | False)"TRUE"
    type(SERVER | CLIENT | AUTO | server | client | auto |
Server | Client | Auto)#REQUIRED
    timestamp   CDATA          "0"
>

<!ELEMENT trace EMPTY>
<!ATTLIST trace
    trace_value (FALSE | TRUE | true | false | True | False)
"FALSE"
>
<!ELEMENT display EMPTY>
<!ATTLIST display
    display_value (FALSE | TRUE | true | false | True | False)
"TRUE"
>
<!ELEMENT program_name (#PCDATA)>
<!ELEMENT timeout (#PCDATA)>

<!-- Note: user prompts are only allowed on server-side
actions -->
<!ELEMENT user_prompts (one_prompt*)>

<!ELEMENT one_prompt (#PCDATA)>
<!ATTLIST one_prompt name CDATA #REQUIRED>

<!ELEMENT status_criteria (filterconfig?)>
<!ELEMENT context_criteria (filterconfig?)>

<!ELEMENT  filterconfig (isa?, (criterion* | filename))>
<!ATTLIST filterconfig
    type         (ASL | EXPRESSION | asl | expression |
Expression)  #REQUIRED
    enable(TRUE | FALSE | true | false | True | False)
"TRUE"
>

<!ELEMENT isa  (#PCDATA)>
```

```
<!ELEMENT filename (#PCDATA)>
<!ELEMENT criterion (#PCDATA)>
<!ATTLIST criterion
    attribute    CDATA   #REQUIRED
>

<!ELEMENT  policyconfig  (filterconfig | pathconfig)*>
<!ATTLIST policyconfig
    name         CDATA               #REQUIRED
    timestamp   CDATA          "0"
>

<!ELEMENT  pathconfig  (filterconfig | retireTime |
enableTime | escalationlevel)*>
<!ATTLIST pathconfig
    name         CDATA               #REQUIRED
    enable(TRUE | FALSE | true | false | True | False)"False"
    retire    (TRUE | FALSE | true | false | True | False)
"False"
>

<!ELEMENT retireTime (#PCDATA)>
<!ELEMENT enableTime (#PCDATA)>

<!ELEMENT  escalationlevel (action)*>
<!ATTLIST escalationlevel
    level      CDATA    #REQUIRED
    interval   CDATA    #REQUIRED
>
<!ENTITY % consoleoper-config SYSTEM "consoleoper-
config.dtd">
&consoleoper-config;
<!ENTITY % map-config SYSTEM "../maps/map-config.dtd">
%map-config;
```

consoleoper-config.dtd

The *consoleoper-config.dtd* is located in the BASEDIR/smarts/conf/ics directory.

```
<?xml encoding="ISO-8859-1"?>

<!ELEMENT consoleoperation (#PCDATA)>

<!ELEMENT consoleoperationconfig (#PCDATA)>
<!ATTLIST consoleoperationconfig
    name CDATA #REQUIRED
    enable (TRUE | FALSE | true | false | True | False) "TRUE"
```

```
    timestamp CDATA "0"
>

<!ELEMENT consoleoperationgroupconfig (consoleoperation)*>
<!ATTLIST consoleoperationgroupconfig
    name CDATA #REQUIRED
    displayname CDATA ""
    template (TRUE | FALSE | true | false | True | False)
"FALSE"
   enable (TRUE | FALSE | true | false | True | False) "TRUE"
   timestamp CDATA "0" >
```

map-config.dtd

The *map-config.dtd* is located in the BASEDIR/smarts/conf/maps directory.

```
<?xml encoding="ISO-8859-1"?>

<!ELEMENT  map_gifconfig  (map | repos | image)*>

<!ELEMENT map (#PCDATA)>
<!ATTLIST map
    class        CDATA   #REQUIRED
    class        (String | Boolean | Time | Integer | Float |
Percentage )  "String"
>

<!ELEMENT repos (#PCDATA)>
<!ATTLIST repos
    class        CDATA   #REQUIRED
    class        (String | Boolean | Time | Integer | Float |
Percentage )  "String"
>

<!ELEMENT image (#PCDATA)>
```

Index

B

C

D

E

F

L

M

P

R

S

T

U

V

W

X