

*InCharge*TM

Service Assurance Manager Adapter Platform User's Guide

Version 6.0



Copyright ©1996-2003 by System Management ARTS Incorporated. All rights reserved.

The Software and all intellectual property rights related thereto constitute trade secrets and proprietary data of SMARTS and any third party from whom SMARTS has received marketing rights, and nothing herein shall be construed to convey any title or ownership rights to you. Your right to copy the software and this documentation is limited by law. Making unauthorized copies, adaptations, or compilation works is prohibited and constitutes a punishable violation of the law. Use of the software is governed by its accompanying license agreement. The documentation is provided "as is" without warranty of any kind. In no event shall System Management ARTS Incorporated ("SMARTS") be liable for any loss of profits, loss of business, loss of use of data, interruption of business, or for indirect, special, incidental, or consequential damages of any kind, arising from any error in this documentation.

The InCharge products mentioned in this document are covered by one or more of the following U.S. patents or pending patent applications: 5,528,516, 5,661,668, 6,249,755, 10,124,881 and 60,284,860.

"InCharge," the InCharge logo, "SMARTS," the SMARTS logo, "Graphical Visualization," "Authentic Problem," "Codebook Correlation Technology," and "Instant Results Technology" are trademarks or registered trademarks of System Management ARTS Incorporated. All other brand or product names are trademarks or registered trademarks of their respective companies or organizations.

Third-Party Software. The Software may include software of third parties from whom SMARTS has received marketing rights and is subject to some or all of the following additional terms and conditions:

Bundled Software

Sun Microsystems, Inc., Java(TM) Interface Classes, Java API for XML Parsing, Version 1.1. "Java" and all Java-based marks are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries. SMARTS is independent of Sun Microsystems, Inc.

W3C IPR Software

Copyright © 2001-2003 World Wide Web Consortium (<http://www.w3.org>), (Massachusetts Institute of Technology (<http://www.lcs.mit.edu>), Institut National de Recherche en Informatique et en Automatique (<http://www.inria.fr>), Keio University (<http://www.keio.ac.jp>)). All rights reserved (<http://www.w3.org/Consortium/Legal/>). Note: The original version of the W3C Software Copyright Notice and License can be found at <http://www.w3.org/Consortium/Legal/copyright-software-19980720>.

The Apache Software License, Version 1.1

Copyright ©1999-2003 The Apache Software Foundation. All rights reserved. Redistribution and use of Apache source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of Apache source code must retain the above copyright notice, this list of conditions and the Apache disclaimer as written below.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the Apache disclaimer as written below in the documentation and/or other materials provided with the distribution.
3. The end-user documentation included with the redistribution, if any, must include the following acknowledgment:
"This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>)."
Alternately, this acknowledgment may appear in the software itself, if and wherever such third-party acknowledgments normally appear.
4. The names "The Jakarta Project", "Tomcat", "Xalan", "Xerces", and "Apache Software Foundation" must not be used to endorse or promote products derived from Apache software without prior written permission. For written permission, please contact apache@apache.org.
5. Products derived from this Apache software may not be called "Apache," nor may "Apache" appear in their name, without prior written permission of the Apache Software Foundation.

APACHE DISCLAIMER: THIS APACHE SOFTWARE FOUNDATION SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES, LOSS OF USE, DATA, OR PROFITS, OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This Apache software consists of voluntary contributions made by many individuals on behalf of the Apache Software Foundation and was originally based on software copyright © 1999, Lotus Development Corporation., <http://www.lotus.com>. For information on the Apache Software Foundation, please see <http://www.apache.org>.

FLEXIm Software

© 1994 - 2003, Macrovision Corporation. All rights reserved. "FLEXIm" is a registered trademark of Macrovision Corporation. For product and legal information, see <http://www.macrovision.com/solutions/esd/flexim/flexim.shtml>.

JfreeChart – Java library for GIF generation

The Software is a "work that uses the library" as defined in GNU Lesser General Public License Version 2.1, February 1999 Copyright © 1991, 1999 Free Software Foundation, Inc., and is provided "AS IS" WITHOUT WARRANTY OF ANY KIND EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED IN THE ABOVE-REFERENCED LICENSE BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL,

INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. JfreeChart library (included herein as .jar files) is provided in accordance with, and its use is covered by the GNU Lesser General Public License Version 2.1, which is set forth at <http://www.object-refinery.com/lgpl.html/>.

BMC – product library

The Software contains technology (product library or libraries) owned by BMC Software, Inc. ("BMC Technology"). BMC Software, Inc., its affiliates and licensors (including SMARTS) hereby disclaim all representations, warranties and liability for the BMC Technology.

Crystal Decisions Products

The Software may contain certain software and related user documentation (e.g., Crystal Enterprise Professional, Crystal Reports Professional and/or Crystal Analysis Professional) that are owned by Crystal Decisions, Inc., 895 Emerson Street, Palo Alto, CA 94301 ("Crystal Decisions"). All such software products are the technology of Crystal Decisions. The use of all Crystal Decisions software products is subject to a separate license agreement included with the Software electronically, in written materials, or both. YOU MAY NOT USE THE CRYSTAL DECISIONS SOFTWARE UNLESS AND UNTIL YOU READ, ACKNOWLEDGE AND ACCEPT THE TERMS AND CONDITIONS OF THE CRYSTAL DECISIONS' SOFTWARE LICENSE AGREEMENT. IF YOU DO NOT ACCEPT THE TERMS AND CONDITIONS OF THE CRYSTAL DECISIONS' SOFTWARE LICENSE, YOU MAY RETURN, WITHIN THIRTY (30) DAYS OF PURCHASE, THE MEDIA PACKAGE AND ALL ACCOMPANYING ITEMS (INCLUDING WRITTEN MATERIALS AND BINDERS OR OTHER CONTAINERS) RELATED TO THE CRYSTAL DECISIONS' TECHNOLOGY, TO SMARTS FOR A FULL REFUND; OR YOU MAY WRITE, CRYSTAL WARRANTIES, P.O. BOX 67427, SCOTTS VALLEY, CA 95067, U.S.A.

GNU eTeks PJA Toolkit

Copyright © 2000-2001 Emmanuel PUYBARET/eTeks info@eteks.com. All Rights Reserved.

The eTeks PJA Toolkit is resident on the CD on which the Software was delivered to you. Additional information is available at eTEKS' web site: <http://www.eteks.com>. The eTeks PJA Toolkit program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License (GPL) as published by the Free Software Foundation; version 2 of the License. The full text of the applicable GNU GPL is available for viewing at <http://www.gnu.org/copyleft/gpl.txt>. You may also request a copy of the GPL from the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA. The eTeks PJA Toolkit program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

For a period of three years from the date of your license for the Software, you are entitled to receive under the terms of Sections 1 and 2 of the GPL, for a charge no more than SMARTS' cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code for the GNU eTeks PJA Toolkit provided to you hereunder by requesting such code from SMARTS in writing: Attn: Customer Support, SMARTS, 44 South Broadway, White Plains, New York 10601.

IBM Runtime for AIX

The Software contains the IBM Runtime Environment for AIX(R), Java™ 2 Technology Edition Runtime Modules © Copyright IBM Corporation 1999, 2000 All Rights Reserved.

HP-UX Runtime Environment for the Java™ 2 Platform

The Software contains the HP-UX Runtime for the Java™ 2 Platform, distributed pursuant to and governed by Hewlett-Packard Co. ("HP") software license terms set forth in detail at: <http://www.hp.com>. Please check the Software to determine the version of Java runtime distributed to you.

DataDirect Technologies

Portions of this software are copyrighted by DataDirect Technologies, 1991-2002.

NetBSD

Copyright (c) 2001 Christopher G. Demetriou. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must display the following acknowledgement:
This product includes software developed for the NetBSD Project. See <http://www.netbsd.org/> for information about NetBSD.
4. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. <<Id: LICENSE, v 1.2 2000/06/14 15:57:33 cgd Exp>>

Contents

Preface	vii
Intended Audience	vii
Prerequisites	vii
Document Organization	viii
Documentation Conventions	viii
InCharge Installation Directory	ix
Additional Resources	xi
InCharge Commands	xi
Documentation	xi
Common Abbreviations and Acronyms	xii
Technical Support	xiii
1 Service Assurance Adapter Platform	1
Service Assurance Adapter Platform Architecture	2
Overview of Configuration for the Adapter Platform	4
Configuring the Adapter Platform Server	5
Editing the icoi.conf File	6
Editing the ics.conf File	8
Configuring Security for the Service Assurance Adapter Platform	9
Starting and Stopping the Adapter Platform Server	13
The sm_edit Utility	13
2 InCharge SNMP Trap Adapter	15
Configuring the InCharge SNMP Trap Adapter	18
Editing the trap_mgr.conf File	18
SNMP Trap Parameters	20
Advanced SNMP Trap Integration	30
Verifying the SNMP Trap Adapter Port Setting	32

SNMP Trap Batching	33
Starting and Stopping the SNMP Trap Adapter	33
3 InCharge Syslog Adapter	35
Configuring the InCharge Syslog Adapter	35
Syslog File Location	36
Editing the my_hook_syslog.asl	37
InCharge Syslog Adapter Parameters	40
Syslog Batching	46
Starting and Stopping the Syslog Adapter	47
4 Command-Line Interface	49
Using the Command-Line Interface	49
A Service Assurance Adapter Platform and Notifications	55
Index	61

Preface

The purpose of this guide is to provide information about how to integrate Service Assurance Manager with third-party applications using the Service Assurance Adapter Platform (formerly InCharge Open Integration).

Intended Audience

This guide is intended to be read by any user who needs to import third-party data into Service Assurance.

Prerequisites

This document assumes that the reader has an installed, functioning InCharge Service Assurance Global Manager, and installed the desired components of the Service Assurance Adapter Platform.

Document Organization

This guide consists of the following sections:

1. SERVICE ASSURANCE ADAPTER PLATFORM	Describes the Adapter Platform architecture and explains how to configure, secure, and operate the Adapter Platform server.
2. INCHARGE SNMP TRAP ADAPTER	Describes how to configure and operate the InCharge SNMP Trap Adapter (Receiver).
3. INCHARGE SYSLOG ADAPTER	Describes how to configure and operate the InCharge Syslog Adapter.
4. COMMAND-LINE INTERFACE	Describes how to use the <code>sm_ems</code> command-line interface.
A. SERVICE ASSURANCE ADAPTER PLATFORM AND NOTIFICATIONS	Describes all the attributes of a Service Assurance notification.

Table 1: Document Organization

Documentation Conventions

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

CONVENTION	EXPLANATION
<code>sample code</code>	Indicates code fragments and examples in Courier font
keyword	Indicates commands, keywords, literals, and operators in bold
<code>%</code>	Indicates C shell prompt
<code>#</code>	Indicates C shell superuser prompt
<code><parameter></code>	Indicates a user-supplied value or a list of non-terminal items in angle brackets
<code>[option]</code>	Indicates optional terms in brackets
<code>/InCharge</code>	Indicates directory path names in italics
<i>yourDomain</i>	Indicates a user-specific or user-supplied value in bold, italics
<code>File > Open</code>	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 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
IP Management Suite	<ul style="list-style-type: none"> • InCharge IP Availability Manager • InCharge IP Performance Manager • InCharge Discovery Manager • InCharge Adapter for HP OpenView NNM • InCharge Adapter for IBM/Tivoli NetView 	/IP
Service Assurance Management Suite	<ul style="list-style-type: none"> • InCharge Service Assurance Manager • Global Console • InCharge Service Assurance Manager Business Impact Manager • InCharge Service Assurance Manager Failover System • InCharge Service Assurance Manager Notification Adapters • InCharge Service Assurance Manager Adapter Platform • InCharge SNMP Trap Adapter • InCharge Syslog Adapter • InCharge XML Adapter • InCharge Adapter for Remedy • InCharge Adapter for TIBCO Rendezvous • InCharge Adapter for Concord eHealth • InCharge Adapter for InfoVista 	/SAM
Application Management Suite	<ul style="list-style-type: none"> • InCharge Application Connectivity Monitor 	/APP
SMARTS Software Development Kit	<ul style="list-style-type: none"> • Software Development Kit 	/SDK

Table 3: Product Suite Directory for InCharge Products

For example, on UNIX operating systems, version 6.0 of 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*.

Additional Resources

In addition to this manual, SMARTS provides the following resources.

InCharge Commands

Descriptions of InCharge commands are available as HTML pages. The *index.html* file, which provides an index to the various commands, is located in the **BASEDIR**/*smarts/doc/html/usage* directory.

Documentation

Readers of this manual may find other SMARTS documentation (also available in the **BASEDIR**/*smarts/doc/pdf* directory) helpful.

InCharge Documentation

The following SMARTS documents are product independent and thus relevant to users of all InCharge products:

- *InCharge Release Notes*
- *InCharge Documentation Roadmap*
- *InCharge Installation Guide*
- *InCharge System Administration Guide*
- *InCharge Operator's Guide*

InCharge Service Assurance Manager Documentation

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

- *An Introduction to InCharge Service Assurance Manager*
- *InCharge Service Assurance Manager Configuration Guide*
- *InCharge Service Assurance Manager Failover System User's Guide*
- *InCharge Service Assurance Manager User's Guide for Business Impact Manager*

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 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
IDS	Incremental Device Support
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 (9:00 A.M.—6:00 P.M. U.S. Eastern Time).

TECHNICAL SUPPORT: *support@smarts.com*

SALES: *sales@smarts.com*

WORLD WIDE WEB: *http://www.smarts.com*

TELEPHONE: +1.914.948.6200

FAX: +1.914.948.6270

You may also contact us at:

SMARTS
44 South Broadway
White Plains, New York 10601 U.S.A.

Service Assurance Adapter Platform

The Service Assurance Adapter Platform (Adapter Platform) imports and normalizes topology and event information from sources other than InCharge, such as SNMP traps, system log files, or events generated from the sm_ems command line interface. Once normalized to the InCharge Common Information Model™ (ICIM) data model, the information is transferred to the Service Assurance Global Manager (Global Manager).

The Service Assurance Adapter Platform works with third-party applications in a variety of ways to import topology and event information and prepare it for use by the Global Manager. The Adapter Platform allows for several methods of receiving events, including:

- SNMP trap integration
- Log file integration
- Integration using a custom adapter
- Command-line interface

The Service Assurance Adapter Platform works with any combination of these methods.

Regardless of how the Adapter Platform receives the event information, it does the following:

- Provides uniform representation of event information regardless of the source of the event. This is called event normalization.
- Consolidates recurring events, rather than considering each new event separately. This reduces the number of events sent to the Global Manager. It is called de-duplication.
- Allows for an incoming message to clear a previous event.
- Provides a mechanism to set an event expiration. This mechanism automatically clears events that have not changed for a period of time.
- Optionally associates events to a topology element, thus placing events in their topological context.

Once the event information is processed, it is sent to the Global Manager.

Service Assurance Adapter Platform Architecture

The Adapter Platform consists of four basic components shown in Figure 1:

- Service Assurance Adapter Platform server
The Adapter Platform server prepares event information for use by the Global Manager. It normalizes the event information and places it in a context understood by the Global Manager.
- InCharge Syslog Adapter
The InCharge Syslog Adapter parses a system log file and generates notifications based on the contents of the file. Either the complete contents of the file are read or the file can be tailed, which means that only newly added information is read.
- InCharge SNMP Trap Adapter (Receiver)
The InCharge SNMP Trap Adapter parses SNMP traps received and generates notifications based on the contents of the traps. The InCharge SNMP Trap Adapter can be configured to handle many different types of traps.

- Command-Line Interface (sm_ems)

The command-line interface creates or modifies notifications that are sent to the Global Manager. This interface may be used in conjunction with third-party applications to send events to the Global Manager.

Note: The InCharge Syslog Adapter and the InCharge SNMP Trap Adapter configurations must be customized to specify the relevant file contents or the SNMP traps to use and how the information maps to objects in the Global Manager repository.

In addition to the basic components, the Adapter Platform server may also be used to integrate events from Concord SystemEDGE agents by configuring the InCharge Adapter for Concord SystemEDGE. For more information, refer to the *InCharge Application Services Manager Adapters User's Guide*.

Topology Importer

In addition to the basic Adaptor Platform components, you can enable an Adapter Platform feature, the InCharge Topology Importer, which provides a consistent method of identifying the systems associated with Adapter Platform events.

The Adapter Platform creates notifications from different sources and tries to associate these notifications to the topology element where the notification occurred. Making this association can be difficult because there are many different methods of identifying topology elements.

The InCharge topology importer collects IP addresses and host names from an InCharge Availability Manager so the Adapter Platform can accurately place events in their topological context. The Adapter Platform updates this list as the information changes in the Availability Managers.

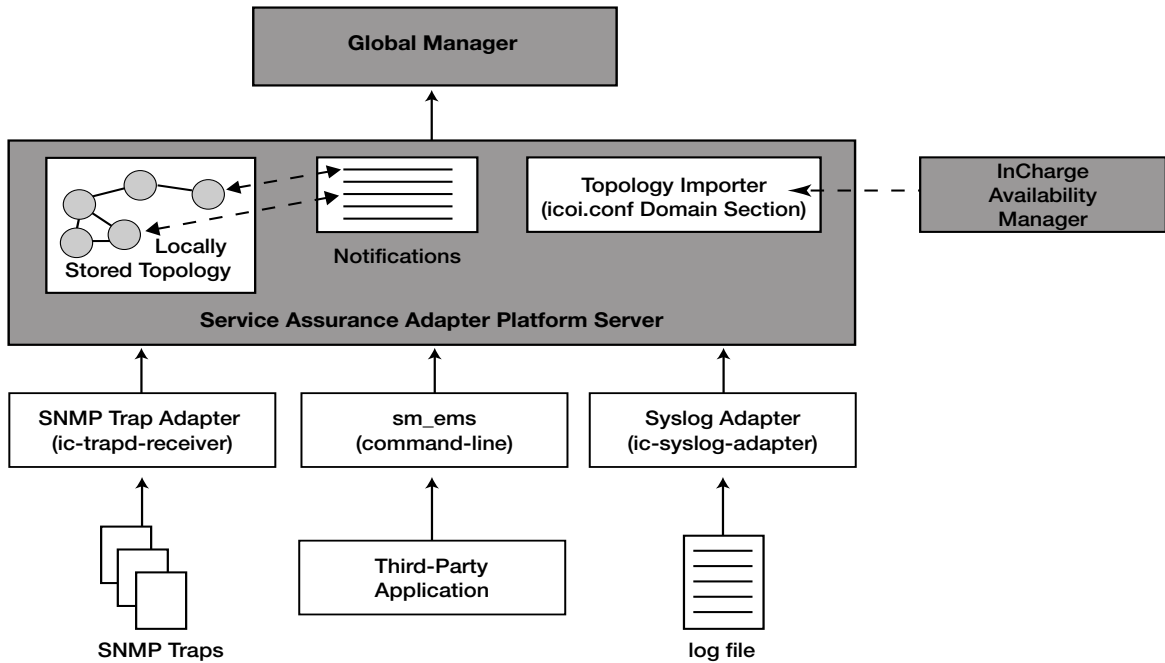


Figure 1: Service Assurance Adapter Platform Architecture

Overview of Configuration for the Adapter Platform

Configuring the Adapter Platform to import and normalize event and topology information requires that you complete one or more of the following procedures. Not all of these configuration procedures are relevant in all Adapter Platform deployments. So, depending on which components you are implementing, you can omit some of these steps.

- Configure the Adapter Platform Server
- Configure the InCharge SNMP Trap Adapter
- Configure the InCharge Syslog Adapter
- Configure the Adapter Platform server and adapters to start and stop automatically.

The following configuration sections describe each of these procedures.

Configuring the Adapter Platform Server

Configuring the Adapter Platform Server involves the following basic steps:

- 1 Edit *icoi.conf* file to suit your needs.
- 2 Configure the connection between a Global Manager and an Adapter Platform server. Use the `sm_edit` utility to edit *ics.conf* file as described in [Editing the ics.conf File](#) on page 8.
- 3 Configure security for the Adapter Platform.
- 4 Configure the Adapter Platform server to start and stop automatically.
- 5 Start the Adapter Platform server.

Table 4 describes the configuration files that are relevant to the Adapter Platform server. The `sm_edit` utility ensures that modified files are saved to the appropriate location in the **BASEDIR**/*smarts/local* directory. You should edit only local copies of the Open Integration configuration files, which are located in **BASEDIR**/*smarts/local/conf/icoi*. If local copies of the configuration files already exist and you attempt to edit non-local files, the `sm_edit` utility will locate and edit the local (not the non-local) versions of files. For information about the `sm_edit` utility, see [The sm_edit Utility](#) on page 13.

Note: For detailed instructions on how to properly modify an InCharge file, refer to the *InCharge System Administration Guide*.

DIRECTORY UNDER BASEDIR/	FILE NAME(S)	USER EDITABLE	DESCRIPTION
<i>smarts/conf/icoi</i>	<i>bootstrap.conf</i>	No	Contains vital information required to start the Adapter Platform server.
<i>smarts/conf/icoi</i>	<i>dxa-sysip.conf</i>	No	Used by the Adapter Platform to import information from cooperating InCharge Availability Managers.
<i>smarts/conf/icoi</i>	<i>icoi.conf</i>	Yes	Main configuration file for the Adapter Platform (or Open Integration).
<i>smarts/conf/icoi</i>	<i>icoi-config.dtd</i>	No	Definition file for XML configuration.
<i>smarts/conf/icoi</i>	<i>icoi-default.xml</i>	Yes	Contains the default notification lists, user profiles, and users for the Service Assurance Adapter Platform when the server is first started, or started without an existing repository file.

DIRECTORY UNDER BASEDIR/	FILE NAME(S)	USER EDITABLE	DESCRIPTION
<i>smarts/conf/icoi</i>	<i>icoi-config-sample.xml</i>	Yes	Sample XML file of notification lists, user profiles, and users. It can be used as a template for creating XML files of these entities.
<i>smarts/conf/icoi</i>	<i>nlconfig-sample.xml</i>	Yes	Sample XML file for notification list configuration. It can be used as a template for creating additional files.
<i>smarts/conf/icoi</i>	<i>profileconfig-sample.xml</i>	Yes	Sample XML file for user profile configuration. It can be used as a template for creating additional files.
<i>smarts/conf/icoi</i>	<i>userconfig-sample.xml</i>	Yes	Sample XML file for user configuration. It can be used as a template for creating additional files.

Table 4: Adapter Platform (Open Integration) Configuration Files

Editing the *icoi.conf* File

You need to add a reference in the DomainSection of *icoi.conf* for each Availability Manager you use as a source of topology if you want Adapter Platform events to be associated with an element in the Global Manager’s repository. The value for the Name parameter is the name of the InCharge Availability Manager. Availability Managers listed in the DomainSection of the *icoi.conf* file must also be listed in the DomainSection of the Global Manager’s *ics.conf* file.

```
DomainSection
{
    DomainType
    {
        ConfFile           = "dxa-sysip.conf";
        MinimumCertainty  = 0.0;
        SmoothingInterval = 0;
        Name               = "INCHARGE-AM";
    }
}
```

If you are not using Availability Manager as a source of topology, you should comment out these lines.

Note: Do not add the Global Manager to the DomainSection of *icoi.conf*. This section should only be a list of the Availability Managers used to collect topology information.

Defining System Defaults

System defaults are system-wide settings that affect the Adapter Platform and its clients, such as the InCharge Syslog Adapter or the InCharge SNMP Trap Adapter. There are six system default settings, all of which are specified in the SystemDefaultsSection of the *icoi.conf* file.

When specifying system defaults, keep in mind that events created in the Adapter Platform are not directly accessible. The Adapter Platform is used to normalize data so that it can propagate to and be managed from the Global Manager. Note that the same set of system defaults are also configurable for the Global Manager using the *ics.conf* file.

The following example illustrates the syntax of the SystemDefaultsSection.

```
SystemDefaultsSection
{
    # After how many seconds should an unowned inactive event
    # be automatically acknowledged?
    #
    AutoAcknowledgementInterval = 300;

    # After how many seconds should an inactive acknowledged
    # event be archived?
    #
    InactiveAutoArchiveInterval = 14400;

    # How large can the audit trail for a notification grow
    # before half of its contents are archived?
    #
    AuditTrailSizeLimit      = 100;
}
```

Table 5 describes the fields of the SystemDefaultsSection.

FIELD	DESCRIPTION
AutoAcknowledgementInterval	Interval, in seconds, after which a cleared (that is, inactive and unowned) notification is automatically acknowledged by the Adapter Platform server. Notifications that are auto acknowledged are owned by the user SYSTEM. Default is 300 seconds. An Acknowledged notification in the Adapter Platform does not propagate to the Global Manager.

FIELD	DESCRIPTION
InactiveAutoArchiveInterval	Interval, in seconds, after which a cleared (inactive) and acknowledged notification is archived by the Adapter Platform server. Default is 14400 seconds (4 hours). If this value is set to zero, archiving is disabled and events will not be deleted, causing the Adapter Platform to use more memory.
AuditTrailSizeLimit	Number of audit log entries for each notification that are saved in the Adapter Platform before the log contents are archived. When this limit is reached, half of the entries are written to the notification archive. The archive is written, by default, to the BASEDIR / <i>smarts/local/logs</i> directory. The name of the file is taken from the name of the Adapter Platform server and appended with <i>.archive</i> .

Table 5: Fields Defining SystemDefaultsSection

Editing the ics.conf File

You configure the connection between a Global Manager and an Adapter Platform server by modifying the local copy of *ics.conf* on the Global Manager. In the Domain section, minimally you need to specify the configuration file and the name of the Adapter Platform server. The configuration file must be *dxa-oi.conf*. For more information about configuring the Global Manager's *ics.conf*, see the *InCharge Service Assurance Manager Configuration Guide*.

Using a Custom Notification List

You do not need to create a notification list in the Adapter Platform server for the Global Manager to use. By default it connects to the Adapter Platform server's Default notification list. However, if you want to use a custom notification list, you can create it through the Global Manager Administration Console, or you can create it through the use of the *nlconfig-sample.xml* file and import it with the *sm_config* command. This allows you to send a subset of notifications from the Adapter Platform server to the Global Manager. Then you need to modify the local copy of the *dxa-oi.conf* on the Global Manager and change the following line:

```
sub    Default/n
to
sub    <notification_list>/n
```

Configuring Security for the Service Assurance Adapter Platform

It is important to secure access to the Service Assurance Adapter Platform server. To summarize, Service Assurance components authenticate users and determine their privileges through three files: *serverConnect.conf*, *clientConnect.conf*, and *brokerConnect.conf*. These files ensure that only authorized users access Service Assurance server applications. You must use `sm_edit` to modify these files, which are located in **BASEDIR**/*smarts/conf*.

The Adapter Platform is a server application. The following are possible clients of the Adapter Platform server:

- Broker
- Command-line Utilities (dmctl)
- Global Manager
- sm_ems
- InCharge SNMP Trap Adapter (Receiver)
- InCharge Syslog Adapter

Note: For a complete list of server and client programs, and a more detailed explanation of the Service Assurance security mechanism, refer to the *InCharge System Administration Guide*.

Figure 2 provides an overview of the security files required in a typical deployment of the Adapter Platform. A circle in the figure indicates that the component represented by the adjacent box is a client application. An arrowhead indicates that the component represented by the adjacent box is a server application. For example, the Adapter Platform is a client of the broker and the Adapter Platform has five clients—the broker, the Global Manager, the InCharge SNMP Trap Adapter, the InCharge Syslog Adapter, and sm_ems. Note that, although not depicted in the figure, the InCharge SNMP Trap Adapter, the InCharge Syslog Adapter, and sm_ems are also clients of the Broker because everything is a client of the Broker.

Note: Typically the Global Manager and the Adapter Platform are deployed on different hosts. If the Adapter Platform resides on the same host as another Service Assurance server application, such as the Global Manager, then they can share the same *serverConnect.conf* file. Likewise, if clients of the Adapter Platform reside on the same host as other Service Assurance client applications, they too can share the same *clientConnect.conf* file.

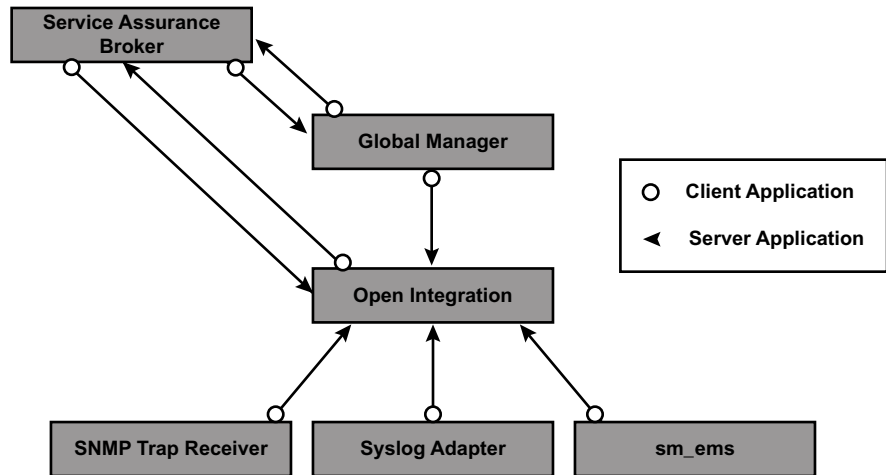


Figure 2: Security Configuration of the Service Assurance Adapter Platform

The following sections provide examples of authentication records for the client programs, InCharge SNMP Trap Adapter (Receiver), InCharge Syslog Adapter, and sm_ems command-line interface and corresponding records for the server program, Adapter Platform server. For examples of authentication records for the broker and the global manager, see the *InCharge System Administration Guide*.

InCharge SNMP Trap Adapter Security

The InCharge SNMP Trap Adapter (Receiver) is a client of the Adapter Platform server. As such it will use information in the *clientConnect.conf* file to provide authentication information to the Adapter Platform server. The Adapter Platform server will compare this information to entries in the *serverConnect.conf* file to validate the connection. Thus, both these files must be configured.

The *clientConnect.conf* file must be configured on the host where the InCharge SNMP Trap Adapter runs. In the *clientConnect.conf* file you should create an entry for the system user ID running the InCharge SNMP Trap Adapter with a corresponding InCharge user name and password. For example,

```
trap-uid1 : INCHARGE-OI : admin : changeme
```

where:

- trap-uid1 is the system user ID running the InCharge SNMP Trap Adapter

- INCHARGE-OI is the Adapter Platform server name
- admin is the InCharge login user name
- changeme is the InCharge login password

The *serverConnect.conf* must be configured on the same host as the Adapter Platform. In the *serverConnect.conf*, an entry must exist that corresponds to the InCharge user name and password specified in the *clientConnect.conf* file for the system user ID running the InCharge SNMP Trap Adapter. The InCharge user must be given All privileges. For example,

```
INCHARGE-OI : admin : changeme : ALL
```

where:

- INCHARGE-OI is the Adapter Platform (Open Integration) server name (you could use an * to indicate all Service Assurance servers)
- admin is the InCharge login user name
- changeme is the InCharge login password
- ALL is the privileges granted this user

The InCharge SNMP Trap Adapter does not have to run on the same host as the Adapter Platform. For more information about security configuration, refer to the *InCharge System Administration Guide*.

InCharge Syslog Adapter Security

The InCharge Syslog Adapter is a client of the Adapter Platform server. As such it will use information in the *clientConnect.conf* file to provide authentication information to the Adapter Platform server. The Adapter Platform server will compare this information to entries in the *serverConnect.conf* file to validate the connection. Thus, both these files must be configured.

The *clientConnect.conf* file must be configured on the host where the InCharge Syslog Adapter runs. In the *clientConnect.conf* file you should create a entry for the system user ID running the InCharge Syslog Adapter with a corresponding InCharge user name and password. For example,

```
syslog-uid1 : INCHARGE-OI : admin : changeme
```

where:

- syslog-uid1 is the system user ID running the InCharge Syslog Adapter
- INCHARGE-OI is the Adapter Platform (Open Integration) server name
- admin is the InCharge login user name

- changeme is the InCharge login password

Note: The system user ID that starts the InCharge Syslog Adapter will need adequate permissions (must be root) on UNIX.

The *serverConnect.conf* must be configured on the same host as Open Integration. In the *serverConnect.conf*, an entry must exist that corresponds to the InCharge user name and password specified in the *clientConnect.conf* file for the system user ID running the InCharge Syslog Adapter. The InCharge user must be given All privileges. For example,

```
INCHARGE-OI : admin : changeme : ALL
```

where:

- INCHARGE-OI is the Adapter Platform (Open Integration) server name (you could use an * to indicate all Service Assurance servers)
- admin is the InCharge login user name
- changeme is the InCharge login password
- ALL is the privileges granted this user

The InCharge Syslog Adapter runs on the same host as the Adapter Platform. For more information about security configuration, refer to the *InCharge System Administration Guide*.

sm_ems Command-Line Interface Security

The sm_ems command-line interface may or may not run on a different host than the Adapter Platform server. The *clientConnect.conf* file must be configured on the host running the command-line interface. You should create a unique user name and password that matches a user in *serverConnect.conf* and give that user All privileges. Typically the user name and password should not be prompted when using the command-line interface from a third-party application. However, if you want to run sm_ems manually, then prompting could be used. For more information about security configuration, refer to the *InCharge System Administration Guide*.

Starting and Stopping the Adapter Platform Server

If you installed the Adapter Platform Server as a service, it automatically starts when the system starts up. The following instructions describe how to use the `sm_service` utility to manually start and stop the Adapter Platform Server.

Issue one of the following from the command line:

```
% BASEDIR/smarts/bin/sm_service start ic-icoi-server
```

or

```
% BASEDIR/smarts/bin/sm_service stop ic-icoi-server
```

Note: The `sm_service` utility is operating system independent and works the same way on both UNIX and Windows operating systems.

For more information about the Adapter Platform service's default start-up options or how to modify them, refer to the *InCharge System Administration Guide*.

On Windows, if the Adapter Platform (Open Integration) was installed as a service, you can also start and stop the service using the Control Panel Administrative Tools.

The `sm_edit` Utility

As part of the InCharge deployment and configuration process, you will need to modify certain files. User modifiable files include InCharge startup scripts, 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, original versions of Global Manager configuration files are installed to **BASEDIR**/*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 **BASEDIR**/*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. 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*. For example, to edit the configuration file for the Global Manager, you invoke the *sm_edit* utility as follows:

```
% BASEDIR/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*.

InCharge SNMP Trap Adapter

The InCharge SNMP Trap Adapter (Receiver) collects and parses SNMP traps and generates notifications to the Global Manager based on the contents of the traps. The trap receiver uses the *trap_mgr.conf* configuration file to translate incoming traps into notifications that are sent to the Global Manager. You need to use `sm_edit` to edit *trap_mgr.conf* (found at **BASEDIR**/*smarts/conf/icoi*) to define the traps that you want to send to the Global Manager. For situations requiring more advanced trap processing, an ASL script can be called for specific traps.

The *trap_mgr.conf* file contains several sections.

- The beginning of the file includes comments regarding the syntax of the file.
- The first line of code sets the `BATCH_NOTIFY_INTERVAL`. For more information about this interval, refer to [SNMP Trap Batching](#) on page 33.
- A default section, which lists default values for the fields. If, for a given trap, no value is set for a field, the value from the default section is used. This section begins with `BEGIN_DEFAULTS` and ends with `END_DEFAULTS`.

- Trap definition section(s), which define mappings of information for specific traps or groups of traps. There can be one or more trap definition sections. Each section begins with BEGIN_TRAP and specifies the trap mapping by OID, generic trap number, and specific trap number. Each trap definition ends with END_TRAP. For more information, refer to [Editing the trap_mgr.conf File](#) on page 18.

Simple Sample SNMP Trap

Figure 3 shows an example of a simple trap and how values in the trap might populate fields in a Service Assurance notification. In this example, the SNMP trap consists of five values:

- 1 Enterprise OID
- 2 Generic Trap Number
- 3 Specific Trap Number
- 4 System Name

The system name sent by the trap translates into the instance name.

- 5 Varbind 1

Varbind 1 translates into a text message for the notification.

Other fields in the notification get populated from values placed in the configuration file. For example, the ClassName is Host not because the information was sent with the trap, but because an association was made in the configuration file.

SNMP Trap (simplified)

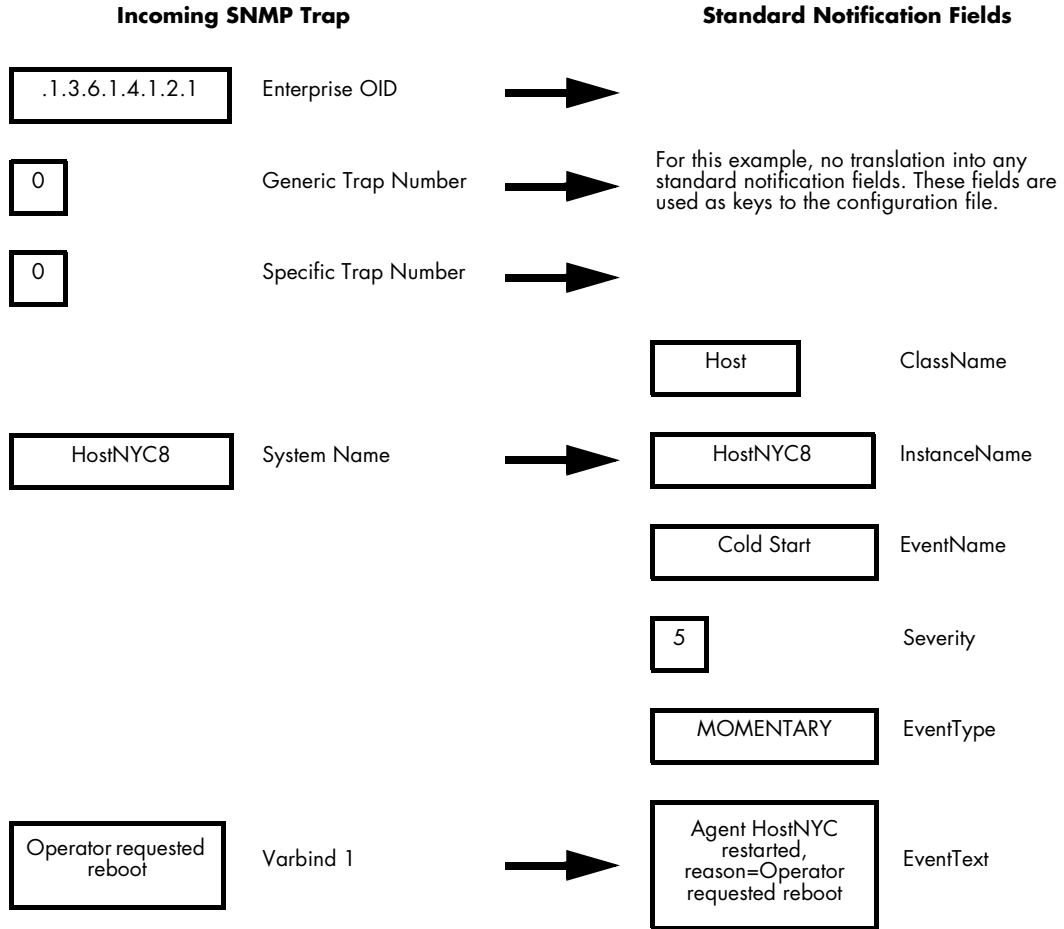


Figure 3: Translation of SNMP Trap to Standard Notification

The following is an example of a portion of the configuration file used by the InCharge SNMP Trap Adapter. This example corresponds to Figure 3. When the trap matches a trap definition (Enterprise OID, generic trap number, and specific trap number), the Adapter Platform populates attributes in a notification based on entries in the trap definition. In the example, the standard notification attribute ClassName is populated with the value "Host."

```

BEGIN_TRAP .1.3.6.1.4.1.2.1 0 0
ClassName:      Host
InstanceName:   $SYS$
EventName:      Cold Start
Severity:       5
EventType:      MOMENTARY
EventText:      Agent $SYS$ restarted, reason=$V1$
END_TRAP
    
```

Configuring the InCharge SNMP Trap Adapter

Table 6 describes the file you use to configure the InCharge SNMP Trap Adapter (Receiver). To configure the Trap Receiver, complete the following steps:

- 1 Define the traps you want forwarded to the Global Manager in your local copy of *trap_mgr.conf* file.
- 2 Set up the proper security between the Adapter Platform and the InCharge SNMP Trap Adapter. Refer to [InCharge SNMP Trap Adapter Security](#) on page 10.
- 3 Verify the port setting for the Trap Receiver. Refer to [Verifying the SNMP Trap Adapter Port Setting](#) on page 32.
- 4 Configure the InCharge SNMP Trap Adapter to start and stop automatically.
- 5 Start the InCharge SNMP Trap Adapter. Refer to [Starting and Stopping the SNMP Trap Adapter](#) on page 33

DIRECTORY UNDER BASEDIR/	FILE NAME(S)	USER EDITABLE	DESCRIPTION
<i>smarts/conf/icoi</i>	<i>trap_mgr.conf</i>	Yes	Used to map incoming SNMP traps to notifications, and to configure notification batch parameters.

Table 6: InCharge SNMP Trap Adapter Configuration and Script Files

Editing the trap_mgr.conf File

Use `sm_edit` to edit the InCharge SNMP Trap Adapter configuration file, `trap_mgr.conf` and to save a local copy to **BASEDIR**/*smarts/local/conf/icoi*. For information about the `sm_edit` utility, see [The sm_edit Utility](#) on page 13.

There is a default section in the trap configuration file that the InCharge SNMP Trap Adapter uses to define the parameters that will be used for all traps. There are also subsequent sample trap definition sections in the file that you can use, alter, or delete.

Default Trap Parameter Section

The format of the default section is as follows.

```
BEGIN_DEFAULTS
  ClassName:          SNMPTrap
  InstanceName:      $$SYS$
  EventName:         $$E$ $$N$ $$S$
  Severity:          2
  EventText:         Varbinds: $$V*$$
  Expiration:        7200
  State:             NOTIFY
  InMaintenance:    FALSE
  ClearOnAcknowledge: TRUE
  EventType:         MOMENTARY
  SysNameOrAddr:    $$A$
  UnknownAgent:     IGNORE
  LogFile:          NONE
END_DEFAULTS
```

Note: It is important that you neither delete nor move the BEGIN_DEFAULT section of this file. This section defines what parameters will be sent for each trap if not specified in subsequent trap definition section(s) of the file. Also, the Map and Aggregate fields are not supported in the Default Trap Parameter section.

The default section demonstrates that you can use variables in descriptions. Of particular note is the use of the varbind variable with an asterisk for the EventText parameter. In this case, EventText stores the value of all of the varbinds associated with a trap. By defining LogFile, the InCharge SNMP Trap Adapter logs all of the traps unless a trap's LogFile is defined as NONE.

Trap Definition Sections

The configuration file contains multiple trap definitions. Each definition specifies what set of traps are processed by the trap definition and what notification values are set for those traps. Incoming SNMP traps are matched against trap definitions in a configuration file.

The beginning of each trap definition starts with BEGIN_TRAP. Each incoming trap is identified by three fields following BEGIN_TRAP. These fields are: Enterprise OID, generic trap number, and specific trap number. The format of the first line is:

```
BEGIN_TRAP <enterprise> <generic_trap> <specific_trap>
```

Each of the fields need to be separated by either a space or a tab. The first field contains the Enterprise OID. The first six numeric values in this field are expected to conform to either the standard Enterprise (vendor) prefix of 1.3.6.1.4.1 or the standard MIB-II (generic) prefix of 1.3.6.1.2.1. However, the InCharge SNMP Trap Adapter does not restrict you to these numeric values. The first field can be any valid OID number or the string 'any' (or *).

You can specify a range or use wildcards in the range. For example, you can specify a range as .1.3.6.1.4.1.<4-10> where the last numeric value can be within the range of 4 through 10. In the example 1.3.6.1.4.1.* the "*" character matches any number for the last OID numeric value.

Additionally, you can list multiple OIDs at the beginning of each trap definition. Specifying multiple OIDs allows the same trap definition to be used for multiple traps. The following example demonstrates listing multiple OIDs, specifying a numeric value range, and using a wildcard:

```
BEGIN_TRAP .1.3.6.1.4.1.1.2.<23-40> 6 1 .1.3.6.1.4.1.6.* 6 1
```

SNMP Trap Parameters

The trap definition follows the BEGIN_TRAP line. The trap definition is the set of values for specified notification attributes that are placed in the Service Assurance notification. You can define these values in any order. The syntax of a line from the trap definition is:

```
<notification_attribute>: <value>
```

Table 7 shows the complete list of parameters to use in a trap definition and in the BEGIN_TRAP line. Each trap definition ends with END_TRAP.

Note: If no parameter values are set within a trap definition BEGIN_TRAP section, the values set in the BEGIN_DEFAULT section are used.

CONFIGURATION FILE PARAMETER	CORRESPONDING STANDARD NOTIFICATION ATTRIBUTE OR DESCRIPTION	VALID VALUES
ASL	For advanced SNMP Trap Integration, this defines the ASL rule set file used to perform additional processing, which may include additional variable substitution. The ASL rule set must be located in BASEDIR /smarts/local/rules/icoi_trapd. (See <i>my_trap-rules.asl</i> for a sample rule set.) The ASL processing overwrites any values set in the trap definition section.	File name
Aggregate	<p>For advanced SNMP Trap Integration, this defines the mapping of a trap definition to an aggregate notification. One or more component notifications comprise an aggregate notification.</p> <p>For details and examples, see Using the Trap Adapter Aggregate Parameter on page 28.</p> <p>Note that the Aggregate field is not supported in the Default Section of the file.</p>	<p>Special</p> <p>EventName:<string> This is the name of the aggregate notification and is required.</p> <p>ElementName:<object-handle></p> <p>An object-handle identifies the InstanceName and the CreationClassName of an element.</p> <p>This is the name of the topology element where the aggregate is defined. This field is optional. If this field is not defined, the ElementClassName and ElementName of the trap definition are used, if available. Otherwise, the ClassName and InstanceName of the trap definition are used.</p> <p>If you define this field and specify an object-handle that does not exist in the topology, then the aggregate is not created.</p> <p>EventText:<string> This is the description of the aggregate notification (event). This field is optional.</p>

CONFIGURATION FILE PARAMETER	CORRESPONDING STANDARD NOTIFICATION ATTRIBUTE OR DESCRIPTION	VALID VALUES
Category	Category	Any string
ClassName	ClassName and DisplayClassName	Any string
ClearOnAcknowledge	ClearOnAcknowledge	TRUE or FALSE
ElementClassName	ElementClassName	Valid Class Name in InCharge topology
ElementName	ElementName	Valid Instance Name in InCharge topology. If you set the UnknownAgent to CREATE, then the instance does not have to previously exist.
EventName	EventName (describing the notification) and the EventDisplayName	Any string
EventText	EventText	Any string
EventType	EventType	MOMENTARY or DURABLE
Expiration	Defines the expiration time in seconds for the notification. Zero indicates that the notification will not expire. 7200 is the default value. If you use zero, you should use some method to eventually clear the notification. For example, you can configure the InCharge SNMP Trap Adapter to send clear state notifications or set the ClearOnAcknowledge parameter to TRUE.	Integer
InMaintenance	InMaintenance	TRUE or FALSE
InstanceName	InstanceName (of the object associated with the notification) and InstanceDisplayName	Any string
LogFile	The name of the file used by the InCharge SNMP Trap Adapter to log information for this trap. If this parameter is 'NONE' or not defined, no information is logged. If this parameter is not defined, the default value is used.	File name

CONFIGURATION FILE PARAMETER	CORRESPONDING STANDARD NOTIFICATION ATTRIBUTE OR DESCRIPTION	VALID VALUES
Map	Map is a special field that enables you to map varbinds to one or more printable strings. See Using the Map Parameter on page 25 and Using the Map Parameter With Tags on page 26 for more information and examples of how to map varbind values. Note that the Map field is not supported in the Default Section of the file.	Special
Severity	Severity	Any integer from 1 to 5
State	Describes the state of the notification.	NOTIFY or CLEAR
SysNameOrAddr	A valid value describes either the system name or address of the entity that sent the trap. Values for this parameter override values in ElementClassName and ElementName.	Any string
UnknownAgent	Describes whether Open Integration should ignore traps related to unknown topology elements or create elements for the traps.	CREATE or IGNORE
UserDefined1	UserDefined1	Any string
UserDefined2	UserDefined2	Any string
UserDefined3	UserDefined3	Any string
UserDefined4	UserDefined4	Any string
UserDefined5	UserDefined5	Any string
UserDefined6	UserDefined6	Any string
UserDefined7	UserDefined7	Any string
UserDefined8	UserDefined8	Any string
UserDefined9	UserDefined9	Any string
UserDefined10	UserDefined10	Any string

Table 7: Trap Definition Parameters

In Figure 3 and the corresponding trap definition, certain values in the SNMP trap were used as attribute values in the notification. This is accomplished through the use of variable substitution. SNMP variable bindings (varbinds) are placeholders for information common to standard SNMP traps and can be assigned to InCharge notification attributes within the trap definitions in the *trap_mgr.conf* file. The following are two lines taken from the trap definition based on Figure 3 that use variable substitution:

```
InstanceName:$SYS$  
EventText:Agent $SYS$ restarted, reason=$V1$
```

The attribute InstanceName is populated with the value of a variable *\$SYS\$*. The contents of this variable comes from the incoming trap. In this example, *\$SYS\$* is the name of the system where the trap originated and *\$V1\$* represents the value of the first varbind. Table 8 describes the variables available for basic SNMP trap integration. When specifying a variable, it must be enclosed by dollar signs (\$).

VARIABLE	DESCRIPTION
\$E\$	Enterprise OID of the SNMP Trap
\$T\$	Timestamp of the SNMP Trap
\$A\$	Address of the agent sending the trap
\$C\$	Community string of the SNMP Trap
\$SYS\$	System where the SNMP trap originated
\$N\$	Generic trap number of the SNMP trap
\$S\$	Specific trap number of the SNMP Trap
\$V<n>[<tag>]\$	Value of the <n>th varbind. The varbind may contain one of the following data types: integer, bit-string, octet-string, IP address, counter, gauge, unsigned integer, time ticks, counter 64, obj id, opaque, or null. Optionally, you can use a text string to tag the value of the <n>th varbind for multiple mappings. or \$V<*>\$
	Also can use an asterisk <*> to specify all varbinds.

Table 8: InCharge SNMP Trap Adapter Variables

Example of Set and Clear Notification

The following example illustrates how to configure the notification and clearing of an event. One trap creates a state of NOTIFY, while the other a state of CLEAR:

```
BEGIN_TRAP .1.3.6.1.4.1.10.1.9.5.1 6 1
ClassName:      Port
InstanceName:   $SYS$ PORT $V1$
EventName:      lerAlarmOn
Severity:       3
EventText:      This is a longer text message
EventType:      DURABLE
Category:       Performance
State:          NOTIFY
END_TRAP

BEGIN_TRAP .1.3.6.1.4.1.10.1.9.5.1 6 2
ClassName:      Port
InstanceName:   $SYS$ PORT $V1$
EventName:      lerAlarmOn
State:          CLEAR
END_TRAP
```

Notifications are uniquely identified using the three attributes: `ClassName`, `InstanceName`, and `EventName`. In the example, the second trap definition will clear an event created with the first trap definition when the values of the three key fields are identical.

Using the Map Parameter

The format of the Map parameter is:

```
Map:      {
          V<n>
            <value_a>=<string>
            <value_b>=<string>
          }
          {
          V<n>
            <value_c>=<string>
            <value_d>=<string>
          }
```

Note: You cannot specify a variable within the definition of a Map parameter.

The following example shows how the Map parameter could be used to substitute text for two different varbind values:

```
BEGIN_TRAP .1.3.6.1.4.1.10 6 0
  ClassName:          Host
  InstanceName:      $SYS$
  EventName:         Coldstart
  Severity:          5
  EventText:         Reason - $V1$
  Expiration:        30
  State:             NOTIFY
  UserDefined1:      Community String: $C$
  ClearOnAcknowledge: TRUE
  Map:               {
                    V1
                    1=UP
                    2=DOWN
                    }
  EventType:         DURABLE
  UnknownAgent:      CREATE
  ElementClass:      Router
  ElementInstance:   $A$
  LogFile:           Coldstart.log
END_TRAP
```

In the example, the Map parameter describes how the enumeration values of the first varbind translate into printable strings. These printable strings are substituted as values in EventText. If a SNMP trap had a value of one for its first varbind, the phrase, "Reason - UP" would be placed in EventText.

If a map is not defined for a specific varbind, the InCharge SNMP Trap Adapter uses the varbind's integer value. In the example, the integer "1" would be used instead of the text "UP," so the phrase, "Reason -1" would be placed in EventText.

Using the Map Parameter With Tags

Each incoming trap is identified by three fields: Enterprise OID, generic trap number, and specific trap number. Some SNMP agents send traps with the same combination of OID, generic, and specific trap numbers but for different reasons. When this occurs, the only way to differentiate the meaning of the trap is to examine the varbinds. You can use the \$V<n>-<tag>\$ variable to map varbinds to different text strings.

The format of the Map parameter when using tags is:

```
Map:      {
          V<n-tag1>
            <value_a>=<string1>
            <value_b>=<string2>
            default=<string3>
          }
          {
          V<n-tag2>
            <value_a>=<string4>
            <value_b>=<string5>
            default=
          }
          {
          V<n-tag3>
            <value_a>=<string6>
            <value_b>=<string7>
            default=
          }
        }
```

The following example defines two $V<n>-[<tag>]$ variables, one for substitution of ClassName and another for substitution of EventName:

```
BEGIN_TRAP .1.3.6.1.4.1.546.1.1 6
  ClassName:          $V2-class$
  EventName:          $V2-event$
.
.
.
```

The mapping defines when to map the appropriate tags to the varbinds.

```
.
.
.
Map:      {
          V2-class
            .1.3.6.1.4.1.546.1.1.7.9.30.0=Memory
            .1.3.6.1.4.1.546.1.1.7.9.2.0=Processor
            default=Host
          }
          {
          V2-event
            .1.3.6.1.4.1.546.1.1.7.9.30.0=NotEnoughMem
            .1.3.6.1.4.1.546.1.1.7.9.2.0=HighUtilization
            default=TrapReceived
          }
        }
```

If the incoming trap has a Varbind 2 value of .1.3.6.1.4.1.546.1.1.7.9.30.0, then Memory is used for the ClassName and NotEnoughMem is used for the EventName. If the incoming trap has a Varbind 2 value of .1.3.6.1.4.1.546.1.1.7.9.2.0, then Processor is used for the ClassName and HighUtilization is used for the EventName. The default, Host, is used for the ClassName and the default, TrapReceived, is used for the EventName for incoming traps with any other Varbind 2 values. If a default is not specified, the original value for the Varbind 2 is returned.

Using the Trap Adapter Aggregate Parameter

An aggregate notification is a notification that is composed of one or more component events. When one of the component events occur, the aggregate notification is notified. Creating an aggregate notification provides several benefits:

- Aggregation compresses multiple component events into a single aggregate notification.
- Aggregation helps you organize related traps into meaningful categories.
- Aggregation provides a method by which the details of the component events can be displayed to an operator. The component events of an aggregate notification are displayed in the Aggregate tab of the Notification Properties window of the Global Console.

You use the aggregate in your trap definition to create an aggregate notification as well as the aggregate's component event(s). You are required to define an EventName for each aggregate or component event. Also, since all notifications are uniquely identified by EventName and the ClassName, and the InstanceName it OccurredOn, values must also be derived for the two latter attributes.

SMARTS recommends that you explicitly define the ElementName:<object-handle> within the Aggregate portion of the trap definition. If you do not explicitly define the ElementName within the Aggregate portion, then the ElementClassName and ElementName of the trap definition are used, if available. Alternatively, if ElementName and ElementClassName are not defined, then the ClassName and InstanceName are used to define the aggregate.

The following example illustrates how to configure two different trap definitions to generate the same aggregate notification. The first trap definition processes an incoming trap that indicates a system is experiencing low memory. The second trap definition processes an incoming trap that indicates that a system's processor is experiencing high utilization. Since both of these traps are indicative of system degradation, if either trap is received, a Service Assurance notification is generated to indicate that the system is degraded.

```
BEGIN_TRAP .1.3.6.1.4.1.546.1.1 6 1
  ClassName:      Memory
  InstanceName:   MEM-$$SYS$
  EventName:      Low
  Aggregate:      {
                  EventName: Degraded
                  ElementName: $$SYS$
                }
  Severity:       1
  Expiratiton:   30
  State:          NOTIFY
  ClearOnAcknowledge: TRUE
END_TRAP

BEGIN_TRAP .1.3.6.1.4.1.546.1.1 6 2
  ClassName:      Processor
  InstanceName:   PRO-$$SYS$
  EventName:      HighUtilization
  Aggregate:      {
                  EventName: Degraded
                  ElementName: $$SYS$
                }
  Severity:       1
  Expiratiton:   30
  State:          NOTIFY
  ClearOnAcknowledge: TRUE
END_TRAP
```

Using the example, if a trap with OID .1.3.6.1.4.1.546.1.1 6 1 is received from Router::CoreRouter1, the component event Memory::MEM-CoreRouter1 Low is generated. Similarly, if a trap with OID .1.3.6.1.4.1.546.1.1 6 2 is also received from the same router, then the component event Processor::PRO-CoreRouter1 HighUtilization is generated. Since the same aggregate is defined for both of these traps, this will also generate the aggregate notification, Router::CoreRouter1 Degraded.

Although all events are propagated to the Service Assurance Global Manager, only the aggregate notification (Router::CoreRouter1 Degraded) will display in the Global Console. This is because the default NotificationList used by the Global Manager is configured to filter out the component (raw trap) events. If desired, the Global Console operator can display the component events from the Notification Properties window by choosing the Aggregate tab.

Advanced SNMP Trap Integration

For more sophisticated SNMP Trap processing, you can define an ASL script in a trap definition that is invoked during the processing of the trap. To define an ASL script, you add the keyword ASL, followed by the ASL script name, to the trap definition. For example:

```
BEGIN_TRAP .1.3.6.1.4.1.9.10.1 0 0
  ClassName: System
  ASL: my-trap-rules.asl
  InstanceName:$SYSS$
  EventName: ColdStart
  Type: MOMENTARY
END_TRAP
```

Notification attribute values are initially determined by the trap definitions. If an attribute value has not been defined by a trap definition, then the value for the attribute will come from the default section. The notification attribute values are exported (made available) from within the ASL script. The ASL script can override (change) default attribute values. After the ASL script completes, the notification is created with those sets of values.

The ASL script relies on two special types of variables: input/output and input-only. Input/output variables correspond to attributes of the notification and the parameters in the configuration file. The input-only variables correspond to the variables used in the configuration file (for example, \$SYSS\$, \$V4\$, and \$N\$).

A DISCARD_TRAP variable is included with the input/output variables. The variable controls whether a given trap message is discarded or processed. The default is "FALSE" which means to process the trap. To override the default and discard the trap, create a rule in the hook script to process the trap, and within this rule set the DISCARD_TRAP to "TRUE."

In order to run the ASL script, you must place it in ***BASEDIR***/*smarts/local/rules/icoi-trapd*.

Example of ASL Script to Set EventText

The following example ASL script extracts the substring "Agent Restarted for Reason:" from the contents of a trap's first varbind and places the remaining string into the notification attribute EventText. In the script, a list of all of the input and output variables as well as the input-only variables precedes the start of the processing (START). Generally, only the variables used by the script need to be declared.

```

/*
 * The first variable binding is a string of the form:
 * "Agent Restarted for Reason: <text to extract>"
 */

// Input/Output variables.

CLASSNAME = "";
INSTANCENAME = "";
EVENTNAME = "";
SEVERITY = "";
EVENTTEXT = "";
CATEGORY = "";
EXPIRATION = "";
STATE = "";
INMAINTENANCE = "";
CLEARONACKNOWLEDGE = "";
EVENTTYPE = "";
ELEMENTCLASSNAME="";
ELEMENTNAME="";
USERDEFINED1 = "";
USERDEFINED2 = "";
USERDEFINED3 = "";
USERDEFINED4 = "";
USERDEFINED5 = "";
USERDEFINED6 = "";
USERDEFINED7 = "";
USERDEFINED8 = "";
USERDEFINED9 = "";
USERDEFINED10 = "";
DISCARD_TRAP = "FALSE";

// Input Variables

TIMESTAMP = "0";
IPADDRESS = "";
ENTERPRISE = "";
GENERIC = "9999";
SPECIFIC = "9999";
V1 = "";
V2 = "";

```

```
V3 = "";  
V4 = "";  
V5 = "";  
V6 = "";  
V7 = "";  
V8 = "";  
V9 = "";  
V10 = "";  
V11 = "";  
V12 = "";  
V13 = "";  
V14 = "";  
V15 = "";  
V16 = "";  
V17 = "";  
V18 = "";  
V19 = "";  
V20 = "";
```

```
START {  
    PARSE_EVENT_TEXT  
}  
  
PARSE_EVENT_TEXT {  
    input = V1;  
    .. "Reason:" EVENTTEXT: rep(word) eol  
}
```

The processing of the trap's first varbind occurs in the rule `PARSE_EVENT_TEXT`. The input to the rule is the contents of the first varbind. The line after the input reads all text up to and including "Reason:". The rest of the string is assigned to the variable `EVENTTEXT`. Any change to the value of the input/output variables automatically gets placed in the corresponding attribute in the translated InCharge notification. For more information about ASL, refer to *InCharge Building Adapters with ASL*.

Verifying the SNMP Trap Adapter Port Setting

The default value for the SNMP Trap Adapter (trap receiver) port is 162. If you need to change this default value, refer to the *InCharge System Administration Guide* for details about how to modify an application's runtime parameters.

SNMP Trap Batching

By default, the InCharge SNMP Trap Adapter immediately converts traps it receives (that match a trap definition) into new Service Assurance notifications or into updates to existing Service Assurance notifications and forwards them on to the Global Manager. SNMP trap batching refers to a process where the Adapter Platform server waits for a specified period of time before forwarding re-notifications (updated notifications) to the Global Manager.

In cases of high frequency of traps, you can use batching to improve performance of clients processing the converted Service Assurance notifications. You configure batching by editing your local copy of *trap_mgr.conf* file so that traps re-notifying an event are held for a specified period of time. Then, once that time is exceeded, only the most recent trap of those bearing the same notification name is sent to the Global Manager.

To set the batch parameter, use `sm_edit` to open ***BASEDIR/smarts/local/conf/icoi/trap_mgr.conf***, and enter the period of time in seconds you want the trap receiver to wait before forwarding re-notifications to the Global Manager. The following text shows the section where you set the batch parameter. By default, the batch setting is 10 seconds. To disable batching, specify zero (0).

```
# This interval (in seconds) will be used to batch updates to
# notifications. In case, where a high frequency of
# notifications occur, batching will improve performance.
# Setting this interval to 0 will disable batching.
```

```
BATCH_NOTIFY_INTERVAL = 10
```

Starting and Stopping the SNMP Trap Adapter

If you installed the SNMP Trap Adapter as a service, it automatically starts when the system starts up. The following instructions describe how to use the `sm_service` utility to manually start and stop the SNMP Trap Adapter.

Issue one of the following from the command line:

```
% BASEDIR/smarts/bin/sm_service start ic-trapd-receiver
```

or

```
% BASEDIR/smarts/bin/sm_service stop ic-trapd-receiver
```

Note: The sm_service utility is operating system independent and works the same way on both UNIX and Windows operating systems.

For more information about the SNMP Trap Adapter's default start-up options or how to modify them, refer to the *InCharge System Administration Guide*.

On Windows, if the SNMP Trap Adapter was installed as a service, you can also start and stop the service using the Control Panel Administrative Tools.

InCharge Syslog Adapter

The InCharge Syslog Adapter tails or parses the contents of any system log file and generates notifications to the Global Manager based on the file contents.

You start the adapter by invoking the `sm_service` command `start ic-syslog-adapter`. First the `syslog_mgr.asl` file parses each syslog message and populates the input variables. Then the `my_hook_syslog.asl` gets executed. This script first populates the notification attribute output variables using the default values. It then uses the `MODIFY_ATTRIBUTES` rule to set additional attributes defined and potentially modify (overriding) any default attribute values. This is how the InCharge notification gets created by the InCharge Syslog Adapter.

Before you configure the InCharge Syslog Adapter, identify the location of the `SYSFILE` you want the adapter to tail or parse and ensure that `sm_service` install command line for the `ic-syslog-adapter` identifies this location. You must also ensure that the file format of the Syslog exactly matches the format described in the following sections.

Configuring the InCharge Syslog Adapter

Table 9 describes the files that you need to use when configuring the InCharge Syslog Adapter. If you edit any of these files, you must use the `sm_edit` utility. The utility will save the local copies to the appropriate InCharge subdirectories under the **BASEDIR**/`smarts/local` directory. For information about the `sm_edit` utility, see [The `sm_edit` Utility](#) on page 13.

DIRECTORY UNDER BASEDIR/	FILE NAME(S)	USER EDITABLE	DESCRIPTION
<i>smarts/rules/icoi-syslog/</i>	<i>my_hook_syslog.asl</i>	Yes	Basic template for processing a syslog file.
<i>smarts/rules/icoi-syslog/</i>	<i>syslog_mgr.asl</i>	Yes	Rule set for parsing each syslog message.

Table 9: InCharge Syslog Adapter Configuration and Script Files

The InCharge Syslog Adapter creates events by parsing the contents of syslog files. You can use it to parse the contents of any text file with entries of the format:

```
month day time hostName applicationName [process_id]:text_message
```

If the format of your syslog file is different from the above format, you can edit *my_hook_syslog.asl* and *syslog_mgr.asl* to pass the entries accordingly.

The InCharge Syslog Adapter can parse the contents of a file or it can tail a file. When the InCharge Syslog Adapter tails a file, it skips the existing content and uses only content added to the file while the adapter is running.

Note: The `process_id` parameter is optional when parsing the contents of syslog files.

The Adapter Platform includes a basic template for processing a syslog file. This file is **BASEDIR**/*smarts/rules/icoi-syslog/my_hook_syslog.asl*.

After ensuring that the Adapter Platform Server and the Global Manager are up and running, complete the following procedures to configure the InCharge Syslog Adapter:

- 1** Check the location of the Syslog file to be sure it is appropriately placed for your operating system.
- 2** Change the parameters in the local copy of *my_hook_syslog.asl* to match your needs.
- 3** Start the InCharge Syslog Adapter.

Syslog File Location

The `sm_service` install command line for the InCharge Syslog Adapter, *ic-syslog-adapter*, is stored in the `sm_service` data base. By default, the `SYSDIR` parameter in this script specifies the location as `/var/log/syslog`. Edit the command line to change the location of the Syslog file if necessary.

Editing the my_hook_syslog.asl

You can use the Adapter Scripting Language (ASL) to modify the functionality of the local copy of *my_hook_syslog.asl* or to create a new file. The basic components of a custom processing file for the InCharge Syslog Adapter are explained below:

```
debug = FALSE;
ASLNAME = " ".getRuleFileName ().": ";

DISCARD = "TRUE";

CLEAR_SYSLOG = "FALSE";

BATCH_NOTIFY_INTERVAL = 10;

// Output variables : This section has all default settings.

CLASSNAME = "Syslog";
INSTANCENAME = "";
EVENTNAME = "";
SEVERITY = "2";
EVENTTEXT = "";
CATEGORY = "";
EXPIRATION = "7200";

STATE = "";
INMAINTENANCE = "FALSE";
CLEARONACKNOWLEDGE = "TRUE";
EVENTTYPE = "";
USERDEFINED1 = "";
USERDEFINED2 = "";
USERDEFINED3 = "";
USERDEFINED4 = "";
USERDEFINED5 = "";
USERDEFINED6 = "";
USERDEFINED7 = "";
USERDEFINED8 = "";
USERDEFINED9 = "";
USERDEFINED10 = "";
```

```
ELEMENTCLASSNAME = "";  
ELEMENTNAME = "";  
SYSNAMEORADDR = "";  
UNKNOWNAGENT = "IGNORE";  
LOGFILE = "NONE";
```

```
// Aggregate Section :
```

```
AGG_EVENTNAME = "";  
AGG_ELEMENTNAME = "";  
AGG_EVENTTEXT = "";
```

The values of the output variables populate the attributes of the standard notification created when the syslog message is imported. The variable names correspond directly to the standard notification's attribute names.

The InCharge Syslog Adapter populates these variables when the syslog entry is parsed.

```
// Input Variables
```

```
SYSLOGTIME = "";  
HOST = "";  
APPLICATION_NAME = "";  
PROCESS_ID = "";  
MESSAGE = "";
```

The START rule takes the text parsed from the syslog entry as input, prints a message, calls three other rules, prints another message, and exits after the processing is complete.

```
START {  
    input=MESSAGE;  
do {  
if (debug) {print(time()).ASLNAME."SYSLOGTIME =" .SYSLOGTIME);} }  
if (debug) {print(time()).ASLNAME."HOST =" .HOST);} }  
if (debug) {print(time()).ASLNAME."APPLICATION_NAME  
=" .APPLICATION_NAME);} }  
if (debug) {print(time()).ASLNAME."PROCESS_ID =" .PROCESS_ID);} }  
if (debug) {print(time()).ASLNAME."MESSAGE =" .MESSAGE);} }  
}  
    PARSE_MESSAGE  
    MODIFY_ATTRIBUTES  
    CUSTOM_RULE?  
} do {  
    if (debug) { print(time()).ASLNAME."Done with  
my_hook_syslog.asl ");} }  
    return;  
}
```

The CUSTOM rule is an example of a rule which performs more customizations. In this case, it saves a prefix and a message description.

```
CUSTOM_RULE {
    unusedPrefix:rep(notany(":")) ":"          /* consume
chars up to : */
    msgDescription:rep(word) eol
} do {
    if (debug) { print(time()).ASLNAME."Executing
CUSTOM_RULE");}
}
```

This PARSE_MESSAGE rule saves only the first 30 characters.

```
PARSE_MESSAGE {
} do {
    // Use a slice of 30 characters as part of EVENTNAME
    slice = substring(MESSAGE, 0, 30);
}
```

The MODIFY_ATTRIBUTES rule assigns values to the notification created from the syslog entry. The value of InstanceName is composed of HOST, APPLICATION_NAME, and PROCESS_ID. These are values parsed from the syslog entry.

```
/*
 * MODIFY_ATTRIBUTES Rule:
 * All your customizations are done here. You can use all
 * the Syslog input variables wherever you want them assigned
 * to ICS_Notification attributes.
 * ----- */

MODIFY_ATTRIBUTES {
} do {

    DISCARD = "TRUE";
    CLEAR_SYSLOG = "FALSE";
    BATCH_NOTIFY_INTERVAL = 10;
    CLASSNAME = "Syslog" ? LOG;
    INSTANCENAME = HOST."_"APPLICATION_NAME."_"PROCESS_ID ? LOG;
    EVENTNAME = slice ? LOG;
    SEVERITY = "2" ? LOG;
    EVENTTEXT = MESSAGE ? LOG;
    CATEGORY = "" ? LOG;
    EXPIRATION = "7200" ? LOG;
    STATE = "NOTIFY" ? LOG;
    INMAINTENANCE = "FALSE" ? LOG;
    CLEARONACKNOWLEDGE = "TRUE" ? LOG;
    EVENTTYPE = "DURABLE" ? LOG;
```

```
USERDEFINED1 = "" ? LOG;
USERDEFINED2 = "" ? LOG;
USERDEFINED3 = "" ? LOG;
USERDEFINED4 = "" ? LOG;
USERDEFINED5 = "" ? LOG;
USERDEFINED6 = "" ? LOG;
USERDEFINED7 = "" ? LOG;
USERDEFINED8 = "" ? LOG;
USERDEFINED9 = "" ? LOG;
USERDEFINED10 = "" ? LOG;

ELEMENTCLASSNAME = "Host";
ELEMENTNAME = HOST;
SYSNAMEORADDR = HOST;
UNKNOWNAGENT = "CREATE";
LOGFILE = "NONE";

AGG_EVENTNAME = "AggEvent-".INSTANCENAME;
AGG_ELEMENTNAME = HOST;
AGG_EVENTTEXT = "This is an Aggregate Test"

}
```

You can optionally add logic that does compares on the input variables and sets the output variables based on them.

Whenever you modify the hook script file, you must restart the adapter for the changes to take effect.

InCharge Syslog Adapter Parameters

Table 10 shows the complete list of parameters that can be defined for a notification forwarded to the Global Manager by the InCharge Syslog Adapter.

Note: If no parameter values are set within the MODIFY_ATTRIBUTES rule, then the values set in the Output variables section are used.

PARAMETER	CORRESPONDING STANDARD NOTIFICATION ATTRIBUTE OR DESCRIPTION	VALID VALUES
AGG_EVENTNAME	This is the event name of the aggregate notification and is required if you are creating an aggregate.	<string>
AGG_ELEMENTNAME	This is the name of the topology element where the aggregate event occurs. This field is optional. If this field is not defined, the ElementClassName and ElementName of the trap definition are used, if available. Otherwise, the ClassName and InstanceName are used. If you define this field and specify an object-handle that does not exist in the topology, then the aggregate is not created.	<object-handle>
AGG_EVENTTEXT	This is the description of the aggregate notification (event). This field is optional.	<string>
CATEGORY	Category	Any string
CLASSNAME	ClassName and DisplayClassName	Any string
CLEARONACKNOWLEDGE	ClearOnAcknowledge	TRUE or FALSE
CLEAR_SYSLOG	Controls whether a given syslog message clears an already existing notification or results in a notify. The default is FALSE, all syslog messages result in notifications. All syslog messages result in notifications unless the syslog messages are processed by subsequent MODIFY_ATTRIBUTES rules, which override the default.	TRUE or FALSE
DISCARD	Controls whether a given syslog message is discarded or processed. The default is TRUE, all syslog messages are discarded. To override the default and process a syslog message, create a rule to process this syslog message. Within this rule, DISCARD should be set to FALSE.	TRUE or FALSE
ELEMENTCLASSNAME	ElementClassName	Valid Class Name in InCharge topology

PARAMETER	CORRESPONDING STANDARD NOTIFICATION ATTRIBUTE OR DESCRIPTION	VALID VALUES
ELEMENTINSTANCENAME	ElementName	Valid Instance Name in InCharge topology
EVENTNAME	EventName (describing the notification) and the EventDisplayName	Any string
EVENTTEXT	EventText	Any string
EVENTTYPE	EventType	MOMENTARY or DURABLE
EXPIRATION	<p>Defines the expiration time in seconds for the notification. Zero indicates that the notification will not expire. 7200 is the default value.</p> <p>If you use zero, you should use some method to eventually clear the notification. For example, you can configure the Syslog Adapter to send clear state notifications or set the ClearOnAcknowledge parameter to TRUE.</p>	Integer
INMAINTENANCE	InMaintenance	TRUE or FALSE
INSTANCENAME	InstanceName (of the object associated with the notification) and InstanceDisplayName	Any string
LOGFILE	The name of the file used by the SNMP Trap Receiver to log information for this trap. If this parameter is 'NONE' or not defined, no information is logged.	File name
SEVERITY	Severity	Any integer from 1 to 5
STATE	Describes the state of the notification. This option is deprecated; its function is replaced by CLEAR_SYSLOG.	NOTIFY or CLEAR
SYSNAMEORADDR	A valid value describes either the system name or address of the entity that sent the trap. Values for this parameter override values in ElementClass and ElementInstance.	Any string
UNKNOWNAGENT	Describes whether Open Integration should ignore traps related to unknown topology elements or create elements for the traps.	CREATE or IGNORE

PARAMETER	CORRESPONDING STANDARD NOTIFICATION ATTRIBUTE OR DESCRIPTION	VALID VALUES
USERDEFINED1	UserDefined1	Any string
USERDEFINED2	UserDefined2	Any string
USERDEFINED3	UserDefined3	Any string
USERDEFINED4	UserDefined4	Any string
USERDEFINED5	UserDefined5	Any string
USERDEFINED6	UserDefined6	Any string
USERDEFINED7	UserDefined7	Any string
USERDEFINED8	UserDefined8	Any string
USERDEFINED9	UserDefined9	Any string
USERDEFINED10	UserDefined10	Any string

Table 10: InCharge Syslog Adapter Parameters

Using the Syslog Aggregate Parameters

You configure aggregates in the Aggregate Section of the local copy of *my_hook_syslog.asl* rule set, located in **BASEDIR**/*smarts/local/rules/icoi-syslog* directory. For general information regarding Aggregates, refer to the [Using the Trap Adapter Aggregate Parameter](#) on page 28.

The following is an example that illustrates how to use the aggregate parameter with the Syslog Adapter.

```

/*
 * my_hook_syslog.asl - Hook adapter for any syslog related
 customizations.
 *
 * Copyright (C) 1997, System Management ARTS (SMARTS)
 * All Rights Reserved
 */

debug = FALSE;
ASLNAME = " ".getRuleFileName().": ";

DISCARD = "TRUE";

CLEAR_SYSLOG = "FALSE";

```

```
/*
 * This interval (in seconds) will be used to batch updates to
 * notifications. In case, where a high frequency of
 * notifications occur, batching will improve performance.
 * Setting this interval to 0, will disable batching.
 */

BATCH_NOTIFY_INTERVAL = 10;

CLASSNAME = "Syslog";
INSTANCENAME = "";
EVENTNAME = "";
SEVERITY = "2";
EVENTTEXT = "";
CATEGORY = "";
EXPIRATION = "300";
STATE = "";
INMAINTENANCE = "FALSE";
CLEARONACKNOWLEDGE = "TRUE";
EVENTTYPE = "";
USERDEFINED1 = "";
USERDEFINED2 = "";
USERDEFINED3 = "";
USERDEFINED4 = "";
USERDEFINED5 = "";
USERDEFINED6 = "";
USERDEFINED7 = "";
USERDEFINED8 = "";
USERDEFINED9 = "";
USERDEFINED10 = "";

ELEMENTCLASSNAME = "";
ELEMENTNAME = "";
SYSNAMEORADDR = "";
UNKNOWNAGENT = "IGNORE";
LOGFILE = "NONE";

/* Need to Declare these, if you want Aggregates
 * ----- */
AGG_EVENTNAME = "";
AGG_ELEMENTNAME = "";
AGG_EVENTTEXT = "";

/*
 * Input Variables: Following are the variable declarations,
 * which hold the Syslog parsed values.
 * -----
 */
SYSLOGTIME = "";
HOST = "";
```

```

APPLICATION_NAME = "";
PROCESS_ID = "";
MESSAGE = "";

if (debug) { print(time().ASLNAME."Activated"); }

/*
 * Start Rule
 * ----- */

START {
    input=MESSAGE;
    MODIFY_ATTRIBUTES
    CREATE_AGGREGATE
} do {
    if (debug) { print(time().ASLNAME."Done with
my_hook_syslog.asl ");}
    return;
}

CREATE_AGGREGATE {
} do {
    // If you see strings "CPU" and "HighUtilization" in
    // the syslog
    // message, then generate and aggregate.
    // -----
    if (glob("*CPU*",MESSAGE) &&
glob("*HighUtilization*",MESSAGE)) {

        AGG_EVENTNAME = "Degraded";
        AGG_ELEMENTNAME = HOST;
        AGG_EVENTTEXT = "Host [".HOST."] is Degraded";
    }
}

MODIFY_ATTRIBUTES {
} do {
    CLASSNAME = "Processor" ? LOG;
    INSTANCENAME = "PRO-".HOST ? LOG;
    EVENTNAME = substring(MESSAGE, 0, 30) ? LOG;
    SEVERITY = "2" ? LOG;
    EVENTTEXT = MESSAGE ? LOG;
    CATEGORY = "" ? LOG;
    EXPIRATION = "7200" ? LOG; //PR:6617
    STATE = "NOTIFY" ? LOG;
    INMAINTENANCE = "FALSE" ? LOG;
    CLEARONACKNOWLEDGE = "TRUE" ? LOG;
    EVENTTYPE = "DURABLE" ? LOG;

```

```
    ELEMENTCLASSNAME = "Processor";
    ELEMENTNAME = "PRO-".HOST ? LOG;
    UNKNOWNAGENT = "CREATE";
    LOGFILE = "Processor.log";
}

DEFAULT {
    msg:{.. eol}
} do {
    print(time().ASLNAME."Reached Default rule: ".msg);
    this->clearVariables();
}

/*
 * These variables describe the formatting of this file.  If
 * you don't like the template defaults, feel free to change
 * them here (not in your .emacs file).
 *
 * Local Variables:
 * mode: C++
 * End:
 */
```

Syslog Batching

By default, except if the DISCARD parameter is set to TRUE, the InCharge Syslog Adapter immediately converts syslog messages it receives into new Service Assurance events or into updates to existing Service Assurance events and forwards them on to the Global Manager. Syslog batching refers to a process where the Adapter Platform server waits for a specified period of time before forwarding re-notifications (updated events) to the Global Manager.

In cases of high frequency of syslog messages, you can use batching to improve performance of clients processing the converted Service Assurance events. You configure batching by editing the *my_hook_syslog.asl* file so that re-notification messages are held for a specified period of time. Then, once that time is exceeded, only the most recent message of those bearing the same event name is sent to the Global Manager.

To set the batch parameter, use `sm_edit` to open `my_hook_syslog.asl`, and enter the period of time (in seconds) you want the InCharge Syslog Adapter to wait before forwarding re-notifications to the Global Manager. By default, the batch setting is 10 seconds. To disable batching, specify zero (0).

```
BATCH_NOTIFY_INTERVAL = 10
```

For an example of a script that contains the batch parameter, see [Using the Syslog Aggregate Parameters](#) on page 43.

Starting and Stopping the Syslog Adapter

If you installed the Syslog Adapter as a service, it automatically starts when the system starts up. The following instructions describe how to use the `sm_service` utility to manually start and stop the Syslog Adapter.

Issue one of the following from the command line:

```
% BASEDIR/smarts/bin/sm_service start ic-syslog-adapter
```

or

```
% BASEDIR/smarts/bin/sm_service stop ic-syslog-adapter
```

Note: The `sm_service` utility is operating system independent and works the same way on both UNIX and Windows operating systems.

For more information about the Syslog Adapter's default start-up options or how to modify them, refer to the *InCharge System Administration Guide*.

On Windows, if the Syslog Adapter was installed as a service, you can also start and stop the service using the Control Panel Administrative Tools.

4

Command-Line Interface

The command-line interface (`sm_ems`) is useful for converting information from third-party applications into events. This interface can create and clear events and create basic topology elements. The command-line interface also can update event attributes passed by the Adapter Platform server, and associate events to topology elements. You can use the ASL scripting language in conjunction with the command-line interface for more advanced processing.

Using the Command-Line Interface

Configuration of the command-line interface consists of two parts.

- 1 You need to set up the proper security between the Adapter Platform and the command-line interface.
- 2 You must modify your third-party application to call `sm_ems`.

Command-Line Interface Usage

The third-party application must be modified to call the `sm_ems` command. Consult the third-party application's documentation for more information about configuring the application to call the command-line interface.

The basic format to call the command-line interface is:

```
% sm_ems --server=<server_name> [options ...] <command>
```

Table 11 describes the options.

OPTION	DESCRIPTION
<code>--server=<name></code>	The name of the Adapter Platform (Open Integration) server. This parameter is required.
<code>--broker=<location></code>	The name of the broker, if you override the broker set by the <code>SM_BROKER</code> environment variable.
<code>--system=<nameOrAddr></code>	<p>Specifies the name or IP address of the existing system (Unitary Computer System) you want to associate with this event.</p> <p>The event will automatically be associated with this system in the Adapter Platform topology.</p> <p>The system name is converted to its canonical name using host name lookups (for example, <code>fleet</code> or <code>10.1.2.345</code> might be converted to <code>fleet.smarts.com.</code>).</p> <p>If the system does not exist in the topology, it may be created automatically if the <code>--create-system</code> option is specified.</p>
<code>--create-system</code>	<p>Determines that the unitary computer system should automatically be created if it does not exist in the topology. The Class defaults to <code>Node</code>, however you can optionally use <code>--element-class</code> to specify the class name. This option is deprecated, but retained for backwards compatibility.</p> <p>The <code>--create-element</code> option can be used instead of this option to create any infrastructure element.</p>
<code>--element-class=<ClassName></code>	<p>Determines the <code>ClassName</code> of the entity being created in the InCharge topology, if the element does not already exist.</p> <p>Issue this option with <code>--create-element</code> and also <code>--element-name</code>.</p>
<code>--element-name=<InstanceName></code>	<p>Determines the <code>InstanceName</code> to be used with the <code>--element-class</code> option. Options provided with <code>--element-class</code> and <code>--element-name</code> will be used to create the object.</p> <p>Note: <code>--system</code> should not be used if you are also issuing the <code>--element-class</code> and <code>--element-name</code> options.</p>
<code>--create-element</code>	<p>Determines that the <code>element-class</code> and <code>element-name</code> you specify should automatically be created if it does not exist in the InCharge topology.</p> <p>Use this option to create any infrastructure element, including business elements, application-related elements, and unitary computer systems.</p>

OPTION	DESCRIPTION
<code>--aggregate-element-class=<ClassName></code>	Determines the <code>ClassName</code> for the Aggregate Notification, if one is being created.
<code>--aggregate-element-name=<InstanceName></code>	Determines the <code>InstanceName</code> for the Aggregate Notification, if an aggregate notification is being created (using the <code>--aggregate-event</code> option). Also creates an <code>OccurredOn</code> relationship between the Instance and the Aggregate Event.
<code>--aggregate-event=<AggregateEventName></code>	Determines the <code>EventName</code> of an Aggregate Notification. This is a required option if you are creating an Aggregate.
<code>--audit=<msg></code>	Optional text to include in the description field of the audit log entry created for this action. Note that this option is ignored for the <code>add-audit-log</code> command.
<code>--traceServer</code>	Enables tracing of server communications.

Table 11: Command-Line Interface Options

Command-Line Interface Commands

There are ten different commands to use with `sm_ems`. Along with the command, you need to pass arguments. Table 12 describes these commands. Where possible in the table, the arguments passed with the command are expressed as attributes from the standard notification. Notification attributes are described in [Service Assurance Adapter Platform and Notifications](#) on page 55.

With the exception of `summarize`, all of the `sm_ems` commands use `ClassName`, `InstanceName`, and `EventName`. These three fields are always required because they uniquely identify a standard notification.

COMMAND	DESCRIPTION
<code>acknowledge <ClassName> <InstanceName> <EventName></code>	Changes the notification's attribute, <code>Acknowledged</code> to <code>TRUE</code> . Attribute values modified using this command argument do not get propagated to Global Manager through the Adapter Platform (Open Integration).
<code>add-audit-log <ClassName> <InstanceName> <EventName> <message></code>	Adds an entry to the Adapter Platform's audit log. The entry contains information about the notification as well as a text message (<code><message></code>).
<code>assign <ClassName> <InstanceName> <EventName> <Owner></code>	Changes the notification's ownership. Attribute values modified using this command argument do not get propagated to Global Manager through the Adapter Platform (Open Integration).
<code>clear <ClassName> <InstanceName> <EventName> <SourceDomainName></code>	Clears an occurrence of a notification. In order for this command to work, <code>SourceDomainName</code> must match the notification's attribute.

COMMAND	DESCRIPTION
<pre>notify <ClassName> <InstanceName> <EventName> <SourceDomainName> <EventType> <Clear-Mode> [<attribute>=<value> ...]</pre>	<p>Creates a notification.</p> <p>If you use the notify command to create a notification that already exists, the Count attribute is increased by 1 and any attribute which is not explicitly assigned a new value will retain its previous value.</p> <p>You must use values for each argument through <Clear-Mode>. Clear-Mode indicates how the notification gets cleared. Valid values for Clear-Mode are:</p> <ul style="list-style-type: none"> • source—The source sends a clear. • <number>—The number of seconds before the event expires. • none—The notification should not expire and the source will not send a clear. <p>At the end of this command, you have the option of defining other attributes for the command. Optional attributes are: Category, Certainty, ClassDisplayName, ClearOnAcknowledge, EventDisplayName, EventText, Impact, InMaintenance, InstanceDisplayName, Severity, TroubleTicketID, and UserDefined1 through UserDefined10.</p> <p>In this case, the attribute name must match the standard notification attribute. If you do not explicitly assign values for other attributes, the default attribute values are used. (For a list of default values, see Service Assurance Adapter Platform and Notifications on page 55.)</p>
<pre>print <ClassName> <InstanceName> <EventName></pre>	<p>Prints all of the notification's attributes to stdout.</p>
<pre>release <ClassName> <InstanceName> <EventName></pre>	<p>Clears the notification's ownership. Attribute values modified using this command argument do not get propagated to Global Manager through the Adapter Platform (Open Integration).</p>
<pre>summarize [<Notification_List>]</pre>	<p>Prints a summary for a given notification list. If a notification list is not specified, the notification list named "Default" is used.</p>
<pre>unacknowledge <ClassName> <InstanceName> <EventName></pre>	<p>Changes the notification's attribute, Acknowledged to FALSE. Attribute values modified using this command argument do not get propagated to Global Manager through the Adapter Platform (Open Integration).</p>
<pre>update <ClassName> <InstanceName> <EventName> <attribute>=<value> ...</pre>	<p>Modifies one or more of a notification's attributes.</p>

Table 12: sm_ems Commands

An Example of the sm_ems Notify Command

The following sm_ems command creates a notification on the Adapter Platform (Open Integration) server ICOI. The notification is for a Router named RouterNY25 and it is Down, which is in this case a durable event. The notification expires in 3600 seconds. The Category attribute of the standard notification contains the name of the event as seen in the third-party application.

```
▼% sm_ems --server=ICOI notify Router RouterNY25 Down
3rdParty DURABLE 3600 Category=3rdParty-Down Severity=2 ▲
```

▼▲ Indicates the command must be typed as one line.

An Example of the sm_ems Create Element Option With Notify Command

The following sm_ems command uses the --create-element option along with the notify command to:

- Create a Host, WebHost (Host::WebHost) if it does not exist in the InCharge topology.
- Generate a notification, ApplicationService::WebServer Stopped. Specifies that the event was sent by a third party, is a momentary type of event, will be cleared by its source, and has a severity level of 2. Also, implies that the event OccurredOn Host::WebHost.

```
▼% sm_ems --server=ICOI --element-class=Host
--element-name=WebHost --create-element notify
ApplicationService WebServer Stopped 3rdParty momentary
source severity=2 ▲
```

An Example of the sm_ems Aggregate Option With Notify Command

The following sm_ems command uses aggregate options along with the notify command to generate two notifications:

- Generate an Aggregate notification, CPU::WebHost-CPU High-Utilization. Specifies that the event was sent by a third party, is a momentary type of event, and has a severity level of 1.
- Generate a component Notification, Host::WebHost Unresponsive, which is associated with the previously generated Aggregate Notification.

```
▼% sm_ems --server=ICOI --aggregate-element-class=Host
--aggregate-element-name=WebHost --aggregate-event-
```

```
name=unResponsive notify CPU WebHost-CPU High-Utilization  
3rdParty momentary source severity=1 ▲
```



Service Assurance Adapter Platform and Notifications

The purpose of the Service Assurance Adapter Platform is to normalize events and place them into a topological context so that they can be imported into a Global Manager. This means that the Adapter Platform server takes incoming events from a variety of other sources and translates the information into standard notifications. The standard notification is based on the ICIM_Notification class and consists of many attributes. When the Adapter Platform is used as a platform for ASM Adapters, the ASM Adapters set the properties of the notification attributes. Depending on the configuration of the adapter(s), the attributes created for each notification may or may not be populated with data.

Also, when the Global Manager collects the notification information from the Adapter Platform server, not all of the attributes of the notification get passed by default. Table 13 lists the attributes of the notifications for Service Assurance and also identifies those that get passed to the Global Manager.

NOTIFICATION ATTRIBUTE	TYPE	VALUE RANGE (DEFAULT VALUE)	PASSED BY DEFAULT	DESCRIPTION
Acknowledged	Boolean	TRUE or FALSE	No	TRUE if the notification has been acknowledged, FALSE if not.
Active	Boolean	TRUE or FALSE (Default: TRUE)	No	TRUE if the notification is active, FALSE if not.
AuditTrail	Special		No	Audit trail for the notification. AuditTrail is a table that includes: <ul style="list-style-type: none"> ActionType SerialNumber Text Timestamp User
Category	String		Yes	Type of notification. Possible values include: <ul style="list-style-type: none"> Availability Discovery Error Operational Performance PowerSupply Resource Temperature IMPACT <p>These categories correspond to the Exceptions of the same name described in the <i>InCharge Availability Manager User's Guide</i>, the <i>InCharge Performance Manager User's Guide</i>, the <i>InCharge Application Services Manager User's Guide</i>, and the <i>InCharge Service Assurance Manager User's Guide for Business Impact Manager</i>.</p>
Certainty	Float	0 to 100 (Default: 100)	Yes	Confidence level that this notification is the correct diagnosis.
ClassDisplayName	String		Yes	Class name of the managed element (where the event occurred) that is displayed to the user.
ClassName	String		Yes	Class name of the managed element where the event occurred. This attribute along with InstanceName and EventName uniquely identify this notification. ClassName may differ from the ElementClassName.

NOTIFICATION ATTRIBUTE	TYPE	VALUE RANGE (DEFAULT VALUE)	PASSED BY DEFAULT	DESCRIPTION
ClearOnAcknowledge	Boolean	TRUE or FALSE (Default: FALSE)	Yes	TRUE if the notification should be cleared when it is acknowledged. Use this in cases when notifications never expire or that have sources that will not generate a clear.
ElementClassName	String		Yes	Class Name of the object that the event occurred on.
ElementName	String		Yes	Element Name of the object that the event occurred on.
EventDisplayName	String		Yes	Name of the notification that is displayed to the user.
EventName	String		Yes	Name of the notification. This attribute along with ClassName and InstanceName uniquely identify this notification.
EventState	String		No	Indicates the state of the notification. Possible values include: <ul style="list-style-type: none"> • ACTIVE • SUSPENDED • WAS_ACTIVE • INACTIVE • UNINITIALIZED
EventText	String		Yes	A description of the notification.
EventType	Special	MOMENTARY or DURABLE (Default: DURABLE)	Yes	Indicates the nature of the event. A MOMENTARY event has no duration (such as an authentication failure). A DURABLE event has a period during which the event is active and after which the event is no longer active (such as a link failure).
FirstNotifiedAt	Integer	(Default: 0)	No	Time since the Epoch when the notification first became active.
Impact	Integer	(Default: 0)	No	A number that quantifies the impact of this notification on the infrastructure or business processes. Larger numeric values indicate a larger impact.
InMaintenance	Boolean	TRUE or FALSE (Default: FALSE)	No	TRUE indicates that the device associated with the notification is in maintenance mode, FALSE if not.
InstanceDisplayName	String		Yes	Name of the instance that is displayed to the user. Note that if tagging is implemented in the Global Manager, the InstanceDisplayName may include a tag suffix.

NOTIFICATION ATTRIBUTE	TYPE	VALUE RANGE (DEFAULT VALUE)	PASSED BY DEFAULT	DESCRIPTION
InstanceName	String		Yes	Name of the instance where the event occurred. This attribute along with ClassName and EventName uniquely identify this notification.
IsRoot	Boolean	TRUE or FALSE	No	TRUE if the notification is a root cause, FALSE if not.
LastChangedAt	Integer	(Default: 0)	No	Time since the Epoch when the status of the notification last changed. This is calculated by the Global Manager
LastClearedAt	Integer	(Default: 0)	No	Time since the Epoch when the notification was last cleared. This is calculated by the Global Manager
LastNotifiedAt	Integer	(Default: 0)	No	Time since the Epoch when the notification was last notified. This is calculated by the Global Manager.
Name	String		No	Internal identifier for the notification.
Occurrence Count	Integer		No	Number of times the notification has occurred.
Owner	String		No	Name of the user responsible for handling this notification.
Severity	Integer	1 to 5 (Default: 5)	Yes	An enumerated value that describes the severity of the notification from the notifier's point of view. Valid values are the integers 1 through 5. 1 - "Critical" Indicates action is needed NOW and the scope is broad, e.g. an outage to a critical resource. 2 - "Major" Indicates action is needed NOW. 3 - "Minor" Indicates action is needed, but the situation is not serious at this time. 4 - "Unknown" Indicates that the element is unreachable, disconnected or in an otherwise unknown state. 5 - "Normal" Used when an event is purely informational. Note that only the numbers, not the text descriptions, are passed by the global manager.
SourceDomainName	String		No	The name(s) of the domain(s) that have notified current occurrences of this event. If there is more than one domain, the attribute lists each separated by a comma.
TroubleTicketID	String		No	Trouble-ticket number associated with this notification.

NOTIFICATION ATTRIBUTE	TYPE	VALUE RANGE (DEFAULT VALUE)	PASSED BY DEFAULT	DESCRIPTION
UserDefined1	String		No	User defined field 1.
UserDefined2	String		No	User defined field 2.
UserDefined3	String		No	User defined field 3.
UserDefined4	String		No	User defined field 4.
UserDefined5	String		No	User defined field 5.
UserDefined6	String		No	User defined field 6.
UserDefined7	String		No	User defined field 7.
UserDefined8	String		No	User defined field 8.
UserDefined9	String		No	User defined field 9.
UserDefined10	String		No	User defined field 10.

Table 13: Notification Attributes

Note: For attributes that contain a time value, time is counted in seconds from Midnight, January 1st, 1970 GMT (the Epoch).

Index

A

- acknowledge, sm_ems command 51
- Acknowledged 56
- Active 56
- Adapter
 - SNMP Trap Receiver 15
 - Syslog 35
- Adapter Platform
 - sm_ems 12
- Adapter Platform server
 - Starting and stopping 13
- Adapter Platform, Service Assurance
 - Architecture 4
 - Defined 2
 - dxs-sysip.conf 5
 - Global Manager, connection to 8
 - icoi.conf 5
 - icoi-config.dtd 5
 - icoi-config-sample.xml 6
 - icoi-default.xml 5
 - nlconfig-sample.xml 6
 - profileconfig-sample.xml 6
 - Security 9, 12
 - sm_ems 9
 - trap_mgr.conf 18
 - userconfig-sample.xml 6
- add-audit-log, sm_ems command 51
- AGG_ELEMENTNAME
 - Syslog Adapter configuration parameter 41
- AGG_EVENTNAME
 - Syslog Adapter configuration parameter 41
- AGG_EVENTTEXT
 - Syslog Adapter configuration parameter 41
- Aggregate
 - SNMP Trap Receiver configuration parameter 21
- ASL
 - Command-line interface 49
 - Input/output variable 30
 - Input-only variable 30
 - SNMP Trap Receiver 30
 - SNMP Trap Receiver configuration parameter 21
- assign, sm_ems command 51
- Attributes

- Notification 55
- AuditTrail 56

B

- BASEDIR ix
- BATCH_NOTIFY_INTERVAL
 - SNMP Trap Receiver 33
 - Syslog Adapter 46
- brokerConnect.conf 9

C

- CATEGORY
 - Syslog Adapter configuration parameter 41
- Category
 - Notification attribute 56
 - SNMP Trap Receiver configuration parameter 22
- Certainty 56
- ClassDisplayName 56
- CLASSNAME
 - Syslog Adapter configuration parameter 41
- ClassName
 - Notification attribute 56
 - SNMP Trap Receiver configuration parameter 22
- CLEAR notification example 25
- clear, sm_ems command 51
- CLEAR_SYSLOG
 - Syslog Adapter configuration parameter 41
- CLEARONACKNOWLEDGE
 - Syslog Adapter configuration parameter 41
- ClearOnAcknowledge
 - Notification attribute 57
 - SNMP Trap Receiver configuration parameter 22
- clientConnect.conf 9, 10, 11, 12
- Commands
 - Command-line interface 49
 - sm_ems
 - acknowledge 51
 - add-audit-log 51
 - assign 51
 - clear 51
 - notify 52
 - print 52

- release 52
- summarize 52
- unacknowledge 52
- update 52
- sm_ems notify, example 53
- Configuration
 - Command-Line Interface 49
 - sm_edit utility 13
 - SNMP trap parameters 20
 - Syslog Adapter parameters 40
- CREATE notification example 25
- CUSTOM rule 39

D

- Defaults
 - SNMP trap receiver parameter 19
 - Standard notifications for Adapter Platform 56
- DISCARD
 - Syslog Adapter configuration parameter 41
- DISCARD_TRAP 30
- dxa-oi.conf 8
- dxa-sysip.conf 5

E

- ELEMENTCLASSNAME
 - Syslog Adapter configuration parameter 41
- ElementClassName
 - Notification 57
 - SNMP Trap Receiver configuration Parameter 22
- ELEMENTINSTANCENAME
 - Syslog Adapter configuration parameter 42
- ElementName
 - Notification 57
 - SNMP Trap Receiver configuration parameter 22
- Enterprise OID 16, 20, 26
- Epoch 59
- EventDisplayName 57
- EVENTNAME
 - Syslog Adapter configuration parameter 42
- EventName
 - Notification attribute 57
 - SNMP Trap Receiver configuration parameter 22
- EventState
 - Notification attribute 57
- EVENTTEXT
 - Syslog Adapter configuration parameter 42
- EventText
 - Notification attribute 57
 - SNMP Trap Receiver configuration parameter 22

- EVENTTYPE
 - Syslog Adapter configuration parameter 42
- EventType
 - Notification attribute 57
 - SNMP Trap Receiver configuration parameter 22
- EXPIRATION
 - Syslog Adapter configuration parameter 42
- Expiration
 - SNMP Trap Receiver configuration parameter 22

F

- FirstNotifiedAt 57

G

- Generic trap number 16, 20, 26

I

- icoi.conf 5, 6
 - SystemDefaultSection 7
 - AuditTrailSizeLimit 8
 - AutoAcknowledgementInterval 7
 - InactiveAutoArchiveInterval 8
- icoi-config.dtd 5
- icoi-config-sample.xml 6
- icoi-default.xml 5
- ic-syslog-adapter 36
- Impact notification attribute 57
- Importing
 - Event information 1
 - Topology elements 3
 - Topology information 3
- InCharge Information Model
 - ICIM_Notification class 55
- INMAINTENANCE
 - Syslog Adapter configuration parameter 42
- InMaintenance
 - Notification attribute 57
 - SNMP Trap Receiver configuration parameter 22
- InstanceDisplayName 57
- INSTANCENAME
 - Syslog Adapter configuration parameter 42
- InstanceName
 - Notification attribute 58
 - SNMP Trap Receiver configuration parameter 22
- IsRoot 58

L

- LastChangedAt 58

LastClearedAt 58
LastNotifiedAt 58
LOGFILE
 Syslog Adapter configuration parameter 42
LogFile
 SNMP Trap Receiver configuration parameter 22

M

Map parameter 26
 SNMP Trap Receiver 23
MODIFY_ATTRIBUTES rule 39
my_hook_syslog.asl 36, 37

N

Name 58
nlconfig-sample.xml 6
Notifications
 Attributes 55
 Example of SET and CLEAR 25
 SNMP Trap, translation of 17
 Standard, Adapter Platform 56
notify, sm_ems command 52

O

Occurrence Count 58
Owner 58

P

PARSE_EVENT_TEXT 32
PARSE_MESSAGE rule 39
print, sm_ems command 52
profileconfig-sample.xml 6

R

release, sm_ems command 52

S

Security
 Adapter Platform 9
 sm_ems command-line interface_ems
 Security 12
 SNMP Trap Receiver 10
 Syslog Adapter 11
serverConnect.conf 9, 10, 11, 12
Set and clear correlation, example 25
SEVERITY
 Syslog Adapter configuration parameter 42

Severity
 Notification attribute 58
 SNMP Trap Receiver configuration parameter 23
sm_edit utility 13
sm_ems command-line interface 49
 Basic format 49
 Commands
 acknowledge 51
 add-audit-log 51
 assign 51
 clear 51
 notify 52
 print 52
 release 52
 summarize 52
 unacknowledge 52
 update 52
 Example of create-element 53
 Example of notify 53
 Options
 aggregate-element-class 51
 aggregate-element-name 51
 aggregate-event 51
 audit 51
 broker 50
 create-element 50
 create-system 50
 element-class 50
 element-name 50
 server 50
 system 50
 traceServer 51
 Starting 49
SNMP Trap Integrator
 CLEAR notification 25
 Varbind processing 32
SNMP Trap Receiver 2
 Batch interval 33
 Default parameters 19
 Defined 15
 Definition parameters 20
 Integration using ASL 30
 Map parameter 26
 NOTIFY 25
 Starting and stopping 33
 Trap definition section 19
 Variables 24
SourceDomainName 58
Specific trap number 16, 20, 26
START rule 38

- Starting and stopping
 - Adapter Platform server 13
 - sm_ems 49
 - SNMP Trap Receiver 33
 - Syslog Adapter 47
 - STATE
 - Syslog Adapter configuration parameter 42
 - State
 - SNMP Trap Receiver configuration parameter 23
 - summarize, sm_ems command 52
 - Syslog (system log) file 35
 - Format of 36
 - Syslog Adapter 2
 - Batch interval 46
 - Configuring 39
 - CUSTOM rule 39
 - MODIFY_ATTRIBUTES rule 39
 - PARSE_Message rule 39
 - Processing 36
 - Security 11
 - START rule 38
 - Starting and stopping 47
 - Tailing 36
 - Template 36
 - syslog_mgr.asl 36
 - SYSNAMEORADDR
 - Syslog Adapter configuration parameter 42
 - SysNameOrAddr
 - SNMP Trap Receiver configuration parameter 23
 - System defaults
 - icoi.conf 7
- ## T
- Tailing, Syslog 36
 - Technical Support xiii
 - Topology Importer 3
 - trap_mgr.conf 18
 - TroubleTicketID 58
- ## U
- unacknowledge, sm_ems command 52
 - UNKNOWNAGENT
 - Syslog Adapter configuration parameter 42
 - UnknownAgent
 - SNMP Trap Receiver configuration parameter 23
 - update, sm_ems command 52
 - userconfig-sample.xml 6
 - USERDEFINED1-10
 - Syslog Adapter configuration parameter 43
 - UserDefined1-10
 - Notification attribute 59
 - SNMP Trap Receiver configuration parameter 23
- ## V
- Variables
 - SNMP Trap Receiver 24
 - Syslog Adapter 38
 - TEXT 32