InCharge™

IP Discovery Guide Supplement for Networking Protocols

Version 1.1

OL-7717-01

smarts®

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 PARITES 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 © 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

RSA Data Security, Inc.

AES

Contents

Preface **vii**

Intended Audience vii
Prerequisites vii
Document Organization viii
Documentation Conventions viii
InCharge Network Protocol Suite Installation Directory ix
InCharge Network Protocol Suite Products x
Additional Resources x
InCharge Commands x
Documentation x
Technical Support xi

1 Discovery Overview **1**

Network Protocol Manager Architecture 1
Routing Protocol and Device Support 2
Polling Support 3
Availability Manager Network Protocol Discovery 3
BGP Conditional Import From Availability Manager 4
OSPF Conditional Import From Availability Manager 5
Network Protocol Manager Discovery 5
Network Protocol Manager for BGP Discovery 6
Network Protocol Manager for OSPF Discovery 8

2 Configuring and Initiating Discovery **11**

Configuring Discovery 11
Initiating Discovery 12
Open the Domain Manager Administration Console 13
Add Availability Manager as a Source 15

Resolve Discovery Errors 16
Resolve Pending List Errors 16

3 Using Post-Processing Scripts 19
About the ASL Discovery Hook Scripts 19
Viewing the ASL Discovery Hook Scripts 20
Modifying an ASL Discovery Hook Script 23
Example BGP ASL Discovery Hook Scripts 23
Example OSPF ASL Discovery Hook Scripts 27

4 Scheduling and Manually Invoking Discovery 31
Scheduling Discovery 31
Pending Discovery Interval 32
Full Discovery Interval 32
Manually Invoking Discovery 33

Index 35

Preface

This document describes how to prepare InCharge Network Protocol Manager for discovery of logical routing topology in an IP environment, how to configure and initiate discovery, and how to customize the discovery process. This document is intended as a supplement to the *InCharge IP Discovery Guide*.

Intended Audience

This document is intended to be read by IT managers seeking to understand how the logical routing discovery process works, and by system administrators responsible for the administration, configuration, or use of InCharge applications that provide logical routing discovery over IP networks.

Prerequisites

It is assumed that readers of this document have read the *InCharge IP Discovery Guide* and are familiar with InCharge discovery concepts.

It is also assumed that InCharge Network Protocol Manager, InCharge IP Availability Manager, and InCharge Service Assurance Manager are installed. The Global Console is required to perform certain configuration tasks and to initiate logical routing discovery.

For information about installing these products, see the *InCharge Network Protocol Management Suite Installation Guide*, the *InCharge IP Management Suite Installation Guide*, and the *InCharge Service Assurance Management Suite Installation Guide*.

Document Organization

This document consists of the following chapters.

Table 1: Document Organization

1. DISCOVERY OVERVIEW	Describes the concepts of discovering logical routing network topology using InCharge Network Protocol Manager.
2. CONFIGURING AND INITIATING DISCOVERY	Explains how to configure and initiate logical routing discovery for a Network Protocol Manager deployment.
3. USING POST-PROCESSING SCRIPTS	Provides information and instructions for using post-processing scripts to customize the logical routing discovery process for Network Protocol Manager.
4. SCHEDULING AND MANUALLY INVOKING DISCOVERY	Provides information about how to schedule pending and full discovery and how to manually invoke discovery for Network Protocol Manager.

Documentation Conventions

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

Table 2: Documentation Conventions

CONVENTION	EXPLANATION
`sample code`	Indicates code fragments and examples in Courier font
keyword	Indicates commands, keywords, literals, and operators in bold
`%`	Indicates C shell prompt
#	Indicates C shell superuser prompt
<parameter>	Indicates a user-supplied value or a list of non-terminal items in angle brackets
[option]	Indicates optional terms in brackets

Table 2: Documentation Conventions (Continued)

CONVENTION	EXPLANATION
/InCharge	Indicates directory path names in italics
yourDomain	Indicates a user-specific or user-supplied value in bold, italics
File > Open	Indicates a menu path in italics
▼▲	Indicates a command is wrapped over one or more lines. The command must be typed as one line.

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 Network Protocol Suite Installation Directory

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

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

The *<n>* represents the InCharge software platform version number. The *<productsuite>* represents the InCharge product suite to which the product belongs. For example, on UNIX operating systems, InCharge Network Protocol Manager is, by default, installed to: */opt/InCharge6/NPM/smarts*. On Windows operating systems, this product is, by default, installed to: *C:\InCharge6\NPM\smarts*. This location is referred to as ***BASEDIR****/smarts*.

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

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

InCharge Network Protocol Suite Products

The InCharge Network Protocol Management Suite includes the following products:

- Network Protocol Manager for BGP
- Network Protocol Manager for OSPF

Additional Resources

In addition to this document, 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 document 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 System Administration Guide*
- *InCharge ICIM Reference*

- *InCharge ASL Reference Guide*
- *InCharge Perl Reference Guide*

InCharge Network Protocol Management Documentation

The following SMARTS documents are relevant to users of the InCharge Network Protocol Management product suite:

- *InCharge Network Protocol Management Suite Installation Guide*
- *InCharge Network Protocol Manager for BGP User's Guide*
- *InCharge Network Protocol Manager for OSPF User's Guide*
- *InCharge Network Protocol Manager Configuration Guide*
- *InCharge IP Discovery Guide Supplement for Networking Protocols*

Refer to the *InCharge Documentation Roadmap* for documentation resources provided with other SMARTS InCharge product suites.

Technical Support

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

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

U.S.A Technical Support

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

EMEA Technical Support

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

Asia-Pac Technical Support

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

You may also contact SMARTS at:

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

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

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

1

Discovery Overview

InCharge Network Protocol Manager manages routing-protocol availability in IP networks. It discovers the logical routing topology, monitors routing-protocol services and routing adjacencies, and explains their failures.

Network Protocol Manager Architecture

Network Protocol Manager can discover and monitor devices running either of the following network protocols:

- Border Gateway Protocol (BGP)
- Open Shortest Path First (OSPF)

A single instance of Network Protocol Manager is required for each protocol: one for BGP and another for OSPF. In addition, the architecture requires the following InCharge products:

- InCharge IP Availability Manager (Availability Manager)
- InCharge Service Assurance Manager (Service Assurance Manager)
- Global Console

Figure 1 illustrates the flow of information between Network Protocol Manager and these other products. In larger InCharge deployments, multiple Availability Managers can feed topology and events to the Network Protocol Manager.

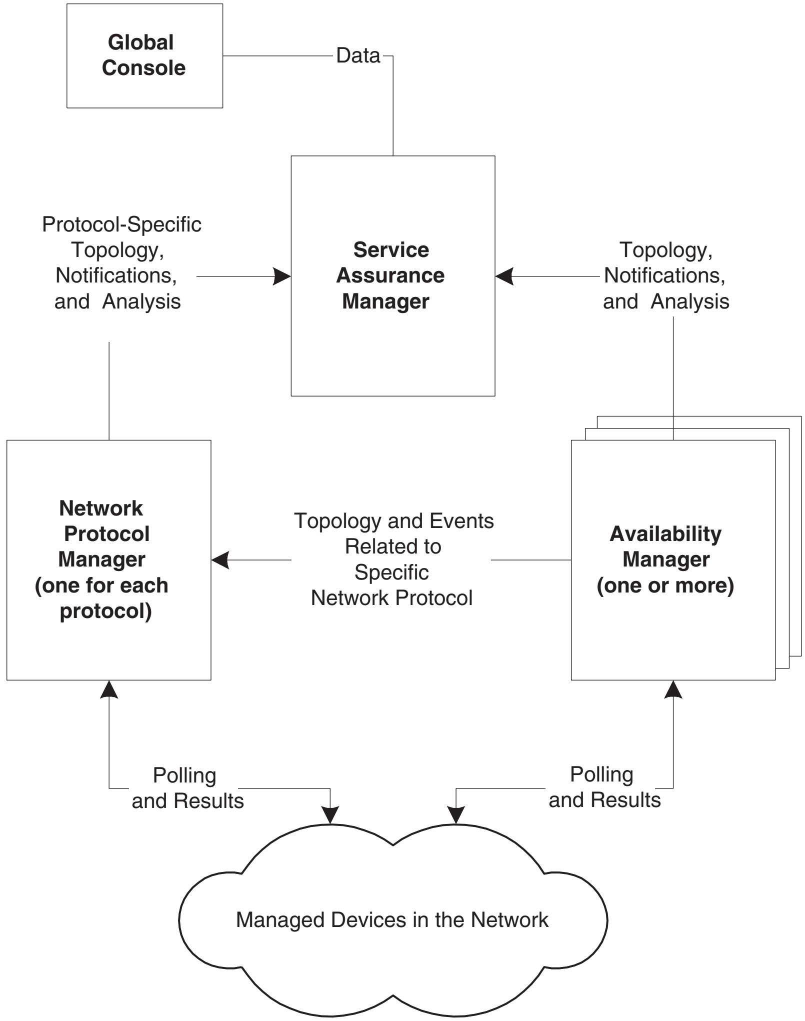

Figure 1: Discovery and the InCharge Network Protocol Manager Architecture

Routing Protocol and Device Support

Network Protocol Manager supports the BGP and OSPF routing protocols and any routing device—router, router switch module (RSM), router switch feature card (RSFC), or multilayer switch feature card (MSFC)—that supports the SNMP BGP-4 MIB defined in RFC 1657 or the SNMP OSPF MIB defined in RFC 1253. RSMs, RSFCs, and MSFCs are routing devices installed as cards in Layer 3 switches.

Note: Network Protocol Manager does not currently support virtual routers that may be implemented within a physical router or switch.

Network Protocol Manager for BGP discovers all BGP services running on a routing device and collectively represents the services as a single BGP service. Similarly, Network Protocol Manager for OSPF discovers all OSPF services running on a routing device and collectively represents the services as a single OSPF service.

Polling Support

Network Protocol Manager support Simple Network Management Protocol (SNMP) V1 and V2C polling but not SNMP V3 polling. In addition, Network Protocol Manager does not support Internet Control Message Protocol (ICMP) polling.

Availability Manager Network Protocol Discovery

Network Protocol Manager imports topology from one or more Availability Managers. To accomplish this end, Availability Manager not only performs its traditional discovery of network elements, but it also creates BGP and OSPF-related element collection sets from the discovered network elements. Network Protocol Manager imports the collection sets from Availability Manager. Figure 2 summarizes Availability Manager service discovery and Network Protocol Manager topology import.

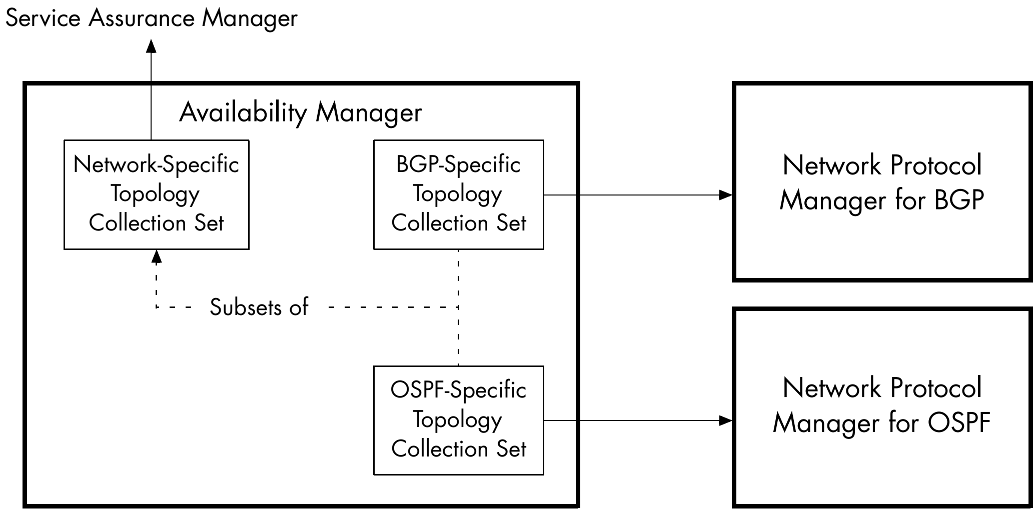

Figure 2: Network Protocol Manager Topology Import from Availability Manager

Availability Manager maintains BGP/OSPF routing device and service elements in the BGP/OSPF element collection set, along with the Interface, IPNetwork, Card, NetworkConnection, and IP elements associated with the BGP/OSPF routing device and service elements.

Each time Availability Manager saves its topology, Network Protocol Manager performs a topology synchronization to import the content of the BGP/OSPF element collection set. Availability Manager saves its topology when it completes a discovery or rediscovery cycle, or whenever a user manually saves the Availability Manager topology.

BGP Conditional Import From Availability Manager

From Availability Manager, Network Protocol Manager for BGP imports the initial BGP-related topology, receives topology updates, and receives Layer 1 and 2 events relevant to the imported BGP-related topology.

During discovery, Availability Manager queries a device's SNMP agent for an additional MIB object named bgpPeerLocalAddr to determine whether the device supports the BGP routing protocol. If bgpPeerLocalAddr exists for the device, Availability Manager creates a ProtocolService::BGPService-*<system_device_name>* element and a HostedBy/HostsServices relationship between the device and the BGP service element; *<system_device_name>* represents the name of the device—router or switch—on which bgpPeerLocalAddr is found. Availability Manager queries every router or switch in the managed environment for BGP support.

OSPF Conditional Import From Availability Manager

From Availability Manager, Network Protocol Manager for OSPF imports the initial OSPF-related topology, receives topology updates, and receives Layer 1 and 2 events relevant to the imported OSPF-related topology.

During discovery, Availability Manager queries a device's SNMP agent for an additional MIB object named ospfIfIPAddress to determine whether the device supports the OSPF routing protocol. If ospfIfIPAddress exists for the device, Availability Manager creates a ProtocolService::OSPFService-*<system_device_name>* element and a HostedBy/HostsServices relationship between the device and the OSPF service element; *<system_device_name>* represents the name of the device—router or switch—on which ospfIfIPAddress is found. Availability Manager queries every router or switch in the managed environment for OSPF support.

Network Protocol Manager Discovery

Network Protocol Manager, in conjunction with Availability Manager, discovers the logical routing topology and uses the InCharge Common Information Model (ICIM) to represent the routing topology in its repository. Network Protocol Manager links the routing topology elements that it discovers with the network elements discovered by Availability Manager to build a complete model of the managed routing network.

Network Protocol Manager for BGP Discovery

After importing the initial BGP-related topology from Availability Manager, Network Protocol Manager for BGP polls certain BGP-4 MIB objects to discover the BGP elements. For a description of the BGP discovery MIB objects, see the *InCharge Network Protocol Manager for BGP User's Guide*.

BGP Data Model

Network Protocol Manager for BGP uses instances of the following ICIM classes to build a data model of the discovered BGP elements in its domain:

- AutonomousSystem
- BGPService
- BGPProtocolEndpoint
- BGPSession

The model, shown in Figure 3, represents the BGP elements, their relationships, and their connections.

During the discovery post-processing phase, Network Protocol Manager for BGP creates the relationships and connections between the BGP elements. For every relationship or connection, there is an inverse relationship or connection. For example, the relationship PartOf is the inverse relationship of ComposedOf.

For descriptions of ICIM classes, relationships, and connections, see the *InCharge ICIM Reference*.

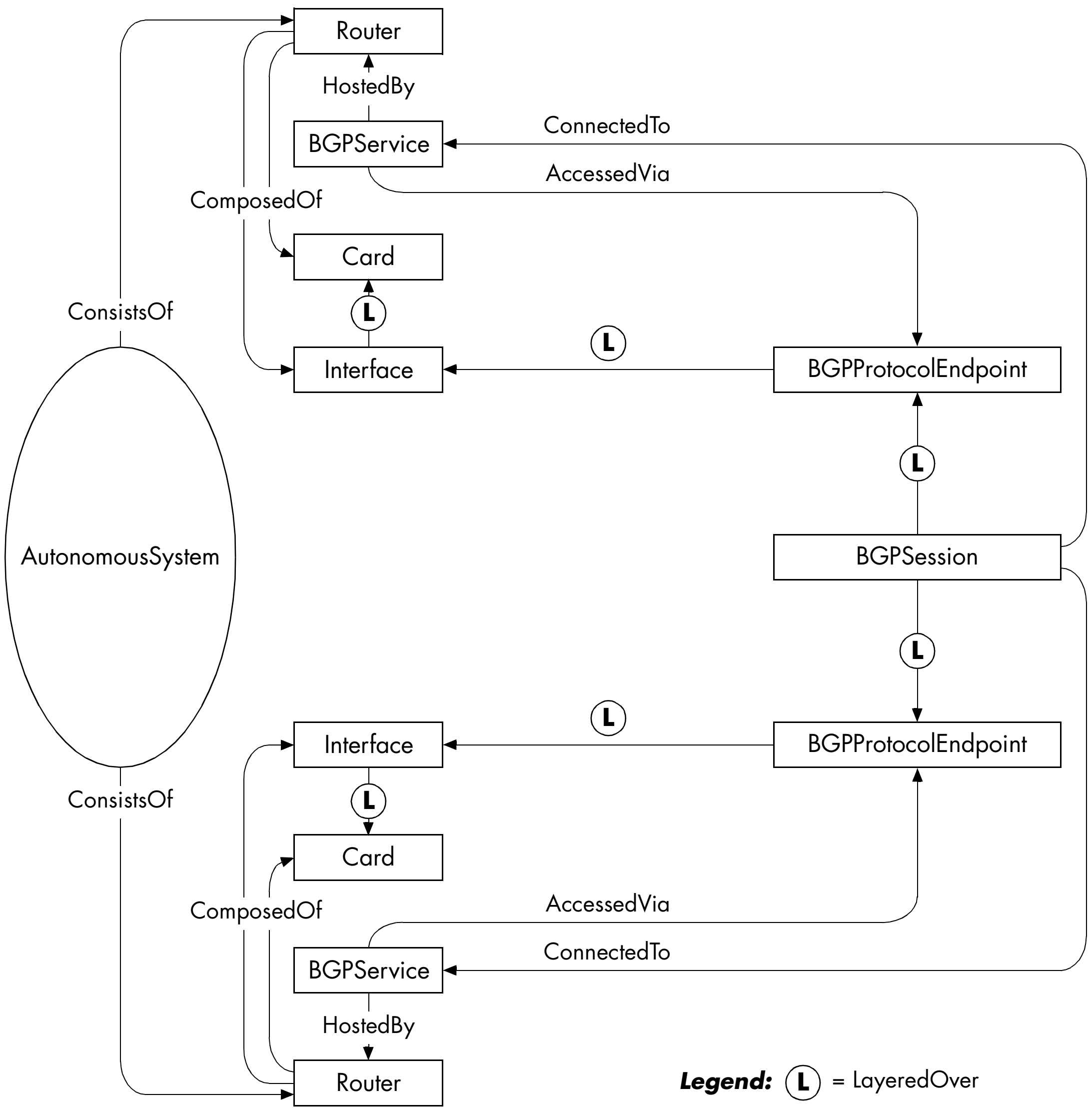

Figure 3: ICIM BGP Reference Model

BGP Discovery Log Messages

Detailed reports of topology synchronizations between Network Protocol Manager for BGP and Availability Manager are written to the log file named *BASEDIR/smarts/local/logs/<NPM_domain_manager_name>.log*; for example, *BASEDIR/smarts/local/logs/INCHARGE-BGP.log*. The log messages include the number of elements and relationships that were created during the topology synchronization.

At the end of the discovery post-processing phase, Network Protocol Manager for BGP prints a report to the log file identifying the number of autonomous systems, BGP services, BGP protocol endpoints, and BGP sessions present in Network Protocol Manager for BGP.

Network Protocol Manager for OSPF Discovery

After importing the initial OSPF-related topology from Availability Manager, Network Protocol Manager for OSPF polls certain OSPF MIB objects to discover the OSPF elements. For a description of the OSPF discovery MIB objects, see the *InCharge Network Protocol Manager for OSPF User's Guide*.

OSPF Data Model

Network Protocol Manager for OSPF uses instances of the following ICIM classes to build a data model of the discovered OSPF elements in its domain:

- OSPFArea
- OSPFAreaConfiguration
- OSPFService
- OSPFNetwork
- OSPFInterface
- OSPFVirtualInterface
- OSPFNeighborEndpoint
- OSPFVirtualNeighborEndpoint
- OSPFNeighborRelationship
- OSPFVirtualLink

The model, shown in Figure 4, represents the OSPF elements, their relationships, and their connections.

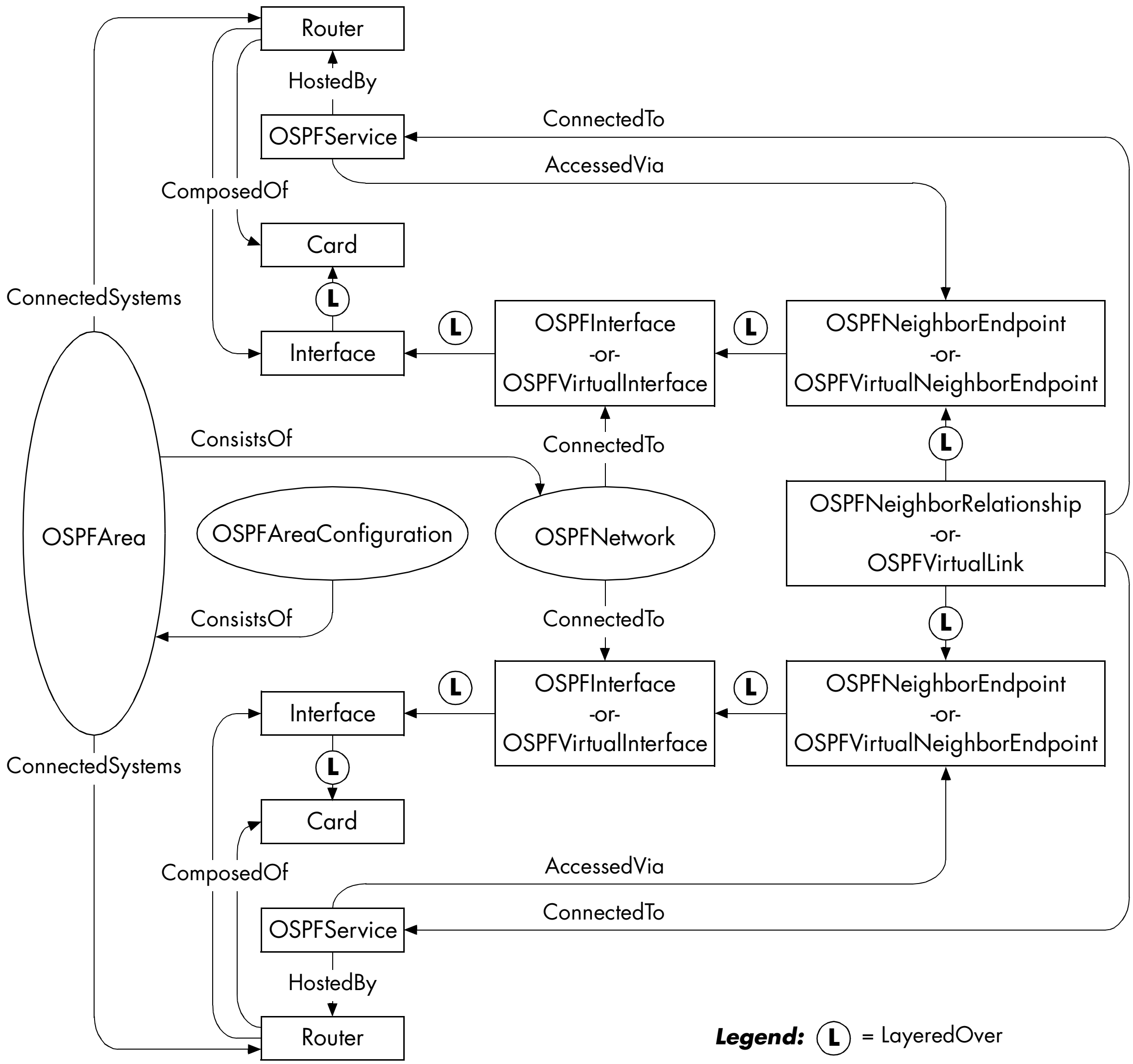

Figure 4: ICIM OSPF Reference Model

During the discovery post-processing phase, Network Protocol Manager for OSPF creates the relationships and connections between the OSPF elements. For every relationship or connection, there is an inverse relationship or connection. For example, the relationship PartOf is the inverse relationship of ComposedOf.

Note: ConnectedSystems has no inverse relationship.

For descriptions of ICIM classes, relationships, and connections, see the *InCharge ICIM Reference*.

OSPF Discovery Log Messages

Detailed reports of topology synchronizations between Network Protocol Manager for OSPF and Availability Manager are written to the log file named *BASEDIR/smarts/local/logs/<NPM_domain_manager_name>.log*; for example, *BASEDIR/smarts/local/logs/INCHARGE-OSPF.log*. The log messages include the number of elements and relationships that were created during the topology synchronization.

At the end of the discovery post-processing phase, Network Protocol Manager for OSPF prints a report to the log file identifying the number of OSPF areas, OSPF area configurations, OSPF services, OSPF networks, OSPF interfaces, OSPF virtual interfaces, OSPF neighbor endpoints, OSPF virtual neighbor endpoints, OSPF neighbor relationships, and OSPF virtual links present in Network Protocol Manager for OSPF.

2

Configuring and Initiating Discovery

Configuring a Network Protocol Manager deployment for discovery requires configuring the Network Protocol Manager, Availability Manager, and Service Assurance Manager applications in the deployment. Initiating discovery for the deployment requires initiating discovery for each Availability Manager application and adding each Availability Manager application as a source to Network Protocol Manager.

Configuring Discovery

To configure your Network Protocol Manager deployment for discovery, follow these steps:

1. Configure each Availability Manager for discovery as instructed in the *InCharge IP Discovery Guide*.
2. Ensure that each Availability Manager is ready for Network Protocol Manager related discovery by checking the following two parameters in the ***BASEDIR***/*smarts/conf/discovery/tpmgr-param.conf* file in the Availability Manager install area:
   - DisableProtocolDiscovery: When set to FALSE, the default, Availability Manager creates BGP and OSPF element collection sets for import by Network Protocol Manager. Do not change this setting.

- ProtocolInterfaceDisabledAlarm: When set to FALSE, the default, Availability Manager makes available Interface Disabled problems to Network Protocol Manager, to be used by Network Protocol Manager in its cross-domain analysis.

  Setting this parameter to TRUE hides Interface Disabled problems from Network Protocol Manager. Doing so inhibits Network Protocol Manager from sending Disabled problem notifications to Service Assurance Manager.

3. Configure each Network Protocol Manager for discovery as instructed in the *InCharge Network Protocol Manager Configuration Guide*.
4. Configure Service Assurance Manager for discovery as instructed in the *InCharge Network Protocol Manager Configuration Guide*. For general instructions on configuring Service Assurance Manager, see the *InCharge Service Assurance Manager Configuration Guide*.

Initiating Discovery

After configuring your Network Protocol Manager deployment for discovery, initiate discovery for each Availability Manager as instructed in the *InCharge IP Discovery Guide*.

Then, initiate discovery for each Network Protocol Manager in the deployment by adding each Availability Manager in the deployment as a source to Network Protocol Manager. When you add Availability Manager as a source, Network Protocol Manager imports the BGP- or OSPF-specific topology from Availability Manager and then sends its own SNMP polls to the routing devices to discover the protocol information that it needs to build the BGP/OSPF network topology.

You use the Domain Manager Administration Console attached to Network Protocol Manager to add one or more Availability Managers as sources to Network Protocol Manager.

Open the Domain Manager Administration Console

Attaching the Global Console to a Domain Manager such as Network Protocol Manager requires an InCharge user account with the following privileges and permissions:

- *All* privileges, specified in the *serverConnect.conf* file (or its equivalent) read by the Domain Manager.
- Permission to use the console operation *Configure Domain Manager Admin Console*. Through the Global Manager Administration Console, this permission is specified in the Console Operations section of the user profile.

For information about configuring access privileges, see the *InCharge System Administration Guide*. For information about configuring permissions to perform specific console operations, see the *InCharge Service Assurance Manager Configuration Guide*.

To open the Domain Manager Administration Console, follow these steps:

1. Start the Global Console.
   - On a Windows system, choose *Start > Programs > InCharge 6 > InCharge Global Console*.
   - On a UNIX system, go to the ***BASEDIR****/smarts/bin* directory in the Service Assurance Manager install area and invoke `./sm_gui`.

   The InCharge Manager attach dialog box opens.
2. In the dialog box,
   - Ensure that the InCharge Broker for your deployment appears in the Broker text box.
   - Select the Network Protocol Manager application in your deployment as the InCharge Manager to which you want to connect.
   - Enter your login user name and password.
   - Click **OK**.

   The Topology Browser Console opens.
3. In the Console, choose *Configure > Domain Manager Administration Console*. The Domain Manager Administration Console opens, an example of which is shown in Figure 5.

In the example display, the Domain Manager Administration Console is attached to a Network Protocol Manager for BGP application named INCHARGE-BGP.

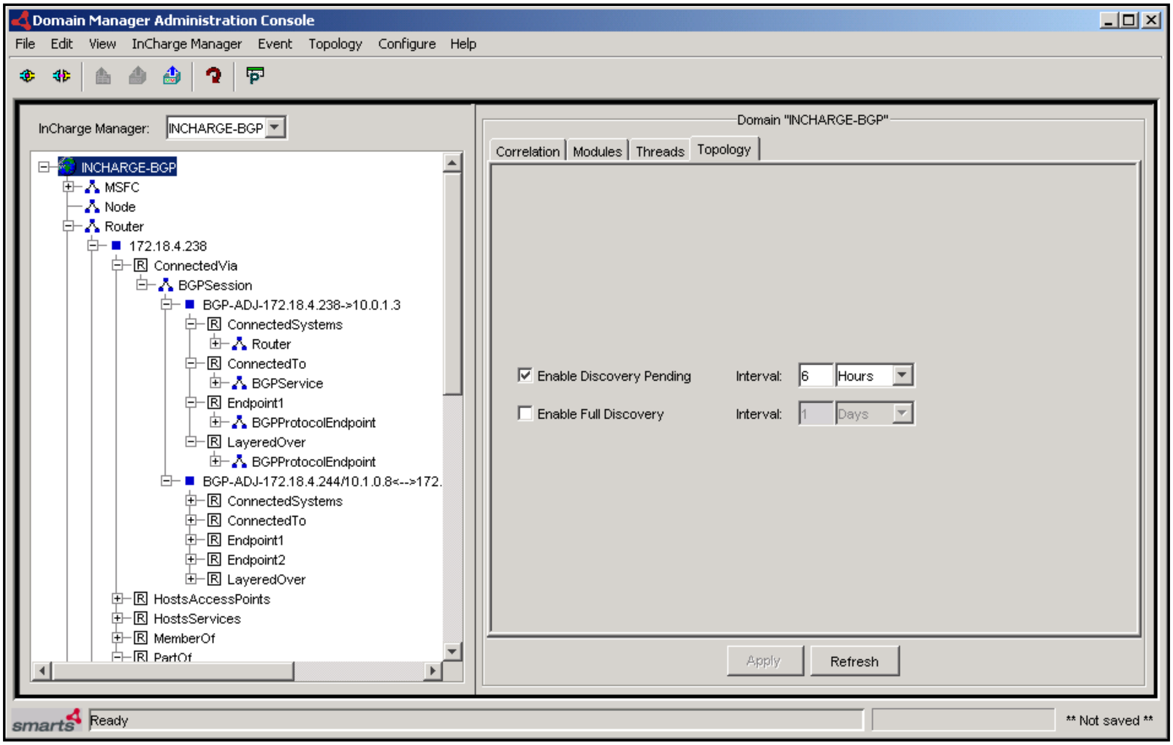

Figure 5: Domain Manager Administration Console—Example

Because Network Protocol Manager imports its topology and topology updates from Availability Manager, the manual discovery and autodiscovery features are not available to Network Protocol Manager. All other administration capabilities normally available through the Domain Manager Administration Console are available to administrators and operators of Network Protocol Manager.

Add Availability Manager as a Source

To add Availability Manager as a source to Network Protocol Manager, follow these steps:

1. In the Domain Manager Administration Console attached to Network Protocol Manager, choose *Topology > Add Source* to launch the Add Source dialog box.

2. In the dialog box, select the AM element type, enter the name of the Availability Manager application, and click **OK**.

   Network Protocol Manager probes the selected Availability Manager application for BGP- or OSPF-related topology information. A Discovery Progress window opens, displays progress messages, and may display a list of elements (pending elements) that could not be discovered. When you see *Last discovery completed* near the end of the progress report in the Discovery Status section, the probing of topology information is complete.

3. If any errors occur during the discovery process, see *Resolve Discovery Errors* on page 16. If any pending elements appear in the Discovery Progress window, see *Resolve Pending List Errors* on page 16. Click **Close** to close the Discovery Progress window.

4. Repeat Steps 1 through 3 until you have added each Availability Manager application in your deployment as a source to Network Protocol Manager.

When Availability Manager is added as a source, Network Protocol Manager immediately imports the BGP or OSPF object collection set from Availability Manager. Thereafter, Network Protocol Manager imports the BGP or OSPF element collection set from Availability Manager whenever Availability Manager saves its topology.

Each time that Network Protocol Manager imports the element collection set, it compares the collection set with the previously imported collection set. Whenever Network Protocol Manager finds a new routing device in the collection set, it sends SNMP polls to the routing device to discover its routing protocol elements and adds those elements to the logical routing topology.

Resolve Discovery Errors

When Network Protocol Manager attempts to discover a routing device and its routing protocol elements, one of the following results occurs:

- Discovery successfully completes, and the routing protocol elements associated with the routing device are added to the managed topology.
- Discovery starts but communication is lost between Network Protocol Manager and the routing device.
- Discovery starts but the routing device does not respond to SNMP polls.
- Discovery starts but one or more of the Network Protocol Manager probes encounters an error during the discovery process.
- Discovery is unable to start or successfully complete.

When a discovery error occurs, follow these basic steps:

1. Check the messages in the Discovery Status section of the Discovery Progress window.
2. Check the log file named *<NPM_domain_manager_name>.log* in the ***BASEDIR****/smarts/local/logs* directory. The discovery process writes an error to this log each time that it encounters an error and discovery is not successful.

Note: Network Protocol Manager does not notify discovery errors.

If discovery for a particular routing device or routing protocol element did not complete successfully, an entry for that element appears in the Pending Elements list.

For descriptions and resolutions of common discovery errors, see the *InCharge IP Discovery Guide*.

Resolve Pending List Errors

The Pending Elements list, which is displayed in the Discovery Progress window, lists elements that could not be discovered by Network Protocol Manager. Network Protocol Manager attempts to discover the elements on the Pending Elements list at the next scheduled pending list discovery. For details about scheduled pending list discovery, see *Scheduling Discovery* on page 31.

The Pending Elements list contains three columns that provide the following information for a pending element:

- Element Name: Name of the routing device or routing protocol element that could not be discovered
- Element Type: Name of the Network Protocol Manager probe that probes the routing device or routing protocol element identified in the Element Name field
- Discovery Error: Comment describing why the element identified in the Element Name field is on the Pending Elements list

Note: To see all the text associated with a particular element name, element type, or discovery error, position the mouse pointer over the element name, element type, or discovery error.

If you right-click an element in the Pending Elements list, a pop-up menu appears listing the following three selections: Remove, Rediscovery, and Show Discovery Time.

- Select *Remove* to remove the selected element from the Pending Elements list.
- Select *Rediscovery* to initiate an immediate rediscovery of the selected element.
- Select *Show Discovery Time* to view the last time that the selected element was discovered. If Show Discovery Time is unavailable (grayed-out), the element was never discovered.

3

Using Post-Processing Scripts

In addition to the post-processing steps performed automatically by Network Protocol Manager during the last phase of the discovery process, you can specify additional post-processing steps to occur at various stages of post-processing. For example, you could use a post-processing script to create system redundancy groups during discovery.

Network Protocol Manager provides two such post-processing script files written in Adapter Scripting Language (ASL). The files are known as ASL discovery hook scripts.

About the ASL Discovery Hook Scripts

The ASL discovery hook scripts, named *custom-end-system.asl* and *custom-end-post.asl*, are located in the ***BASEDIR****/smarts/rules/bgp/custom* directory and the ***BASEDIR****/smarts/rules/ospf/custom* directory of a Network Protocol Management Suite installation. The pair of ASL scripts in the *.../bgp/custom* directory is intended for Network Protocol Manager for BGP, and the pair of ASL scripts in the *.../ospf/custom* directory is intended for Network Protocol Manager for OSPF.

The ASL scripts are invoked at the following points during Network Protocol Manager discovery:

- After a routing device is discovered (*custom-end-system.asl*)
- After discovery post-processing (*custom-end-post.asl*)

Viewing the ASL Discovery Hook Scripts

By default, the ASL discovery hook scripts, shown in Figure 6 and Figure 7, do not perform any discovery processing, meaning that you must provide the ASL code that performs the actual processing. You use the *sm_edit* utility to edit these scripts and add additional initialization code, convenience patterns, parsing rules, processing rules, and additional EOF or DEFAULT code (if necessary).

```
/*
 * Copyright (C) 2004, System Management ARTS (SMARTS)
 * All Rights Reserved
 *
 * RCS $Id: custom-end-system.asl,v 1.1.2.2 2004/08/06
       01:27:13 sv1 Exp $
 *
 * This asl file is used at the end of probing of each
       device. */

default AgentName  = "";
default debug = FALSE;
default detailDebug = FALSE;
default objectName = "";
default elementName = "";
default elementType = "";
default changed = "TRUE";
default result = "SUCCESS";
default discoveryError = "";

agentObj = object(elementName);

if (agentObj->isNull()) {
    stop();
}

nodeObj = agentObj->getSystem();
if (nodeObj->isNull()) {
    stop();
}

factory = object(getInstances("ICIM_ObjectFactory")[0]);
topo_manager =
       object(getInstances("ICF_TopologyManager")[0]);
default DEBUG = topo_manager->DebugEnabled;

me = this->ReadsRulesFrom->fileName.": ";
blanks = "    ";
```

```
/*
 * Add additional initialization here.
 */


/*
 * Add convenience patterns here.
 */

/***********************************************************/

START {
    /*
     * Add Parsing rules here.
     */
    .. eol
} do {
    /*
     * Add processing rules here.
     */

    changed = "TRUE";
    result = "SUCCESS";
    objectName = agentObj->Name;

    if (DEBUG) {
        print("Executing rule set ".me);
    }

}

/*
 * Add additional EOF or DEFAULT here, if necessary.
 */

/***********************************************************/

/*
 * Local Variables:
 * mode: C++
 * End:
 */
```

Figure 6: ASL Discovery Hook Script custom-end-system.asl

```
/*
 * Copyright (C) 2004, System Management ARTS (SMARTS)
 * All Rights Reserved
 *
 * RCS $Id: custom-end-post.asl,v 1.1.2.2 2004/08/06 01:27:13
       sv1 Exp $
 *
 * This asl file is used at the end of discovery post
       processing.
 */

factory = object(getInstances("ICIM_ObjectFactory")[0]);
topo_manager =
       object(getInstances("ICF_TopologyManager")[0]);
default DEBUG = topo_manager->DebugEnabled;

me = this->ReadsRulesFrom->fileName.": ";
blanks = "    ";

/*
 * Add additional initialization here.
 */

/*
 * Add convenience patterns here.
 */

/************************************************************/

START {
    /*
     * Add Parsing rules here.
     */
    .. eol
} do {
    /*
     * Add processing rules here.
     */

    if (DEBUG) {
        print("Executing rule set ".me);
    }
}

/*
 * Add additional EOF or DEFAULT here, if necessary.
 */
```

```
/*********************************************************I**/

/*
 * Local Variables:
 * mode: C++
 * End:
 */
```

Figure 7: ASL Discovery Hook Script custom-end-post.asl

Modifying an ASL Discovery Hook Script

To modify an ASL discovery hook script, follow these steps:

1. Open the ASL script using the *sm_edit* utility in the ***BASEDIR****/smarts/bin* directory. For example:

   ```
   % ./sm_edit rules/bgp/custom/custom-post-system.asl
   ```

2. Add your ASL code and save the file. The modified ASL script is saved to the ***BASEDIR****/smarts/local/rules/bgp/custom* directory.

Example BGP ASL Discovery Hook Scripts

As an example of using an ASL discovery hook script for Network Protocol Manager for BGP, you could use the *custom-end-post.asl* script to create BGP Route Reflector (RR) system redundancy groups (SystemRedundancyGroup elements) after the initial discovery completes. Currently, SystemRedundancyGroup elements cannot automatically be discovered and instantiated by Network Protocol Manager.

Examples of adding code to the *custom-end-system.asl* and *custom-end-post.asl* scripts to create an RR SystemRedundancyGroup element are given in Figure 8 and Figure 9. The ASL code in Figure 8 directs Network Protocol Manager to assign an RR role to each discovered Juniper routing device running a BGP service. The ASL code in Figure 9 directs Network Protocol Manager to create an instance of the SystemRedundancyGroup class and then insert the RR routing devices participating in the redundancy group into the ComposedOf relationship of the redundancy group.

```
/*
 * Copyright (C) 2004, System Management ARTS (SMARTS)
 * All Rights Reserved
 *
 * RCS $Id: custom-end-system.asl,v 1.1.2.2 2004/08/06
       01:27:13 sv1 Exp $
 *
 * This asl file is used at the end of probing of each
       device. */

default AgentName  = "";
default debug = FALSE;
default detailDebug = FALSE;
default objectName = "";
default elementName = "";
default elementType = "";
default changed = "TRUE";
default result = "SUCCESS";
default discoveryError = "";


agentObj = object(elementName);

if (agentObj->isNull()) {
    stop();
}

nodeObj = agentObj->getSystem();
if (nodeObj->isNull()) {
    stop();
}

factory = object(getInstances("ICIM_ObjectFactory")[0]);
topo_manager =
       object(getInstances("ICF_TopologyManager")[0]);
default DEBUG = FALSE;

me = this->ReadsRulesFrom->fileName.": ";
blanks = "    ";

/*
 * Add additional initialization here.
 */


/*
 * Add convenience patterns here.
 */
```

```
/**********************************************************/

START {
    /*
     * Add Parsing rules here.
     */
    .. eol
} do {
    /*
     * Add processing rules here.
     */

    changed = "TRUE";
    result = "SUCCESS";
    objectName = agentObj->Name;

    if (DEBUG) {
        print("Executing rule set ".me);
    }
    if (nodeObj->Vendor == "JUNIPER") {
        Srvc = nodeObj->HostsServices;
        foreach svc (Srvc) {
            if (svc->isInstanceOf("BGPService")) {
            bgpeps = svc->AccessedVia;
       bgpep = bgpeps[0];
       if (sizeof(bgpep) == 0) {
           return;
       }
                 bgpep->IsRouteReflector = TRUE ;
            }
        }
   }
}

/*
 * Add additional EOF or DEFAULT here, if necessary.
 */


/**********************************************************/

/*
 * Local Variables:
 * mode: C++
 * End:
 */
```

Figure 8: Example BGP Discovery Hook Script custom-end-system.asl

```
/*
 * Copyright (C) 2004, System Management ARTS (SMARTS)
 * All Rights Reserved
 *
 * RCS $Id: custom-end-post.asl,v 1.1.2.2 2004/08/06 01:27:13
       sv1 Exp $
 *
 * This asl file is used at the end of discovery post
       processing.
 */

factory = object(getInstances("ICIM_ObjectFactory")[0]);
topo_manager =
       object(getInstances("ICF_TopologyManager")[0]);
default DEBUG = FALSE;

me = this->ReadsRulesFrom->fileName.": ";
blanks = "    ";

/*
 * Add additional initialization here.
 */


/*
 * Add convenience patterns here.
 */

/*************************************************************/

START {
    .. eol
} do {
    /*
     * Add processing rules here.
     */

    if (DEBUG) {
    print("Executing rule set ".me);
    }

    SRG = create("SystemRedundancyGroup","BGP-RR-Group");
    foreach bgpSys (getInstances("BGPService")) {
    bgpSys = object(bgpSys);
    if (bgpSys->IsRouteReflector == TRUE) {
        SRG->ComposedOf += bgpSys->HostedBy;
    }
    }
```

```
}

/*
 * Add additional EOF or DEFAULT here, if necessary.
 */


/*************************************************************/

/*
 * Local Variables:
 * mode: C++
 * End:
 */
```

Figure 9: Example BGP Discovery Hook Script custom-end-post.asl

Analysis and monitoring of the RR system redundancy group occur automatically when the redundancy group is created and its RR members are inserted into the group. The state of the redundancy group is based on the status of its RR members.

Example OSPF ASL Discovery Hook Scripts

As an example of using an ASL discovery hook script for Network Protocol Manager for OSPF, you could use the *custom-end-post.asl* script to create OSPF Area Border Router (ABR) system redundancy groups (SystemRedundancyGroup elements) after the initial discovery completes. Currently, SystemRedundancyGroup elements cannot automatically be discovered and instantiated by Network Protocol Manager.

An example of adding code to the *custom-end-post.asl* script to create an ABR SystemRedundancyGroup element is given in Figure 10. The ASL code directs Network Protocol Manager to create an instance of the SystemRedundancyGroup class and then insert the ABR routing devices participating in the redundancy group into the ComposedOf relationship of the redundancy group.

```
/*
 * Copyright (C) 2004, System Management ARTS (SMARTS)
 * All Rights Reserved
 *
 * RCS $Id: custom-end-post.asl,v 1.1.2.2 2004/08/06 01:27:13
       sv1 Exp $
 *
 * This asl file is used at the end of discovery post
       processing.
 */

factory = object(getInstances("ICIM_ObjectFactory")[0]);
topo_manager =
       object(getInstances("ICF_TopologyManager")[0]);
default DEBUG = FALSE;

me = this->ReadsRulesFrom->fileName.": ";
blanks = "    ";

/*
 * Add additional initialization here.
 */


/*
 * Add convenience patterns here.
 */

/*************************************************************/

START {
    .. eol
} do {
    /*
     * Add processing rules here.
     */

    if (DEBUG) {
    print("Executing rule set ".me);
    }

    SRG = create("SystemRedundancyGroup","OSPF-ABR-Group");
    foreach ospfSys (getInstances("OSPFService")) {
    ospfSys = object(ospfSys);
    if (ospfSys->IsABR == TRUE) {
        sysObj = ospfSys->HostedBy;
        if (sysObj->isInstanceOf("Router")) {
            SRG->ComposedOf += ospfSys->HostedBy;
```

```
        } else {
        SRG->ComposedOf -= ospfSys->HostedBy ? IGNORE;
        }
    }
    }
}

/*
 * Add additional EOF or DEFAULT here, if necessary.
 */


/************************************************************/

/*
 * Local Variables:
 * mode: C++
 * End:
 */
/*
```

Figure 10: Example OSPF Discovery Hook Script custom-end-post.asl

Analysis and monitoring of the ABR system redundancy group occur automatically when the redundancy group is created and its ABR members are inserted into the group. The state of the redundancy group is based on the status of its ABR members.

4

Scheduling and Manually Invoking Discovery

After initial Network Protocol Manager discovery, you can schedule automatic discovery at a regular interval and manually invoke discovery at any time. These functions are available through a Domain Manager Administration Console attached to Network Protocol Manager.

Scheduling Discovery

You can schedule *pending discovery* and *full discovery* intervals:

- Scheduling a pending discovery interval determines how often Network Protocol Manager attempts to discover the elements on its Pending Elements list.
- Scheduling a full discovery interval determines how often Network Protocol Manager attempts to rediscover the existing elements in its repository.

The parameters that control scheduled discovery are located on the Topology tab page of the Domain Manager Administration Console.

Pending Discovery Interval

The Pending Discovery Interval is enabled and set to six hours by default.

To change the Pending Discovery Interval, follow these steps:

1. Attach the Domain Manager Administration Console to the target Network Protocol Manager.
2. In the left panel of the Domain Manager Administration Console, click the Network Protocol Manager name to display a multiple-tab window in the right panel of the console.
3. Select the Topology tab.
4. Type a value in the Discovery Pending Interval field and select a time interval (Seconds, Minutes, Hours, or Days) from the drop-down menu.
5. Click **Apply**.

Before you change the pending discovery interval, you should be aware that every 10 minutes Network Protocol Manager checks whether the previous discovery started more than 30 minutes earlier and whether any elements are currently on the Pending Elements list. If both conditions are true, Network Protocol Manager performs a pending list discovery. You cannot disable or modify this behavior.

Full Discovery Interval

The Full Discovery Interval is disabled and set to one day by default.

To enable Full Discovery and specify a Full Discovery Interval, follow these steps:

1. Attach the Domain Manager Administration Console to the target Network Protocol Manager.
2. In the left panel of the Domain Manager Administration Console, click the Network Protocol Manager name to display a multiple-tab window in the right panel of the console.
3. Select the Topology tab.
4. Click the *Enable Full Discovery* checkbox.
5. Type a value in the Interval field for Full Discovery and select a time interval (Seconds, Minutes, Hours, or Days) from the drop-down list.
6. Click **Apply**.

Manually Invoking Discovery

There are several methods for manually invoking a discovery from the Domain Manager Administration Console, including:

- Choose an element and then select *Topology > Rediscover* to initiate a discovery of the selected element. You can also rediscover an element by right-clicking it and selecting **Rediscover** from the drop-down menu.
- Select *Topology > Discover Pending* to initiate a discovery of the elements on the Pending Elements list.
- Select *Topology > Discover All* to initiate a discovery of all managed elements, including those on the Pending Elements list.
- A single element on the Pending Elements list can be rediscovered by right-clicking the element and selecting **Rediscover** from the drop-down menu.

You can also invoke a pending discovery and a full discovery using the *dmctl* utility. (Doing so is useful if you want to schedule a pending discovery or a rediscovery at a specific time using the *cron* or *sm_sched* utility.) Invoke these commands from the ***BASEDIR****/smarts/bin* directory in the Network Protocol Manager install area:

```
▼ % ./dmctl -s <NPM_domain_manager_name> invoke
ICF_TopologyManager::ICF-TopologyManager discoverPending ▲
```

```
▼ % ./dmctl -s <NPM_domain_manager_name> invoke
ICF_TopologyManager::ICF-TopologyManager discoverAll ▲
```

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

Index

A
ASL
  custom-end-post.asl 19
  custom-end-system.asl 19

B
BASEDIR ix

C
custom-end-post.asl 19
custom-end-system.asl 19

D
Discovery
  Custom scripts
    custom-end-post.asl 19
    custom-end-system.asl 19
  Discover Pending command 33
  Discovery All command 33
  dmctl 33
  Full discovery interval 32
  Manual rediscovery 33
  Pending discovery interval 32
  Pending Elements list 16, 31
  Rediscover command 33
Discovery Progress window 15, 16
dmctl 33
Domain Manager Administration Console 14
  Discover All command 33
  Discover Pending command 33
  Rediscover command 33

F
Full discovery interval 32

I
InCharge IP Availability Manager 1
InCharge Service Assurance Manager 1

P
Pending discovery interval 32
Pending Elements list 16, 17, 31
  Contents description 17
  Rediscovery 17
  Remove 17
  Show Discovery Time 17

R
Rediscover command 33
Rediscovery
  dmctl 33
  Full discovery interval 32
  Pending discovery interval 32
  Starting manually 33

S
serverConnect.conf 13
sm_edit 20, 23

T
Technical Support xi