HP Operations Manager

Software Version: 9.22
UNIX and Linux operating systems

HPOM Concepts Guide

Document Release Date: December 2016
Software Release Date: December 2016

Legal Notices

Warranty

The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

The information contained herein is subject to change without notice.

Restricted Rights Legend

Confidential computer software. Valid license from HP required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Copyright Notice

Trademark Notices

Adobe® is a trademark of Adobe Systems Incorporated.

Microsoft® and Windows® are U.S. registered trademarks of Microsoft Corporation.

UNIX® is a registered trademark of The Open Group.

Documentation Updates

The title page of this document contains the following identifying information:

- Software Version number, which indicates the software version.
- Document Release Date, which changes each time the document is updated.
- Software Release Date, which indicates the release date of this version of the software.

To check for recent updates or to verify that you are using the most recent edition of a document, go to: **https://softwaresupport.hpe.com**

This site requires that you register for an HP Passport and sign in. To register for an HP Passport ID, go to: **https://cf.passport.hpe.com/hppcf/createuser.do**

Or click the **the Register** link at the top of the HP Software Support page.

You will also receive updated or new editions if you subscribe to the appropriate product support service. Contact your HP sales representative for details.

Support

Visit the HP Software Support Online web site at: **https://softwaresupport.hpe.com**

This web site provides contact information and details about the products, services, and support that HP Software offers.

HP Software online support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued support customer, you can benefit by using the support web site to:

- Search for knowledge documents of interest
- Submit and track support cases and enhancement requests
- Download software patches
- Manage support contracts

- Look up HP support contacts
- Review information about available services
- Enter into discussions with other software customers
- Research and register for software training

Most of the support areas require that you register as an HP Passport user and sign in. Many also require a support contract. To register for an HP Passport ID, go to:

https://cf.passport.hpe.com/hppcf/createuser.do

To find more information about access levels, go to:

https://softwaresupport.hpe.com/web/softwaresupport/access-levels

HP Software Solutions Now accesses the HPSW Solution and Integration Portal Web site. This site enables you to explore HP Product Solutions to meet your business needs, includes a full list of Integrations between HP Products, as well as a listing of ITIL Processes. The URL for this Web site is **https://softwaresupport.hpe.com**

Contents

Chapter 1: HPOM Overview 17
In this Chapter 17
Who Should Read this Chapter 17
What this Chapter Does 17
HPOM Concepts 17
Benefits of HPOM 17
Client-Server Concept 18
Management Server 19
Managed Nodes 20
Intercepting Messages 20
Monitoring Performance 20
Comparing Messages 20
Logging Messages 21
Buffering Messages 21
Correcting Problems 21
Configuring Nodes 21
Basic Permissions and Basic User Types 21
Basic Permissions 21
Basic User Types 22
HPOM Features 22
Registering Problems 22
Solving Problems 23
Documenting Solutions 23
Generating Reports 23
HPOM Functions 26
Events 26
Correlating Events 27
Messages 27
Intercepting Messages 27
Applying Policy Conditions to Messages 28
Linking Messages Logically 28
Processing Messages 28
Managing Messages 30

Filtering Messages 30
Classifying Unmatched Messages 30
Formatting Messages 30
Responding to Messages 31
Defining Message Groups 31
Actions 31
Automatic Actions 32
Operator-Initiated Actions 33
Applications 34
HPOM Users 34
User Roles 34
Multiple Operators 35
Access Restrictions 35
User Profiles 35
Administrator 35
Operators 36

Chapter 2: Configuring and Maintaining HPOM 37
In this Chapter 37
Who Should Read this Chapter 37
What this Chapter Does 37
Administrator Environment 38
Securing Your Environment 38
System Security 39
Security Techniques 39
HPOM Security Methods 39
Organizing Managed Nodes 39
Building Managed Nodes 39
Types of Managed Nodes 40
Managed Node Arrangements 40
HPOM Node Bank 41
HPOM Node Hierarchy 41
Setting Up a Node Hierarchy 42
Applying Actions to All Nodes in a Hierarchy 42
Actions You Can Apply to All Nodes 42
Actions You Cannot Apply to All Nodes 42

Conditions for Applying Actions to All Nodes ... 43
Adding Nodes ... 43
Adding Internal Nodes ... 43
Characteristics of Internal Nodes ... 43
Reasons Not to Install HP Operations Agents ... 44
Adding External Nodes ... 44
Conditions for Adding External Nodes ... 45
Characteristics of External Nodes ... 45
Adding Nodes by Using the Administration UI ... 46
Setting up Security for Different Types of Managed Node ... 47
Managing Disabled Nodes ... 47
Configuring Node Groups ... 48
Determining the Status of Nodes ... 48
Assigning Categories to Node Groups ... 48
Managing Policies in HPOM ... 49
HPOM Policies ... 49
Policy Types ... 50
Multiple Policy Types ... 51
Policy Versions ... 51
Managing Conflicts between Policy Versions ... 53
Policy Data Exchange Model ... 54
Updating Policy Assignments ... 54
Managing Policy-Assignment Conflicts ... 55
Assigning Categories to Policies ... 56
Policy Type Callbacks ... 56
Edit Callback ... 58
Check Callback ... 58
Deploy Callback ... 58
Cleanup Callback ... 59
Using ConfigFile Policies to Run Callbacks on Agents After Deployment ... 59
Policy Groups ... 59
Default Policy Groups ... 59
Managing of Subagents in HPOM ... 60
Subagent Policies ... 60
Upgrade Considerations ... 61
Organizing Message Groups ... 61
Adding Message Groups ... 61

Reviewing Message Groups ... 62
Organizing Applications ... 62
Grouping Applications ... 62
Adding Applications ... 62
Customizing the Application Startup ... 63
HPOM Licenses ... 65
Types of Licenses ... 65
License Verification ... 65
Target Connector Count ... 65
License Notification ... 66
Support for Small Environments ... 66
License Availability ... 67
Setting Up Users and User Profiles ... 67
Adding Users ... 67
Adding Operators ... 68
Operator Responsibilities ... 68
Configuring Operators ... 68
Configuring Operator Profiles ... 68
Default Operator Profile ... 69
Assigning Message and Node Groups ... 69
Guidelines for Assigning Groups to Operators ... 69
Defining Different Layers of Operator Responsibility ... 70
Assigning an Operator's Managed Node Hierarchy ... 70
Assigning Tools to an Operator ... 70
Defining a Toolset for an Operator ... 70
Verifying the Operator Has the Proper Tools ... 71
Assigning Applications and Groups to Users ... 71
Assigning User Profiles ... 71
Configuring User Profiles ... 72
Non-root Operation ... 72
Requirements for Non-root Operation ... 73
Limitations of Non-root User ... 73
Non-root User Configuration ... 74
Non-root User Privileges and Capabilities ... 74
Updating the HPOM Configuration ... 75
Distributing the Configuration ... 75

Distributing Parts of the Configuration 75
Preparing for Software Distribution 76
Distributing the Configuration to Managed Nodes 76
Forcing Updates 77
Distributing Policies to Managed Nodes 77
Policies Installed by HPOM 78
Avoiding Duplicate Policy-Node Combinations Automatically 78
Distribution Tips 78
Updating Only Those Nodes Requiring an Update 79
Reducing Nodes and Priorities During Distribution 79
Manually Distributing Policies 79
Distributing Without Operator Configurations 80
Synchronization of the Configuration Changes 80
Synchronizing the GUI After Reconfiguration 81
Accessing Up-to-Date Content Automatically 82
Restarting Sessions Manually 82
Locking Components During Transactions Automatically 82
Backing Up and Restoring Data 82
Backing Up Data 82
Comparison of Backup Methods 83
Restoring Data 84
Restoring an HPOM Database 84
Message Ownership 85
Marking and Owning a Message 85
Ownership Display Modes 85
Ownership Modes 86
Generating Reports 87
Reporting Tools 87
HPOM Reports 87
Administrator and Operator Reports 87
Message, Error, and Configuration Reports 88
Service Reports 88
Generating Reports 89
Creating PGM and INT Reports 89
Format of Internal Reports 89
Defining Your Own Reports 89
Integrating Your Own Reports 90

Chapter 3: HPOM Managed Nodes Concepts 91
In this Chapter 91
HTTPS Agent Overview 91
HP Operations HTTPS Agent Architecture 93
HTTPS Communication in HPOM 94
Advantages 94
Firewall Friendly 94
Secure 95
Open 97
Scalable 97
Communication (Broker) Architecture 97
Security Concepts 99
HTTPS-based Security Components 99
Certificates 101
HP Certificate Server 101
Certification Authority 101
Remote Actions 102
Roles and Access Rights 103
About Roles 103
About Access Rights 103
Authorization Mappings 104
Managing HTTPS Nodes 106
Configuration Deployment to HTTPS Nodes 107
Policy Management 107
Instrumentation Management 107
Manual Installation of Policies and Instrumentation 108
HTTPS Agent Distribution Manager 108
Configuration Push 109
Delta Distribution 110
Remote Control of HTTPS Nodes 110
Heartbeat Polling 110
Creating and Distributing Certificates 111
Virtual Nodes in HPOM 111
Terminology 112
General High Availability Terms 112
Cluster Terms Used in HPOM 112

Virtual Node Concepts ... 113
Proxies in HPOM ... 114
Tracing HPOM ... 116
Tracing HPOM ... 116
HP-Style Tracing Overview ... 116
HPOM Trace-Enabled Applications ... 117
Server and Agent Applications ... 119
HP BTO Software and HPOM Specific Components ... 119
HPOM Specific and XPL Standard Categories ... 121
Firewalls and HTTPS Communication ... 123
Contacting an Application on the Internet from an Intranet Using an HTTP Proxy ... 124
Contacting an Application on the Internet from an Intranet Without an HTTP Proxy ... 124
Contacting an Application Within a Private Intranet from an HP Operations Application on the Internet ... 124
Contacting an Application Within a Private Intranet from an HP Operations Application on the Internet Without Using HTTP Proxies ... 125
Configuring HTTPS-Based Communication ... 125

Chapter 4: Implementing Message Policies ... 127
In this Chapter ... 127
Who Should Read this Chapter ... 127
What this Chapter Does ... 127
Message Management ... 128
Centralizing Actions ... 128
Detecting Problems Early ... 128
Improving Productivity ... 128
Distributing Policies ... 128
Consolidating Messages in the Browser ... 128
Managing Message Source Policies ... 129
Elements of a Message Source Policy ... 129
Configuring Message Source Policies ... 130
Message Source Policies ... 131
Creating Policies for Message Sources ... 132
Organizing Policy Groups ... 132
Advantages of Policy Groups ... 132
Listing Policy Groups ... 133

Creating Policy Groups 133
Regrouping Messages 133
Assigning Policies 134
Assigning Policies to Managed Nodes 134
Distributing Assigned Policies 134
Distributing Message Source Policies 135
Evaluating Message Sources 135
Where to Look for Messages 135
How to Evaluate Messages 135
Evaluating the Severity of Messages 136
Starting With Message Catalogs 136
Collecting Messages 136
Creating Message Status 136
Intercepting Messages 137
Processing Messages 138
How Messages Are Processed by Policies 140
Setting Message Defaults 140
Configuring Multiple Policies 141
Processing Multiple Policies at the Same Time 141
Configuring Your Own Application-Specific Policies 144
Filtering Messages with Conditions 144
Filtering Message Sources 144
Processing Messages on the Management Server 145
To Set Up Message Conditions 146
Message and Suppress Conditions 147
Conditions That Can Be Applied to Events 148
Comparing Incoming Messages with Match Conditions 148
Pattern Matching in Messages 150
Pattern Matching with Mathematical Operators 150
Pattern Matching Without Case-Sensitivity 150
Examples of Pattern-Matching Conditions 150
Details of Pattern-Matching Expressions 151
Numeric Range Operators 154
Less Than or Equal To (-le) Operator 154
Less Than (-lt) Operator 155
Greater Than or Equal To (-ge) Operator 155
Greater Than (-gt) Operator 155

Equal To (-eq) Operator 155
Not Equal To (-ne) Operator 156
To Insert Expression Symbols in Pattern-matching Expressions 156
Variables and Parameters of Pattern-matching Expressions 156
Rules Used by HPOM to Assign Strings to Variables 157
Using Sub-patterns to Assign Strings to Variables 157
Configuring a Message Policy Condition to Match a Multiline Message 158
Displaying Matched Messages 158
Assigning Message Settings 158
Adding Custom Message Attributes to Your Message 159
Adding Instructions to Your Message 160
Responding to a Message 161
Responses 161
Configuring Automatic Annotations and Acknowledgments 162
Strategies for Optimal Message Filtering 162
Filtering Messages 162
Optimizing Performance 163
Organizing Conditions into a Sequence 163
Deploying Suppress Unmatched Conditions 163
Reducing the Number of Messages 164
Correlating Messages and Events 164
Message Keys 166
Guidelines for Effective Message Keys 166
Guidelines for Effective Message Key Relations 167
Recommendations for Message Key Correlation Patterns 168
Automating Standard Scenarios 168
State-Based Browsers 169
Acknowledging Messages with Message Keys 169
Annotating Acknowledged and Acknowledging Messages 169
To Generate a Default Message Key and Message Key Relation 170
Sending a Reset Message Automatically 170
Example of an Automatic Reset Message 171
Suppressing Duplicate Messages 173
Verifying Suppression Types 174
Type of Suppression Settings 176
Suppression Based on Time 176
Suppression Based on Counter 177

Suppressing Duplicate Messages on the Management Server 178
Enabling Duplicate Message Suppression on the Management Server 179
Suppressing Duplicate Messages in Flexible Management Environments 179
Updating Severity and Message Text of Duplicate Messages 180
Logging Messages 180
Regrouping Messages 181
Defining a Regroup Condition 182
Examples of Regroup Conditions 182
Log File Messages 183
Log File Encapsulator 183
Logfile Entry Policies 184
Monitoring Log Files on Nodes 185
Monitoring Log Files on External Nodes 185
Defining Advanced Options for Message Policies 186
Specifying Conditions for Messages 186
LogFile Entry Policy Body Example 186
HPOM Message Interface 187
Messages from Threshold Monitors 188
Starting Corrective Actions in Response to Messages 188
Integrating Monitoring Programs or Utilities 188
How the Monitor Agent Works 188
Monitoring Program or MIB Objects with Polling Intervals 189
Monitoring External Objects with opcmon 190
Monitoring Performance Metrics 191
Performance Metrics 191
Setting Up Performance Thresholds 192
Selecting Variables to Monitor 193
Selecting a Threshold Type 193
Selecting a Message Generation Policy 194
Message Generation with Reset 194
Message Generation Without Reset 195
Continuous Message Generation 195
Short-Term Peaks 196
Integrating a Threshold Monitor 196
Integrating a New Threshold Monitor 196
Configuring a Threshold Monitor 198

Default Threshold Monitors ... 199
To Set Conditions for Advanced Monitoring ... 199
Threshold Monitoring with Multiple Conditions ... 199
Examples of Threshold Monitor Conditions ... 201
SNMP Traps and Events ... 202
Defaults for Intercepting Traps and Events ... 202
Adding SNMP Trap Policies ... 202
Filtering Internal HPOM Error Messages ... 204
Event Correlation in HPOM ... 204
How Event Correlation Works ... 205
Where to Correlate Messages ... 206
Correlating Messages from Different Sources ... 207
HPOM Event Interceptor ... 207
Correlating Events in NNMi Before They Reach HPOM ... 208
Synchronizing HPOM and NNMi Event Correlation ... 208
Correlating Messages on Managed Nodes ... 208
Correlating Messages on the Management Server ... 210
Correlating Messages in Flexible Management Environments ... 211
Accessing External Data ... 212
Data and Fact Stores ... 212
Updating Fact Stores and Data Stores ... 214
Automating Distribution to the UNIX Nodes ... 215
Annotate Mechanism ... 216
Internal Annotation Server ... 217
Annotation Script ... 218
User-built Annotation Server ... 219
Importing ECS Circuits ... 219
Modifying the ECS Circuit Names ... 221
Service Hours ... 222
Buffering Messages ... 222
Unbuffering Messages Automatically ... 223
Unbuffering Messages Manually ... 223
Defining Service Hours ... 223
Scheduled Outages ... 223
Scheduling an Outage ... 224
Defining a Scheduled Outage ... 224

Configuring Service Hours and Scheduled Outages 224
Setting Custom Message Attributes Based on Message Selection Criteria 224

Chapter 5: Scalable Architecture for Multiple Management Servers 226
In this Chapter 226
Who Should Read this Chapter 226
What this Chapter Does 226
Flexible Management 227
Default Setup 227
Primary Manager 227
Advantages of Flexible Management 227
Follow-the-Sun Control 228
Competence Centers 230
Distributing Responsibility in Competence Centers 231
Configuring Competence Centers 231
Backup Servers 232
Configuring Backup Servers 232
Distributing Configurations and Policies 232
Management Hierarchies 232
Management Profiles in the Management Hierarchies 233
Setup Ratio in Management Hierarchies 233
Management Responsibilities in Domain Hierarchies 233
Regional Management Servers 234
Central Server 234
Configuring the Management Server 234
Configuring the Central Server as Action-allowed Manager 234
Specifying the Central Server as Secondary Manager 235
Configuring the Central Management Server 235
Configuring Responsible Managers 236
Creating a Configuration File 236
Distributing the Configuration File 236
Message Target Rules 237
Parts of a Message Target Rule 237
Example of a Message Target Rule for Printing Group 237
Example of a Message Target Rule for a Database Group 238
Time Policies 238

Setting Time Intervals 238
Configuring Time-Indifferent Policies 239
Specifying the Primary Manager 239
Switching to a Secondary Manager 239
Enabling a Secondary Manager to Execute Actions 240
Reversing a Manager Switch 241
Delegating Manager Responsibilities 241
Switching to a Backup Manager 241
Specifying Action-Allowed Managers 241
Distributing Configurations to Other Servers 242
Uploading Configuration Data on the Local Management Server 243
Forwarding Messages Between Management Servers 243
Normal and Notification Messages 244
Message Forwarding Policy 245
Message Distribution Lists 246
Controlling the Size of Distribution Lists 246
Connecting Management Servers to Trouble Ticket Systems 248
Managing Forwarded Messages 248
Planning Your Message Forwarding Strategy 248
Troubleshooting Problems in the Message Forwarding Policy 250
Scalability Scenarios 251
Scenario 1. Single Server Managing a Set of Nodes 251
Scenario 2. HP Operations Agents Monitoring IP Devices 252

Appendix A: Policy Body Grammar 253
In this Appendix 253
Policy Body Grammar 253

Send Documentation Feedback 262

Chapter 1: HPOM Overview

In this Chapter

This chapter introduces operators to the concept, functionality, and structure of HP Operations Manager (HPOM).

Who Should Read this Chapter

This chapter is designed for HPOM operators.

What this Chapter Does

This chapter describes the following:

- "HPOM Concepts" below
- "HPOM Features" on page 22
- "HPOM Functions" on page 26
- "HPOM Users" on page 34

HPOM Concepts

HPOM is a distributed client-server software solution designed to help system administrators detect, solve, and prevent problems occurring in networks, systems, and applications in any enterprise. HPOM is a scalable and flexible solution that can be configured to meet the requirements of any information technology (IT) organization and its users. System administrators can expand the applications of HPOM by integrating management applications from HPOM partners or other vendors.

Benefits of HPOM

HPOM helps you do the following:

- **Maximize Network**

  Maximize the availability of network components.
- **Reduce Downtime**

Reduce the time lost by end-users as a result of system downtime.

- **Decrease Workload**

  Reduce unnecessary user actions by automatically solving problems.
- **Prevent Problems**

  Reduce the number of problems through preventive actions.
- **Decrease Delays**

  Decrease the time needed to solve problems.
- **Reduce Costs**

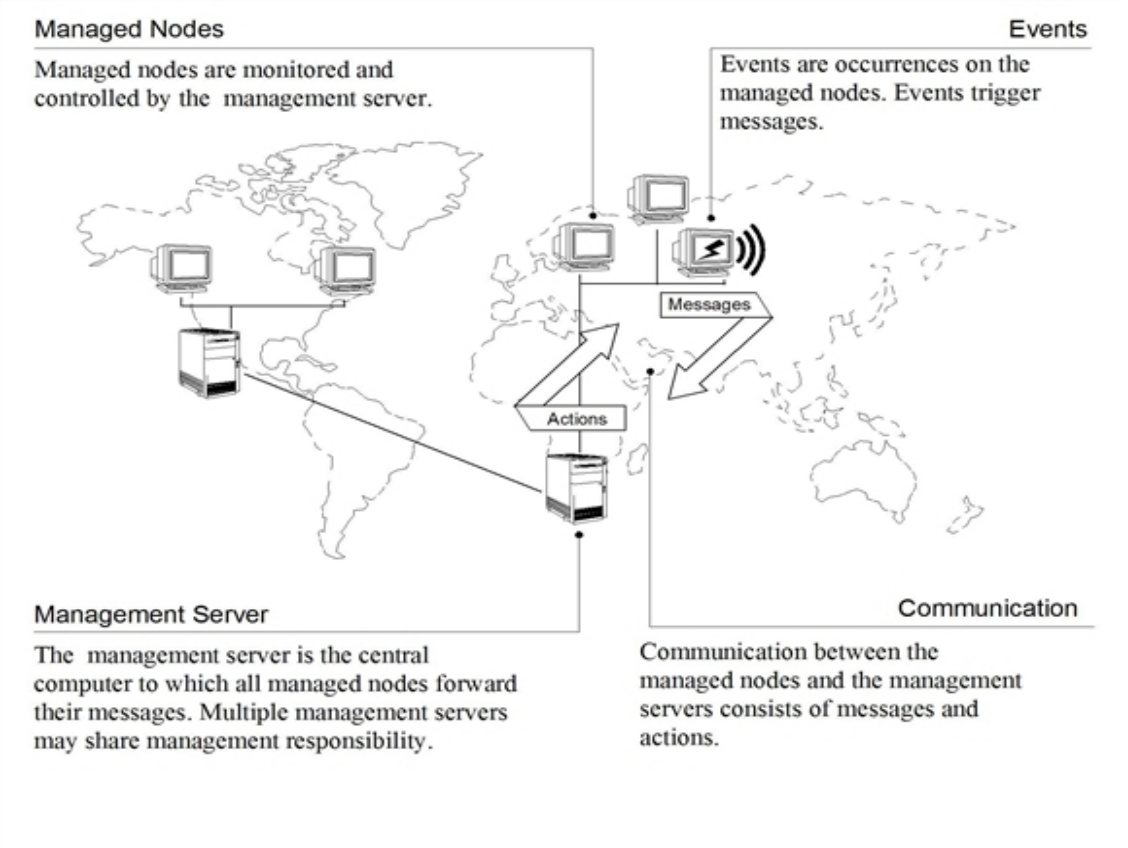
  Reduce the cost of managing the client-server environment.

Client-Server Concept

The HPOM management concept is based on communication between **a management server** and **managed nodes**. Management server processes running on the central management server communicate with **HP Operations agent processes** running on managed nodes throughout the environment. The HP Operations agent processes collect and process **events** on the managed nodes, then forward relevant information in the form of **HPOM messages** to the management server. The management server responds with **actions** to prevent or correct problems on the managed nodes.

Figure 1 shows the management concept of HPOM.

Figure 1: HPOM Client-Server Concept



The agent on the management server also serves as the **local managed node**.

A **database** serves as the central data repository for all messages and configuration data. You can use this runtime and historical data to generate reports. Historical data can also be helpful when creating instructions to help operators solve problems caused by similar events, and to automate certain problem resolution processes. The database processes run on the management server.

Management Server

The management server performs the central processing functions of HPOM. The entire software package, including the complete current configuration, is stored on the management server.

The management server does the following:

- **Collects Data**

  Collects data from managed nodes.
- **Manages Messages**

  Manages and groups messages.
- **Manages Actions**

Calls the selected agent to:

- *Start actions*

  Start local automatic actions on the managed node.

- *Initiate sessions*

  Initiate sessions on managed nodes.

- **Manages History**

  Controls the history database for messages and performed actions.
- **Forwards Messages**

  Forwards messages to other management servers or to systems where HPOM is running.
- **Installs Software**

  Installs HP Operations agent software on managed nodes.

The management server also notifies the managed nodes about configuration changes and initiates any updates.

Managed Nodes

Managed nodes are computers that are controlled and monitored by HPOM. HPOM manages these nodes by installing and running agent processes on them.

Intercepting Messages

After installed and running, the **HP Operations agent software** reads log files and SNMP traps. If so configured, the **HPOM message interceptor** can intercept messages from any application running locally on the managed node.

Monitoring Performance

Performance values are **monitored at configurable intervals**, and messages can be generated when performance varies from limits.

HPOM can also monitor its **own processes**.

Comparing Messages

The HP Operations agent compares all messages with conditions in preconfigured policies, then forwards unexpected or important messages to the management server while ignoring unimportant messages. If so configured, the agent even suppresses duplicate or similar events. You determine your

message filtering policy either by modifying existing policies or by configuring your own set of policies and conditions.

Logging Messages

All messages can be **logged** locally on the managed node or written directly into the history database on the management server. This history functions enables you to examine all messages, even those you configured the system to disregard as unimportant.

Buffering Messages

If the management server is not reachable, messages are retained in a **storage buffer** until the management server can receive them again.

Correcting Problems

Corrective actions can be started locally on the managed node in response to a message, and can be stopped and restarted, if necessary.

Configuring Nodes

Your HPOM environment can be composed of different **types** of managed nodes (for example, nodes marked controlled, monitored, message-allowed, or disabled). You can also add a range of IP addresses, so all nodes become known when they become part of a specific network or are added manually.

Basic Permissions and Basic User Types

The default security settings on a file or folder can be described by summarizing the permissions granted to different user groups.

Basic Permissions

The permissions on a file or folder specify how it can be accessed and changed. These permissions apply to the basic user types as well as all of the Access Control List (ACL) default types. Only a file or folder owner can change basic permissions and set or modify ACLs.

Read Permission	Allows access to retrieve, copy, or view the contents of the object.

Write Permission	For a file, allows access to change the contents of the file. For a folder, allows access to create or delete objects from the folder.
Execute Permission	For a file, allows access to run the file (for executable files, scripts, and actions). For a folder, allows access to search and list the folder's contents.

To make an object that you have created available for everyone to use, but protect it so it is not inadvertently overwritten:

Change the file's properties, giving read and execute permission to owner, group, and other. Do not give anyone write permission.

Basic User Types

For basic permissions on a file or folder, the three types of users are:

Owner	The user who owns the file or folder. Only a system administrator (root user) can change the owner of a file or folder.
Group	Users who have been grouped together by the system administrator. For example, the members of a department might belong to the same group. This group is the owning group and usually includes the file or folder's owner.
Other	All other users on the system besides the owner and owning group.

For example, to make a folder private, change the folder's properties, giving yourself (the owner) read, write, and execute permission, but giving no permissions for group and other. After you give yourself these permissions, only you and the root user can view the contents of the folder.

HPOM Features

HPOM helps you solve problems in your computing environment proactively. These problems can occur anywhere within a heterogeneous distributed environment consisting of network elements, systems, and applications.

Registering Problems

HPOM notifies you when a problem is about to occur, and then provides you with the resources you need to avoid the problem. Likewise, HPOM notifies you when a problem has just occurred, and provides you with the resources you need to solve the problem.

For example, if an unauthorized user attempts to log on to a managed node, the node registers this problem in one of several possible ways. The node may write an entry to a system log file, send an SMNP trap, or use an application programming interface (API) to communicate directly with the management server.

In this example, the unauthorized user logon has written an entry in a log file. HPOM reads this log file, and uses preconfigured conditions to decide whether to generate a message. If the conditions allow a message to be generated, HPOM uses the entry in the log file to create a meaningful message, attaches attributes (additional information) to the message, and sends the complete message to the management server.

Solving Problems

On the management server, the message is displayed in a browser. The message tells you how severe the problem is, where (that is, on which managed node) the problem occurred, and what triggered the message. Depending on the type of action response you have configured for the event, the arrival of the message might trigger an automatic action executed immediately on the managed node. Or the message might trigger another message instructing the user to start a corrective action manually.

Documenting Solutions

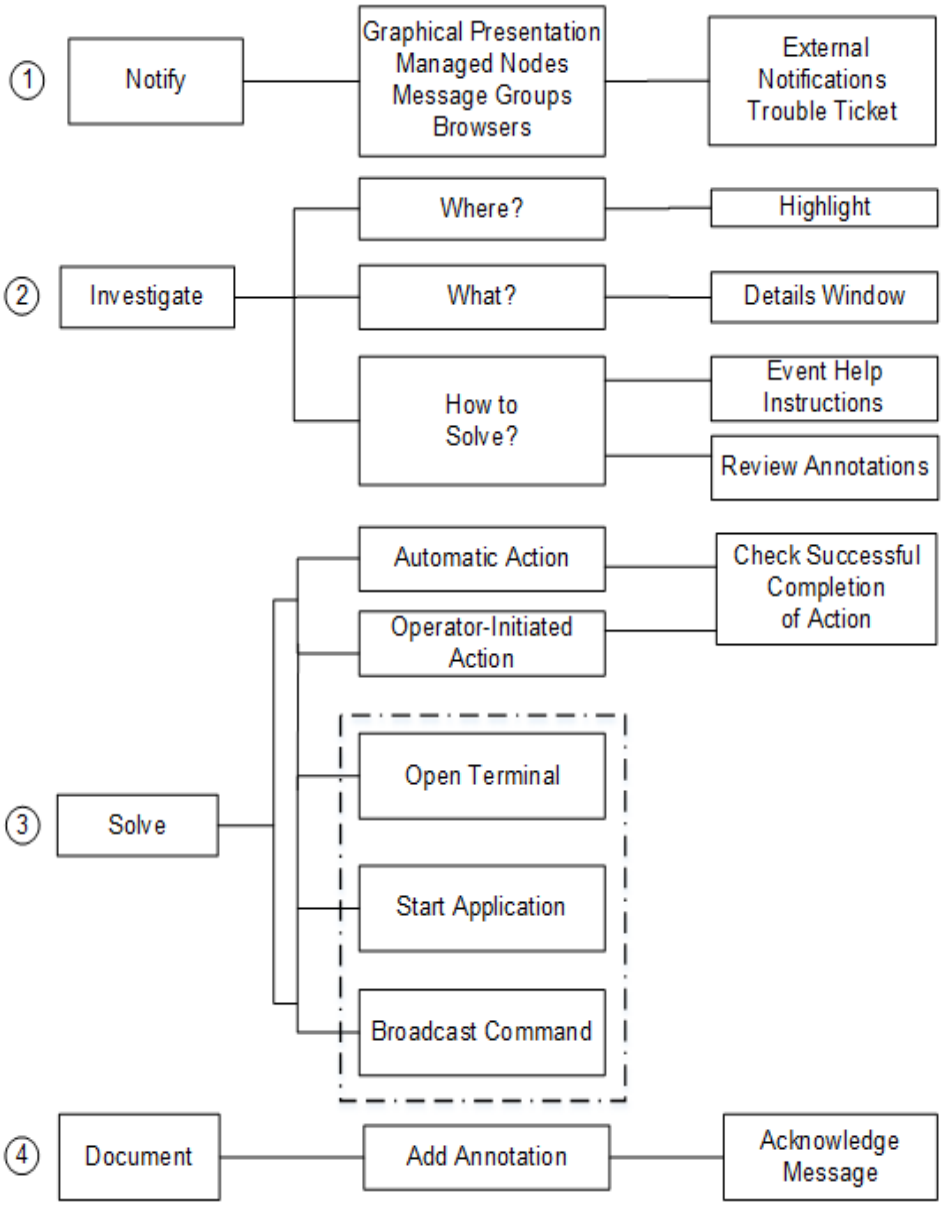
After the action has been completed successfully, the user can annotate the message with a comment, and then move the annotated message into the history database by acknowledging it.

Generating Reports

To help find information about unauthorized user logons and actions associated with them, you can also generate reports from the database.

The major areas of problem solving are shown in Figure 2. These problem solving areas are described in more detail in the *HPOM Java GUI Operator's Guide*.

Figure 2: Components of Problem Solving



The following is the key to Figure 2, describing the components of problem solving:

1. **Notify**

   HP Operations agents monitor log files and system activity.

   If an HPOM problem occurs, an HP Operations agent notifies you in one of the following ways:

   - Sends a message to the management server.
   - Changes the colors of node icons, based on the severity of the problem.

- Colors the Severity field in the message browser, or, if so configured, the entire message line to reflect the status of the message.
- Displays a message and its attributes (for example, time sent, status of actions, and so on).
- Forwards the message to external notification or trouble ticket services, if they are configured.

You can see the severity of the problem and the affected object at a glance. You can then review every detail of the problem and solve it.

2. **Investigate**

   Understand the problem and its cause. In large environments, it is crucial to be able to pinpoint problems quickly.

   HPOM helps you pinpoint trouble in your environment by providing a fast link between the HPOM system and the network management view.

3. **Solve**

   Initiate a corrective action to solve the problem.

   HPOM provides the following solutions:

   - *Automatic Actions*

     As soon as it receives an error message, HPOM starts corrective actions automatically. You can restart these automatic actions manually as many times as needed.

     Note: To use the service id within an automatic action you can use the variable *<$MSG_SERVICE>*.

   - *Operator-initiated Actions*

     As soon as they receive and review error messages, operators can start corrective actions manually. These corrective actions can also be stopped manually.

   - *User Instructions*

     As attachments to error messages, you can send specific problem solving instructions to users. These instructions should explain exactly how to solve the problem.

   - *History Logs*

     You can also use history logs (including message annotations) of related problems to track the techniques used to solve similar problems or previous occurrences of the same problem.

   - *Java GUI Console*

You can use the Java GUI console to start different types of applications or broadcast commands to multiple systems. Then you can begin corrective actions from the Java GUI console itself.

4. **Document**

   Close the problem and document the solution, for future reference.

HPOM Functions

The primary objective of HPOM is to monitor, control, and maintain systems in distributed heterogeneous environments.

HPOM performs the following tasks:

- **Events**

  Notes an event in your environment.
- **Reports**

  Generates a meaningful message, or report, about the event.
- **Actions**

  Responds to the event with an action.

HPOM uses messages to communicate with you. Messages are structured, readable pieces of information about system status, system events, or problem related to a managed node within the system. HPOM notifies you of a status change, an event, or a problem on a managed node by sending you a message. If the event that triggers the message is a problem, HPOM can start an action to correct the problem. The original message, the result of the corrective action, and other associated information (for example, user annotations) are stored in the database.

Events

An event is a particular fault or incident within the computing environment that occurs on an object. Typically, an event represents either a change in status or a threshold violation. For example, the status of a printer changes when the paper tray empties. Likewise, a threshold is violated when the available disk space falls below a certain level. Each of these occurrences is an event. For each of these events, you can create a message.

Many events are problems that must be corrected. However, there are events that do not require user action. For example, when a user logs on or off a system, the status of the system changes, an event occurs, but no user action is required.

Correlating Events

Events generate messages. The more events a system generates, the more messages users are likely to receive. “Event storms” overload the management server and can overwhelm the operator in charge.

Event correlation (EC) enables real-time processing of event groups, known as “event streams”. This real-time processing identifies relationships between event streams, and creates a smaller stream with more useful and more manageable information. This information can be used to diagnose and solve problems more effectively. Event correlation filters out duplicate or related events, replacing a set of related messages with a single message.

For information about supported agent and management server platforms, see the *HPOM Administrator’s Reference*. To better understand how event correlation works in HPOM, see "Implementing Message Policies" on page 127. To learn how to set up event correlation in HPOM, see the *HPOM Administrator’s Reference*.

Messages

Messages are structured chunks of information that are triggered by events. HPOM intercepts events, then creates messages to inform you of those events.

Intercepting Messages

HPOM intercepts messages from a variety of sources, including:

- **Log Files**

  Log file encapsulator extracts message information from application and system log files (Eventlogs, for example).
- **SNMP Events**

  SNMP event interceptor captures events on the management server and on selected agent platforms. For more information, see the *HPOM Administrator’s Reference*.
- **HPOM Messages**

  Message interface enables you to generate messages by an HPOM command or API (`opcmsg (1|3)`).
- **Monitored Objects**

  You can set up threshold levels for monitored objects. When measured values of monitored objects exceed configured threshold levels, HPOM generates messages.
- **User Applications**

  All applications that write messages to log files either use HPOM APIs or send SNMP traps that can provide information to HPOM.

Applying Policy Conditions to Messages

After an event has been identified, HPOM applies policy conditions, which act like filters, to determine whether to generate a message or to suppress the information about the event. If a message is generated, HPOM can completely restructure it (for example, to present the information to users in an understandable format). If a message reports a problem, you can define actions to solve that problem.

Linking Messages Logically

Messages may also be linked logically to one or more messages and automatic actions attached to the correlated output. For more information on the built-in message-correlation and filtering capabilities of HPOM, see "Implementing Message Policies" on page 127.

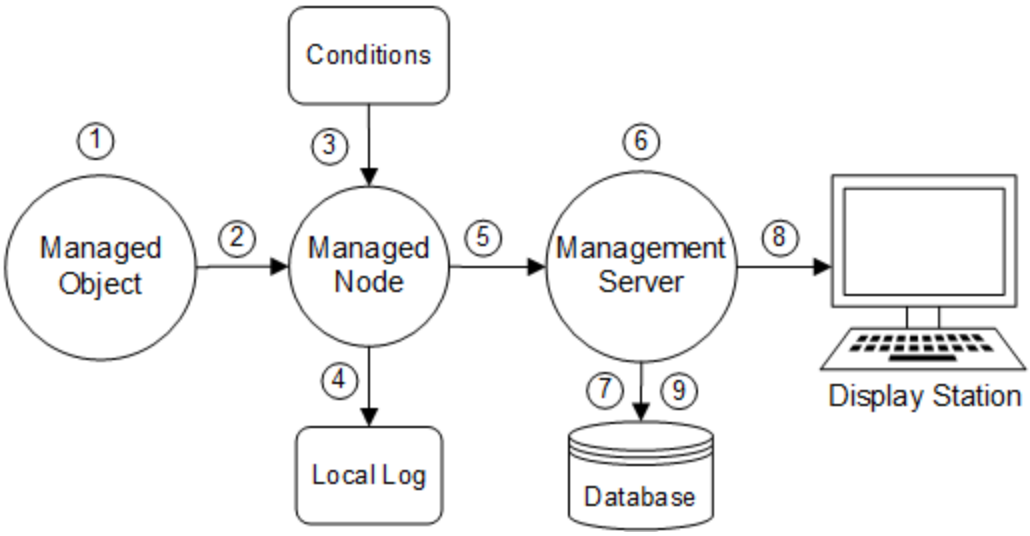
Processing Messages

HPOM uses messages to perform the following tasks:

- Communicate information about events.
- Inform users of status changes within the environment.
- Trigger corrective actions.

Figure 3 illustrates how HPOM processes messages.

Figure 3: Message Processing Flow



As illustrated in Figure 3, messages are processed as follows:

1. **Created on Managed Object**

   An event occurs on a managed object, and a message is created as a result. For example, a backup attempt fails because of an improperly loaded tape and creates a message.

2. **Received by Managed Node**

   The HP Operations agent on the managed node receives the message.

3. **Forwarded or Suppressed**

   The message is compared to filters. Messages matching suppress conditions or duplicate messages are suppressed. Other messages are forwarded.

4. **Logged**

   The message can be logged locally, if HPOM is so configured.

5. **Forwarded to Management Server**

   Messages matching filters are converted to the HPOM message format, and forwarded to the management server. If a local action is configured, it will be started.

6. **Processed by Management Server**

   The management server processes the message in one of the following ways:

   - *Regroup*

     Automatically assigns the message to another message group (regrouping).

   - *Start Actions*

     Automatically starts non-local actions configured for the message on the specified node.

   - *Forward*

     Forwards the message to external notification interfaces and trouble ticket service, if HPOM is so configured.

   - *Buffer*

     Buffers the message in the Pending Filtered Browser, if HPOM is so configured.

7. **Stored in Database**

   The active message is stored in the database.

8. **Displayed**

   The message is displayed in the Message Browser window in one or more HPOM display stations.

9. **Stored in History Database**

   When the message is acknowledged, it is removed from the active browser and put into the history database.

For more information about how operators can respond to messages, see the *HPOM Java GUI Operator's Guide*.

For more information about how administrators can configure messages and actions, see "Configuring and Maintaining HPOM" on page 37.

Managing Messages

The message management functions of HPOM can combine messages into logically related groups. A message group brings together messages from a variety of related sources, providing status information about a class of managed objects or services. For example, the message group BACKUP can be used to gather all messages related to system backups from sources such as backup applications and tape drives.

Filtering Messages

Other message management operations let you classify and filter messages, positively or negatively, to ensure that important information is displayed clearly:

- **Positive Filters**

  Forward messages that match a specified pattern to the operator.
- **Negative Filters**

  Suppress messages that match another specified pattern.

Suppressed messages can be stored in a local log file. You can later use the log file to analyze trends, review filter applicability, and track status patterns of managed objects.

Classifying Unmatched Messages

HPOM classifies messages that do not match a filter as "unmatched". Unmatched messages often occur with new or undefined sources. The unmatched category indicates that no relevant message classification has been possible.

You can do one of three things with unmatched messages:

- Log locally
- Forward to the management server
- Ignore

Formatting Messages

All messages forwarded to the central management system use the same format in a message browser. The HPOM color coding highlights the message severity. By default, only the `Severity`

column of the message browser is colored to reflect the severity of the message. However, you can configure HPOM to color each message line within the message browser. You can also configure HPOM to start external notification services, such as pagers or automatic calling systems.

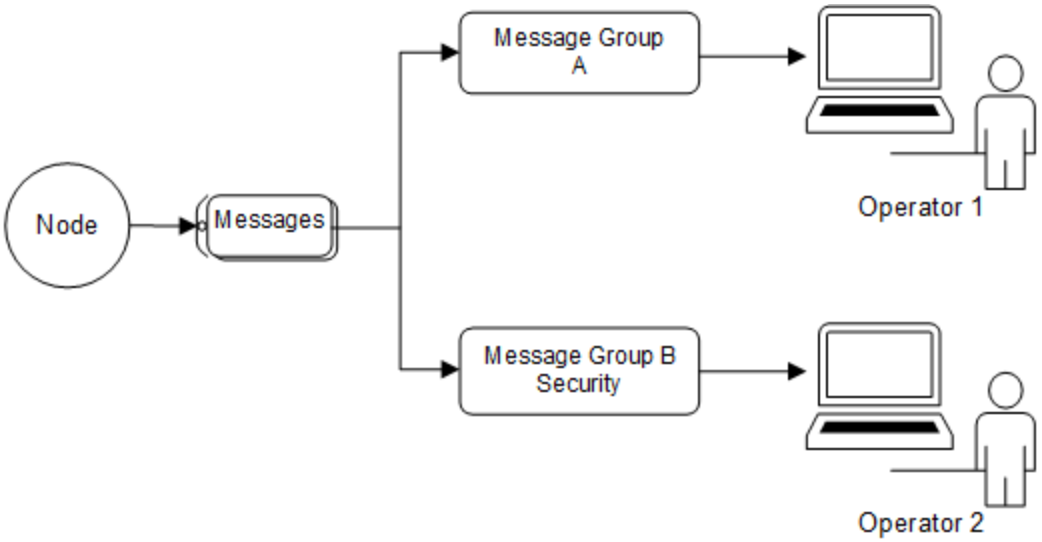
Responding to Messages

The operator uses the message browser as a starting point to review and respond to each message. Here the operator finds all related message information, including the availability and status of all preconfigured corrective actions. A service for documenting these or any other actions is also included.

Defining Message Groups

The HPOM administrator can assign messages from sensitive applications or functions to a single message group. In the example shown in Figure 4, two operators share responsibility for a single node that issues sensitive messages requiring restricted access. Network security is maintained by assigning the restricted-access messages to a message group called *Security*, then assigning this group to the operator with a security clearance. The other operator, who does not have a security clearance, does not see messages from the group *Security*.

Figure 4: Security-sensitive Messages Sent to Only One Operator



Actions

An action is a response to a message. If the event that creates the message is a problem, HPOM can start an action to correct the problem.

Actions are also used to perform daily tasks, such as starting an application every day on the same nodes. An action can be a shell script, program, command, application start, or any other response required.

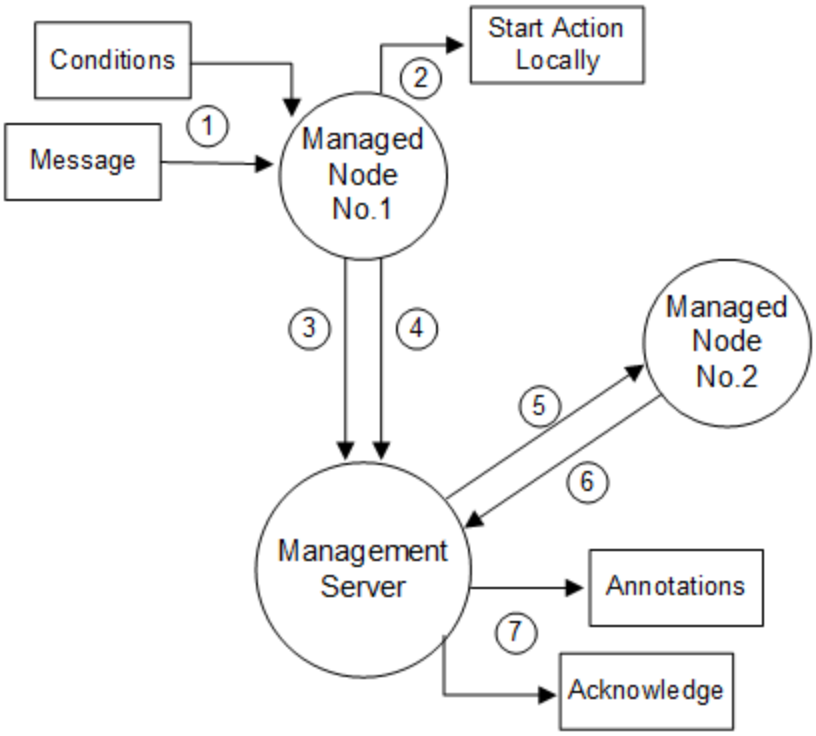
HPOM offers the following types of actions:

- Automatic actions
- Operator-initiated actions
- Applications

Automatic Actions

Automatic actions are preconfigured, message-linked responses to problems. Automatic actions do not require operator interaction. HPOM starts these actions as soon as a message is received. The operator can manually restart or stop them if necessary.

Figure 5: Starting an Automatic Action



As shown in Figure 5, automatic actions are processed as follows:

1. **Intercept Message**

   The message is intercepted on the managed node according to the conditions defined.
2. **Start Action**

   If the target node is *Node No.1*, the action is started locally.
3. **Report Results**

Node No.1 reports the results of the action to the management server.

4. **Inform Management Server**

   If the target node is *Node No.2*, *Node No.1* informs the management server.

5. **Start Action**

   The management server sends instructions to *Node No.2* to start the action.

6. **Report Results**

   Node No.2 reports the results of the action to the management server.

7. **Log Annotations**

   If the message is so configured, annotations about the execution of the automatic action are passed to the management server for logging. Logging can also be automatically acknowledged after successful completion of the action.

Operator-Initiated Actions

Like automatic actions, operator-initiated actions are preconfigured, message-linked responses to problems. These actions are started and stopped by an operator.

Administrators may choose to configure an operator-initiated action for a message instead of an automatic action because of the following:

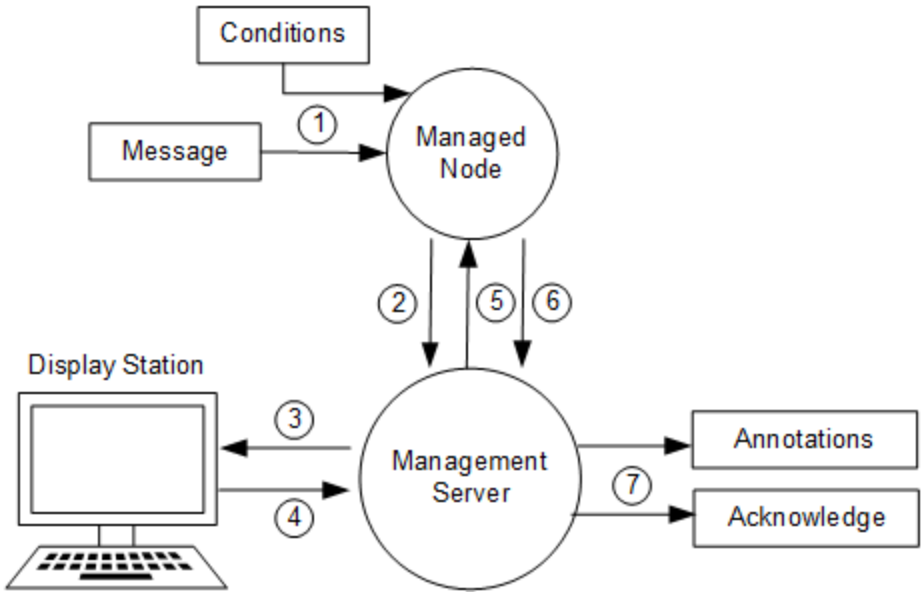
- **Manual Operations**

  Operator may need to perform manual operations in conjunction with the action.

- **Preconditions**

  Starting the action may be contingent on conditions within the environment that must first be checked by the operator.

Figure 6: Starting an Operator-initiated Action



As shown in Figure 6, an operator-initiated action is processed as follows:

1. **Interception**

   The message is intercepted on the managed node according to the setup conditions. For example, the message can prompt either an operator-initiated or an automatic action.
2. **Forwarding**

   The message is forwarded to the management server.
3. **Display**

   The message is sent to the display station of the responsible operator. A message attribute in the Message Browser window indicates that the message has a preconfigured operator-initiated action.
4. **Action**

   The operator starts the action by clicking a button in the message browser.
5. **Instructions**

   Instructions are sent to the managed node to start the action.
6. **Reporting**

   The managed node reports the results of the action to the management server.
7. **Logging and Acknowledgement**

   If they are so configured, annotations about the execution of operator-initiated actions are passed to the management server for logging. If the message has been so configured, it is acknowledged automatically after successful completion of the action.

Applications

Applications are scripts or programs that have been integrated into HPOM. Unlike operator-initiated and automatic actions, which are directly associated with a message and can be started or stopped from the browser windows, applications are tools that are available in the operator's `Applications` folder. For details, see the *HPOM Java GUI Operator's Guide*.

HPOM Users

The HPOM user concept distinguishes real users, such as the HPOM administrator and the HPOM operators, from user profiles. User profiles describe the configuration of abstract users. These abstract user configurations can be used to create real user configurations.

User Roles

The primary user roles of HPOM are:

- **HPOM Administrator**

  User with unlimited authority. Primarily responsible for installing and configuring the HPOM software, and establishing the initial operating policies and procedures.
- **Operator**

  User with no authority. Uses HPOM almost continuously to maintain, manage, monitor, and control systems and objects.

Multiple Operators

Adaptable to different organizational rules and requirements, HPOM permits multiple operators on a single management system. Operators are then assigned specific responsibilities and capabilities according to their individual know-how. Depending on the size of the computing environment managed by HPOM, two of these roles might be performed by a single person.

Access Restrictions

Access to the HPOM user interface is restricted. All operators and administrators must provide the correct logon name and password before gaining access to their customized HPOM user interface. These HPOM passwords are not related to the operator's system logon name and password.

User Profiles

User profiles are useful in large, dynamic environments with many HPOM users. You can configure profiles of abstract users, then assign these predefined profiles to real HPOM users. Profiles allow you to quickly set up users with a default configuration. You can create as many profiles as you need, and arrange them in a user profile hierarchy. For details, see "Configuring User Profiles" on page 72.

Administrator

The HPOM administrator, `opc_adm`, has many tasks and responsibilities within the HPOM working environment.

The administrator does the following:

- **Customizes User Environments**

  Defines a custom environment for each user. Manages all installation, configuration, and customization adaptations. These adaptations to the system add or change operators, nodes, retrieved messages, and so on.
- **Maximizes Operator Efficiency**

Matches corrective actions to specific events, and provides individual instructions for other events.

- **Delegates Responsibility**

  Defines responsibility and capability sets, and decides which tools the operator needs to maintain the assigned nodes and perform the required tasks.

- **Develops Guidelines**

  Develops the guidelines to implement a message policy. The administrator defines responsibility for policies or policy groups.

- **Maintains History**

  Maintains and reviews the HPOM history data. This history tracking enables the administrator to intelligently modify or develop automatic and operator-initiated actions, provide specific event instructions, and track recurring problems. For example, reviewing history data would reveal which nodes have consistently high disk space use.

- **Solves User Problems**

  Acts as any operator to verify their configuration and help them resolve any problems they may have with the system.

- **Expands HPOM**

  Extends the scope of HPOM by integrating additional applications and monitored objects, and ensures consistent presentation and invocation of services by registering new applications.

- **Maintains HPOM**

  Maintains the software, and defines management processes and security policies. For more information on HPOM security, see "Securing Your Environment" on page 38 of this guide, or the *HPOM Administrator's Reference*.

Operators

The HPOM operator, `opc_op`, controls system management functions. Every operator's environment consists of a set of managed nodes. These nodes are the basis for daily operator tasks, such as application startups. The nodes also provide the information operators use to solve problems.

HPOM operators have customized views of their own managed environments. For example, one operator might be responsible for all nodes at a facility. Another operator might be responsible for a subset of nodes at another facility. By creating task-orientated environments, HPOM operators see the information only from systems and objects under their control.

For more detailed information on the default HPOM operators, see "Setting Up Users and User Profiles" on page 67.

Chapter 2: Configuring and Maintaining HPOM

In this Chapter

This chapter explains how to configure various HPOM elements, such as managed nodes, node and message groups, and applications and operators.

Who Should Read this Chapter

This chapter is designed for HPOM administrators.

What this Chapter Does

This chapter explains the following topics to HPOM administrators:

- "Administrator Environment" on the next page
- "Securing Your Environment" on the next page
- "Organizing Managed Nodes" on page 39
- "Managing Policies in HPOM" on page 49
- "Managing of Subagents in HPOM" on page 60
- "Organizing Message Groups" on page 61
- "Organizing Applications" on page 62
- "HPOM Licenses" on page 65
- "Setting Up Users and User Profiles" on page 67
- "Updating the HPOM Configuration" on page 75
- "Backing Up and Restoring Data" on page 82
- "Message Ownership" on page 85
- "Generating Reports" on page 87

Administrator Environment

The HPOM administrator environment is a superset of the HPOM operator environment. As an administrator, you can access all operator GUIs and configurations, as well as to additional administrative capabilities and command-line tools.

As an HPOM administrator, you can configure the following primary HPOM elements:

- Node Hierarchy Bank
- Node Group Bank
- Message Group Bank
- Application Bank
- User Bank
- User Profile Bank
- Message Source Policies

Securing Your Environment

To secure your environment, you need to investigate the following:

- **System Security**

  To improve overall security, you need to look first at system security and then investigate problems that relate to network security. System security covers the problems that need to be addressed to enable the HP Operations management server and managed nodes to run on a trusted system. For more information about system-level security policies, see the product documentation for the relevant operating system.
- **Network Security**

  Network security involves the protection of data that is exchanged between the management server and the managed node. HPOM protects this data by using methods that ensure the authentication of the parties in a connection. For more information about network security, see the *HPOM Administrator's Reference*.
- **HPOM Security**

  You need to investigate the security implications that are addressed during the configuration of HPOM itself. That is, you need to look at the security-related aspects of application setup and execution, operator-initiated actions, and so on. For more information on the working directories, file access and permissions of HPOM users, see the *HPOM Administrator's Reference*.

System Security

A secure or trusted system uses a number of techniques to improve security at the system level. These techniques must be taken into account when setting up and configuring the HPOM environment.

Security Techniques

The security techniques include the following system-level components:

- **Authentication**

  Strict password and user-authentication controls
- **Auditing**

  Auditing networking, shared memory, file systems, and so on
- **Terminal Access**

  Controlling access to terminals
- **File Access**

  Managing access to files

HPOM Security Methods

At the network level, HPOM protects data by using the following methods:

- **Authentication**

  Verifies the identity of the parties involved in a connection.
- **Auditing**

  Verifies that a message has not been changed since it was generated by a legitimate source.

For more information about network security, and specifically how HPOM addresses the problem of inter-process communication, see the *HPOM Administrator's Reference*.

Organizing Managed Nodes

Managed nodes are systems you want HPOM to monitor and control. Your environment can be composed of different types of managed nodes.

Building Managed Nodes

The initial default configuration of HPOM includes the management server as the only managed node. You add other nodes to the configuration. In this way, you build an HPOM management environment.

For details about different types of managed nodes, see "Types of Managed Nodes" below.

You can build a HPOM environment by arranging nodes to layout groups or node groups by using the `opcnode` and `opclaygrp` command-line tools. For more information, see the *opcnode (1m)* and *opclaygrp (1m)* manual pages. For details about managed node groups, see "Managed Node Arrangements" below.

Types of Managed Nodes

Your managed nodes can be complete systems or intelligent devices:

Controlled	All management and monitoring capabilities can be applied to controlled nodes.
Monitored	Management information is collected and forwarded to the management server, but no corrective actions or operations can be started. Monitored nodes are useful if you want to restrict access to the node for security purposes.
Message Allowed	No agent or subagent software is loaded, but messages are accepted by HPOM. For example, intelligent network devices like peripherals or nodes belonging to remote networks can be message-allowed nodes.
Disabled	No agent or subagent processes are started. Incoming messages from these nodes are ignored. This is a temporary state of a controlled, monitored, or message-allowed node.

Managed Node Arrangements

You use the following primary groups for organizing managed nodes:

- **HPOM Node Bank**

  Contains all nodes in the HPOM management environment.
- **HPOM Node Hierarchy Bank**

  Contains the node bank as the default HPOM node hierarchy.
- **HPOM Node Group Bank**

  Arranges nodes into logical responsibility groups.

Note: When you are configuring HPOM users, node groups are used to define the responsibilities of the operator, and node hierarchies are used to organize the operator's `Managed Nodes` group.

HPOM Node Bank

The HPOM node bank is the default **node hierarchy** for HPOM. This default hierarchy contains all nodes managed or monitored by HPOM. In addition, symbols representing a collection of HPOM external nodes can be part of the HPOM node bank. No HPOM software runs on these nodes, but events from these nodes are accepted.

For more information about node hierarchies, see "HPOM Node Hierarchy" below.

Initially, the HPOM node bank contains the management server. You add all other nodes, external nodes, and node layout groups.

You can add nodes by using the `opcnode` command-line tool. For details, see the *opcnode(1m)* manual page.

The HPOM node bank is a static map. You control all changes to it. You decide when to add nodes to the bank or delete them.

In an environment in which hundreds of nodes are being managed, the nodes and their names can become difficult to read. To avoid confusion, you can use **node layout groups** to organize the node bank into several hierarchical levels. For more information, see "Setting Up a Node Hierarchy" on the next page.

HPOM Node Hierarchy

The HPOM node bank is a default node hierarchy. A **node hierarchy** represents the hierarchical organization of nodes and layout groups. Each node hierarchy contains all managed nodes that are configured in the HPOM environment. These hierarchies differ only in how they organized these nodes.

Node hierarchies are assigned to the HPOM operators and display in their Managed Nodes window. However, although each node hierarchy contains every node configured in the HPOM environment, operators see only those managed nodes for which they are responsible.

You can use layout groups to group the nodes and arrange them hierarchically. Layout groups can contain nodes and other layout groups creating a node hierarchy. The `opclaygrp` command-line tool can be used to manage layout groups and node hierarchies.

You can use the `opcnode` command-line tool to add nodes to HPOM node bank. You can also add nodes to any other node hierarchy. Each node you add is also added to all other node hierarchies. These new nodes are added to the specified node and layout group, and by default, to the topmost level of all other hierarchies. Similarly, deleting a node in one node hierarchy also removes it from all other node hierarchies and thereby from the HPOM environment.

For more information, see the *opclaygrp(1m)* and *opcnode(1m)* manual pages.

Setting Up a Node Hierarchy

If you have a great amount of nodes in the node hierarchy bank, you can use layout groups to organize the nodes in logical entities.

A layout group is a hierarchy that allows you to see only the logically structured node hierarchy levels that interest you. The layout group principle can be compared to creating a UNIX file tree directory out of a collection of flat files or directories. Instead of a flat structure, you can build a hierarchy or a tree structure for your nodes by moving and nesting one layout group in another.

For detailed information about creating, modifying, and deleting layout groups and node hierarchies, see the *opclaygrp(1m)* manual page.

Applying Actions to All Nodes in a Hierarchy

After you have created a node hierarchy, HPOM provides the `opcnode(1m)` command-line tool, which allows applying actions to all the nodes contained in the parent group simultaneously. For example, to assign a policy to a series of nodes, specify the parent node group of the nodes you want to assign the policies to and the policy you want to assign:

opcnode -assign_pol pol_name=*<policy_name>* pol_type=*<policy_type>* version=*<policy_version>* group_name=*<nodegrp_name>*

HPOM automatically assigns the policy to the node group including all nodes contained in that group.

Actions You Can Apply to All Nodes

You can apply the following actions to a managed node, the `HPOM` node hierarchy, and the `HPOM` node bank:

- Modify a node group by using the `opcnode` command-line tool.
- Start up HPOM applications by using `Customized Startup`.
- View the messages of selected node symbols.
- Start and stop agent services.
- Distribute software, configurations, and so on.
- Assign policies.

Actions You Cannot Apply to All Nodes

You cannot apply any of the following actions to a managed node, the `HPOM` node hierarchy bank, or the `HPOM` node bank:

- Issue reports.
- Add or modify applications.
- Start up applications by using `Customized Startup`.

Conditions for Applying Actions to All Nodes

Consider the following conditions when applying actions to hierarchical groups:

- **Multiple Parent Groups**

  If a node occurs more than once under multiple parent groups, HPOM recognizes only one instance of this node and applies the action only once.
- **Groups**

  You can apply actions to several groups as well as a combination of nodes and node groups.
- **HPOM Applications**

  HPOM applications can contain HPOM objects other than nodes, such as LAN cards, devices, file systems, and so on. HPOM applications function the same as other applications, they receive the same HPOM file action list.

  For more information about HPOM applications and HPOM services, see "Adding Applications" on page 62.

Adding Nodes

Use the `opcnode` command-line tool to add nodes to the HPOM environment.

For more information, see the *opcnode (1m)* manual page.

Adding Internal Nodes

As a general rule, add all IP nodes by using the `opcnode` command-line tool.

Note: When you enter the hostname, HPOM tries to get as many characteristics about the node as possible. HPOM requests the MIB (through SNMP), the machine type, and then determines the corresponding IP address. For detailed information about adding nodes with multiple addresses, see the *HPOM Administrator's Reference*.

Characteristics of Internal Nodes

Nodes added by using the `opcnode` command-line tool have the following characteristics:

- **IP Nodes**

  Internal nodes are IP nodes.
- **HP Operations Agents**

  Internal nodes usually have HP Operations Agents running on them.
- **Individual Addition**

  Internal nodes are added to HPOM individually by their hostnames and specific IP addresses.
- **Unlimited Functionality**

  Internal nodes offer all HPOM functions, including:
  - Log file monitoring
  - HPOM message interception

Reasons Not to Install HP Operations Agents

You may choose not to install HP Operations Agents for any of the following reasons:

- Security concerns
- Agents not required
- Platform or operating system version not supported by HPOM

Adding External Nodes

You can use the `opcnode` command-line tool and a pattern-matching method for adding the external nodes. Use the `opcnode` command-line tool to install nodes with limited functionality if the following is true:

- Node has no IP address (SNA, DECnet).
- You group together a certain set of IP nodes for a specific hostname pattern or IP-address range.

When using the `opcnode` command-line tool, you need to set the machine type of a node to `MACH_BBC_OTHER_NON_IP` and the communication type to `COMM_UNSPEC_COMM`.

Example:

```
opcnode -add_node node_name=computer.company.com net_type=NETWORK_OTHER mach_
type=MACH_BBC_OTHER_NON_IP group_name=external ccomm_type=COMM_UNSPEC_COMM
```

As there are two ways of adding external nodes, it is possible that nodes may be represented more than once. For example, a node added with the `opcnode` command-line tool could easily be added again in like external node, only with a different pattern. Similarly, a node added by pattern matching could be added again by a similar pattern match. This is not a problem for HPOM. However, when a user performs a task and nodes are highlighted, more than one node may be highlighted as the source of a message.

For more information, see the *opcnode(1m)* manual page.

Conditions for Adding External Nodes

For nodes you want to add to the HPOM environment, one of the following conditions must be true:

- Known to the name server
- Configured in the `/etc/hosts` file

Note: External nodes reduce the performance of your management server system because of the necessary pattern-matching activity.

Characteristics of External Nodes

External nodes added by using the `opcnode` command have the following characteristics:

- **All Node Types**

  External nodes are of any type (for example, SNA, DEC, and IP).
- **No HP Operations Agents**

  External nodes do not have HP Operations Agents running on them.
- **Batch Addition**

  External nodes are added to HPOM in batches, by matching their names or addresses against a pattern.
- **Limited Functionality**

  External nodes offer the following functions:

  - *Trap Interception*

    Traps can be intercepted by the HP Operations management server (see the HPOM Administrator's Reference).

  - *Symbol Highlighting*

    Node symbols in the Java GUI Object Pane can be highlighted by HPOM to indicate the source of messages found in the `Message Browser`. Note that these nodes can also be set up to receive messages from a proxy.

  - *Message Filtering*

    Nodes symbols in the Java GUI Object Pane can be selected by users to filter the messages sent to the `View Browser`.

  - *Status Coloring*

    Change color to indicate message severity status, as set in HPOM status propagation.

- External nodes do not offer some functions available to internal nodes:
  - *No Message Policies*

    Messages from log files, monitors, `opcmsg`, and so on. That is, policies cannot be assigned to them.

  - *No HPOM Applications*

    Ability to run HPOM applications.

  - *No Broadcasts*

    Ability to run broadcasts.

  - *No Scheduled Actions*

    Ability to run scheduled actions.

Adding Nodes by Using the Administration UI

If you need to specify a complete set of node attributes when adding a node, use the Administration UI.

The Administration UI provides the following node attributes:

- *Automatic Update of System Resource Files*

  You can integrate the HP Operations Agent command `opcagt` (for example, `/etc/rc.config.d/opcagt` on HP-UX managed nodes).
- *Node advanced options*
  - Virtual terminal emulator
  - Physical terminal
  - Character format
  - Message stream interface output
  - Logging information
- *Communication Options*

  HTTP/SSL is the default communication type for new HPOM nodes.

  You can define the following:
  - Security parameters

- Installation method
- Buffer file size limitation

Setting up Security for Different Types of Managed Node

Use the `opcnode -list_node` command to list the types of managed nodes. To change a node type use the `opcnode -chg_nodetype` command.

- **Monitored Nodes**

  Secure nodes within the environment (that is, nodes on which the operator logon or the starting of actions is restricted) can be designated as **monitored-only** nodes. Operators can receive and review messages issued by the node, but all actions (that is, broadcast commands, logons, automatic and operator-initiated actions) are prohibited.
- **Controlled Nodes**

  Operators can start and stop actions and perform logons on **controlled** nodes. The administrator reviews the security aspects of each node individually, permitting operator access and defining whether actions and commands can be started.

If an operator tries to start an action simultaneously on both a controlled node and a monitored node, the action is sent only to the controlled node on which an action agent resides.

Managing Disabled Nodes

Some messages, such as those caused by planned outages, require no attention from an operator but may obscure other unrelated messages that need urgent attention. For outages that occur regularly, administrator can suppress such messages by configuring scheduled outages (see "Implementing Message Policies" on page 127). For a single planned outage, an administrator may instead prefer to isolate a managed node by temporary making it a **disabled** node. From such a node, all further messages are suppressed from being sent to the management server. Operators can continue working with messages from other managed nodes that remain in the managed environment.

A disabled node is still part of HPOM, but is excluded from the environment of all operators who have the node in their responsibility matrix. All of the node attributes are known to HPOM, and the node is still part of the node bank.

When conditions are suitable for a disabled node to be returned to the managed environment (for example, when planned outage work is completed), the administrator can re-enable the node. Messages available before the node was disabled are available again, and new messages are again accepted by the management server.

Configuring Node Groups

A node group is a logical group of systems or intelligent devices that the HPOM administrator sets up and delegates to an operator to manage. Because a single system can belong to multiple node groups, and may therefore be influenced by several operators, the use of node groups considerably reduces the effectiveness of the administrator's configuration efforts.

Node groups can also be used to simplify the configuration process. For example, you can assign the same policy group to all systems of a particular node group. If you later add a new node to the node group, the policies assigned to the node group are automatically assigned to the new node. Nodes within a single group usually share common characteristics.

For example, you might group all nodes with the following shared characteristics:

- Same location
- Similar function
- Similar type

The policies you apply to grouping can be freely selected to meet the requirements of your environment.

Note: When you configure operators, you assign node group responsibilities to operators. Group the nodes of your environment into logical responsibility categories that can be assigned to the different operators. A single node can belong to more than one group. You can check which nodes are assigned to a specific node group by using the `opcnode` command:

```
opcnode -list_ass_nodes group_name=<nodegrp_name>
```

Initially, HPOM has the default node groups `hp_ux` or `solaris`, and `net_devices`.

Determining the Status of Nodes

When no browser windows are open, all known nodes are colored green and all unknown nodes are colored blue. When a `Message Browser` is open, the color of the node reflects the status of the most severe unacknowledged message that concerns the node. For more information on how message ownership can affect status propagation, see the *HPOM Java GUI Operator's Guide*.

Assigning Categories to Node Groups

Categories can also be assigned to node groups. Categories define the link between the node group and the related instrumentation such as scripts and binaries that are needed by the HP Operations Agent to successfully monitor nodes within this node group.

For detailed information about category assignments, see the *HPOM Administrator's Reference*.

Managing Policies in HPOM

HPOM policies are managed in a way that allows the policies to be registered in the database, assigned to managed nodes, and distributed to them.

For the conceptual information about the policies, see "HPOM Policies" below.

For information about administration tasks related to the policies, such as adding policies, registering policies and policy types, and so on, see the *HPOM Administrator's Reference*.

Policies can have multiple versions on the HPOM 9.xx management server. For detailed information, see "Policy Versions" on page 51 and "Policy Groups" on page 59. For more information about managing multiple versions of HPOM configuration on managed nodes, see the *HPOM Administrator's Reference*.

HPOM Policies

A policy is a configuration element which consists of data and meta information. Policies are deployed to managed nodes. The data information part usually consists of a set of rules for generating the messages on the managed node to which the policy is deployed. While the data information part is completely defined by the user, the meta information part is used for administrative tasks and is managed by the HPOM product. Policies are used for configuring HP Operations Agents on managed nodes and for determining the conditions under which the messages are produced and sent to the management server, informing operators on events.

The policy consists of at least two files: a **policy header** and one or more **policy bodies**. The policy header is an XML file containing attributes such as name, type, version, and so on.

Note: Policy names should not contain the colon character (`:`) because it is used for identifying policies in the `opcpolicy` usage syntax. For usage details, see the *opcpolicy (1m)* manual page.

Policy bodies contain actual agent configuration. Policies are identified by their names, versions, and types, or by their UUIDs. A **policy type** is a policy attribute that determines the rules to which policy bodies must adhere. All policies of the same type are used by the same HP Operations Agent process on the managed node. In addition to the policy types predefined in HPOM, custom policy types can be created by the user.

A **policy container** represents a set of policies with the identical name and the type, but different versions. All policy versions in the policy container are unique. Some operations can be performed on policy containers, which means that they are performed on each policy in the container. Each container

has a unique ID that is shared by all the policies in that container. Containers are identified by their names and types, or by their IDs.

Policy Types

HPOM provides several different policy types. Each type of policy enables you to perform a different monitoring or configuration task. Therefore, a policy type is a set of configuration information that defines what a policy can manage. Every policy belongs to one policy type.

The following policy types enable you to perform monitoring or configuration tasks:

- ConfigFile Policy Type[ab]
- Event Correlation Policy Type
- Event Correlation Composer Policy Type[a]
- Flexible Management Policy Type[ac]
- Logfile Entry Policy Type
- Measurement Threshold Policy Type
- Node Info Policy Type[a]
- Open Message Interface Policy Type
- Remote Action Security Policy Type[ad]
- SNMP Interceptor Policy Type
- Scheduled Task Policy Type
- Service Auto-Discovery Policy Type[a]
- Service/Process Monitoring Policy Type
- SiteScope Policy Type[a]
- Subagent Policy Type[ae]
- Windows Event Log Policy Type
- Windows Management Interface Policy Type

[a]For this policy type, you cannot use the policy body grammar described in Appendix A.

[b]For detailed information about the ConfigFile policies, their syntax and keywords, see the HPOM Administration UI Help.For more information on how to enable running callbacks on an agent, see the *HPOM Administrator's Reference*.

[c]For this policy type syntax, use the *opcmom* manual page as a reference. See also the *HPOM Administrator Reference* for more information about the flexible management configuration.

[d]See the *HPOM Administrator's Reference* for details on remote action authorization.

[e]See "Subagent Policies" on page 60 for more information.

You can manage policy types and policies by using the Administration UI. All policy types are described in the *HPOM Administration UI Online Help*.

HPOM provides policy editors, which enable you to create new policies or modify existing policies to suit your requirements. There is a different policy editor for each policy type. For more information about the policy editor, see the *HPOM Administrator's Reference*.

For detailed information about editing default policy types, see "Policy Body Grammar" on page 253.

Multiple Policy Types

It is possible to have multiple types of policies on the management server mapped to the same policy type on the managed nodes. Therefore, during the policy type registration you can specify the type by which the policy is classified on the managed node.

If this type is not specified, the policy type name registered on the management server is used instead. In this case, if there are no corresponding consumers on the managed node, the policies will nevertheless be deployed, but ignored afterwards.

Table 1: Examples of Policy Types on the Management Server and on Managed Nodes

On the management server	On the managed node
`Logfile Entry` `Windows Event Log`	`le`
`Measurement Threshold` `Service/Process Monitoring`	`monitor`

Policy Versions

Having multiple versions of policies stored on the management server enhances the flexibility in operating with policies and policy groups, and allows simplified interoperability between HPOM on Unix, Linux, and Windows platforms. It also ensures the following:

- *SPI and customer config migration-related issues get simplified*

  Having multiple SPI versions on the server allows their deployment to different group of nodes, without the need to rename policies or policy groups.
- *Easier management of the configuration data*

  With policy versioning, it is possible to determine the versions of configuration elements (for example, policies) that are present on the management server and the managed nodes. For example, the simplified administration of different subagent versions is enabled, as well as their assignment and deployment to the managed nodes. For detailed information, see "Managing of Subagents in HPOM" on page 60 and the HPOM Administrator's Reference.

HPOM for different platforms (UNIX, Linux, and Windows) uses the aligned policy versioning functionality. This allows a single set of SPI policies to be delivered for HPOM on both platforms, as well as the simplified data exchange between them. For data exchange considerations, see "Policy Data Exchange Model" on page 54.

All policies have version numbers. A version number consists of two numbers (both with up to four digits) with the format `major.minor` (for example, 1.0).

The major number represents the number of the consecutive release, and it can also be reserved for the specified purposes (for example, for SPIs release tracking). The minor number is used for patching and customizations. The initial minor number for each release should be 0.

Note: Major numbers may be aligned with versions of the monitored applications. For example, they can consist of three digits which could be reserved for future releases of the application. This means that major version 450 could be reserved for application version 4.5, major version 460 could be reserved for application version 4.6, and so on.

The version of all newly created policies is set to 1.0, while the version of all default policies delivered with HPOM 9.xx is 9.0.

When the policy content is modified, the minor digit of the policy version number is automatically incremented, for example, from 1.0 to 1.1. If the new version already exists, the next available value is chosen, for example 1.2. To learn how to overcome the conflicts between the versions, see "Managing Conflicts between Policy Versions" on the next page.

Note: A change in the policy header (for example, changing the policy description) does not result in the creation of a new policy version.

The policy version can be replaced according to your preferences by using the `opcpolicy` command-line tool. The policy version numbers can also be changed without the need to modify the policy content.

This is especially useful when aligning the policy versions that are released together. For usage details, see the opcpolicy (1m) manual page.

Note: The new version number creation results in the creation of a new policy, even if the content is unchanged. The new policy has a new version UUID, but the container ID is same as before. On the other hand, changing the policy name results in a new object in the database with a new version UUID and a new container ID.

Only one version of each policy can be installed on a managed node. When deploying a new policy version to a managed node, the new policy version replaces the existing version, regardless of its version number. This is because both versions use the same container UUID that is used on the managed node to identify the policies.

Note: In production environments policies should be assigned to policy groups, which are then deployed to nodes and node groups. Policy versions should correspond to the versions of the policy groups to which they are assigned.

To learn how to handle multiple versions of HPOM configuration (which includes, in addition to the policies, policy groups and the instrumentation data) on managed nodes, see *HPOM Administrator's Reference*.

Managing Conflicts between Policy Versions

Policy versions come into conflict when the version number of a policy that you intend to assign already exists in the database. The following modes are available for managing conflicts between versions:

- *Overwrite mode*

  The conflicting version in the database is replaced.
- *New version mode*
  - In case of the conflict between the versions, the minor digit is automatically incremented for the version being uploaded. If the new version also exists, the first available value is chosen. The user is informed about the version number and the UUID.
  - If there are no conflicting versions, the policy is uploaded with the current version.
- *Error mode*

  In case of the conflict between the versions, an error is returned and the upload is canceled.

If one version of a policy is assigned directly to a managed node at the same time as another version of the same policy is assigned indirectly (for example, by assignment to a node group or policy group), it can sometimes be unclear which version of the assigned policy should actually be deployed to the managed node.

HPOM assumes a higher priority for direct assignments than for any assignment made indirectly, for example, through groups. In other words, in the event of a conflict between directly and indirectly assigned policy versions, direct assignment to a managed node is considered more important (and overwrites) any indirect assignment even if the version specified by the indirect assignment is more recent.

For example, if policy version 1.3 is assigned directly to managed node AA and version 1.6 of the same policy is assigned to a node group to which managed node AA belongs, then any policy deployment would deploy version 1.3 of the policy in preference to version 1.6, which although more recent is assigned indirectly through a node group.

Note: HPOM is not able to prioritize different versions of a policy if both versions are assigned

indirectly (for example, to two different node groups or policy groups). To prevent unexpected deployment results, try to avoid assigning different versions of the same policy to different node groups or policy groups.

For more information about automating policy assignments and managing version conflicts, see the *HPOM Administrator's Reference*.

Policy Data Exchange Model

HPOM for Unix and Linux and HPOM for Windows share the common data exchange model for policies and policy groups. Consider the following:

- The policy is identified on the management server in one of the following ways:
  - By UUID
  - By name, version, and type
- All policies with the identical name and the type have a common identifier named container UUID. The policies that share the same container UUID have a different version number and a corresponding individual identifier named policy ID.
- It is possible to compare policies by using checksums. The checksum fields, stored in a policy header, are considered during the configuration download and upload. There are two types of checksums:
  - Policy header checksum

    Used to detect prohibited policy modifications (license violations).
  - Policy data file checksum

    Used for faster content comparison.
- A policy type is an agent-known string (for example: `monitor`, `le`, `trapi`, and so on) that also includes the UUID and the syntax version.

Updating Policy Assignments

Policies can be assigned to nodes, node groups, and policy groups. The modification of the policy leads to a new policy version. This means that the existing assignments point to the older policy version, and not to the modified one. Therefore, the assignments must be updated. The update of the assignments can be done automatically. The assignment modes are the following:

- FIX

  This is the default mode. The policies are deployed after the modification, but the policy assignments do not change regardless of the creation of the new policy versions.

- LATEST

  The assignments of policies to policy groups, nodes, and node groups are automatically updated as soon as the new highest policy versions are generated. This is enabled by setting the LATEST flag, which is a property of the 'policy to policy group assignment' object.

  This mode is recommended only for testing and development environments.

- MINOR_TO_LATEST

  Policy assignments are automatically updated to the latest version. The major version number is kept unchanged. For example, if you set the MINOR_TO_LATEST flag to update from existing 1.0 version, the result would be the highest 1.x version.

  This mode is recommended for production environments.

You can specify the assignment mode by using the opcpolicy and opcnode command-line tools. For usage details, see the opcpolicy (1m) and opcnode manual pages.

Note: When using the LATEST and MINOR_TO_LATEST modes, consider that these modes automatically update existing policy assignments when new versions that comply with the above criteria are created. With the FIX mode, policy assignments are kept unchanged (unless they are changed manually).

For detailed information about policy assignment tasks in HPOM, see the *HPOM Administrator's Reference*.

Managing Policy-Assignment Conflicts

Conflicts can occur if there are differences between the version of a policy assigned directly to a managed node and other versions of the same policy that are assigned indirectly, for example, by assignment to one or more node or policy groups to which the original managed node belongs. In the event of a version conflict, HPOM assumes a higher priority for policies that are directly assigned.

Note: HPOM is not able to prioritize different versions of a policy if both versions are assigned indirectly (for example, to two different node groups or policy groups). To prevent unexpected deployment results, try to avoid assigning different versions of the same policy to different node or policy groups.

For more information about conflicts between policy versions in the context of policy assignment, refer to the *HPOM Administrator's Reference* guide.

Assigning Categories to Policies

Policies can also contain category assignments. Categories define the link between the policy and the related instrumentation such as scripts and binaries that are needed by the HP Operations agent to successfully monitor the resources referred to in the policy.

For detailed information about category assignments, see the *HPOM Administrator's Reference*.

Policy Type Callbacks

Policy type callbacks are executables that are run at predetermined moments in the lifecycle of a policy of the specific type.

There are four types of callbacks: Edit, Check, Deploy and Cleanup. You can find more information about them later in this section.

There are a number of variables that can be used to define callbacks. Table 2 lists these variables and their replacement values.

Table 2: Variables and their Replacement Values

Variable	Replacement Value
FILENAME	Full path of the temporary file into which policy body contents are dumped.
POLICY	Policy name.
VERSION	Policy version.
UUID	Policy ID.
SYNTAX	Policy syntax version.
TYPE	Name of the policy type.
AGENT_TYPE	Type of the policy as recognized by the HPOM agent.
MGMT_SV	Fully qualified domain name of the management server where the policy resides.
ENCODING	Content encoding of the policy body.
OPTION_X	Arguments passed to the editor by the edit callback. X represents the number of the passed arguments, so that the first argument is named OPTION_1, second argument is OPTION_2, and so on.
NODENAME	Fully qualified domain name of the managed node to which the policy is

Variables and their Replacement Values, continued

Variable	Replacement Value
	deployed. This variable can be used with Deploy or Cleanup callbacks.
NODE_TYPE	Operating system type of the managed node to which the policy is deployed. This variable can be used with Deploy or Cleanup callbacks.

Before the callback is executed, a variable replacement is performed. This allows runtime values to be passed to the callbacks as parameters. If no variables are present in the callback invocation string, value of $FILENAME is automatically appended to the string before the callback is executed.

To reduce the risk of registering callbacks that are executed with the administrator privileges by the unauthorized users, the following security levels for callback executables are available:

- **STRICT** (default)
  - *When running HPOM as a root user*

    If any of the following conditions is met, the callback is not registered, modified, or executed:
    - The owner of a file or a directory is not root.
    - Either a file or a directory has write-,read-, or execute-by-group bit set.
    - Either a file or a directory has write-,read-, or execute-by-others bit set.
  - *When running HPOM as a non-root user*

    If any of the following conditions is met, the callback is not registered, modified, or executed:
    - The owner of a group is not opcgrp.
    - Either a file or a directory has write-,read-, or execute-by-others bit set.
- **RELAXED**
  - *When running HPOM as a root user*

    If the following conditions are met, the callback is not registered, modified, or executed:
    - Either a file or a directory has write-by-group bit set.
    - Either a file or a directory has write-by-others bit set.

    If this level is enabled, anyone can view the directories and run the executables, but only root has permissions to write.
  - *When running HPOM as a non-root user*

    The callback is not registered, modified, or executed if either a file or a directory has write-by-others bit set.

- **NONE**

  No checks are done on the callbacks or the directories where their callbacks reside.

Note: The security level is configured by using an HPOM variable `OPC_POLICY_CALLBACK_SECURITY`.

The mode of each callback script or binary is checked by using `stat()` in the following cases:

- at registration, before writing to the database
- at modification, before writing to the database
- before execution

If any of the listed conditions is met, an error is reported either to the Message Browser (if detected before execution), or to the terminal in which modification or registration is attempted. The error is then audited and logged into to the HPOM database.

For all callbacks the following requirements apply:

- if unsuccessful, returns a value different from 0
- if successful, returns value 0

Edit Callback

Edit callback is an executable that is run before the editor is invoked. Full path of the temporary policy data file is passed to the callback as a command line argument.

Check Callback

Check callback is an executable that is run before the policy is uploaded to the database. Check callback is run if `opcpolicy_add()` API is called with the appropriate arguments or if `opcpolicy` is invoked with the `check=yes` command line option.

When check callback successfully finishes, the policy is uploaded to the database.

Deploy Callback

Deploy callback is an executable that is run on a temporary file that contains the body of a policy before its deployment. This executable is used for changing the policy contents depending on the information available at the deployment time (for example, node name or node type).

When the deploy callback makes any modification in the policy body, all changes must be preformed in the temporary file whose name is passed to the callback executable. If the processing requires copying of the file, the callback must rename the modified file back to the original file name, or all changes are lost.

If the execution of the deploy callback fails, the policy deployment is controlled by the value of the HPOM variable `OPC_DEPLOY_IF_CALLBACK_FAILS`. The default value of the variable is `TRUE`, which means that the policy is deployed in any event. Eventual warning message is logged into the `system.txt` file and in some cases also to the message browser.

Cleanup Callback

Cleanup callback is an executable that is run after a policy is deployed to the managed node.

Even if cleanup callback fails, the policy is still marked as successfully deployed to the managed node, while a message with the information about the problem is sent to the active Message Browser.

Using ConfigFile Policies to Run Callbacks on Agents After Deployment

The ConfigFile policy type allows you to run callbacks on an agent after the deployment of the ConfigFile policy to perform some post-processing, such as loading OV Composer fact stores. For more information on how to enable running callbacks on an agent, see the *HPOM Administrator's Reference*.

For more information on the ConfigFile policies and their syntax and keywords, see *HPOM Administration UI Help*.

Policy Groups

The policy group hierarchy is organized as a tree-like structure. Each policy group is referenced through a unique path within the tree and has a world-wide unique UUID. Arbitrary links between these groups are not possible.

Note: Policy group names should not contain the slash (/) or the backslash (\) characters.

There can be multiple versions of policy groups on managed nodes. For information about managing these versions of policy groups, see *HPOM Administrator's Reference*.

Default Policy Groups

Default policy groups are provided with the HP Operations management server. To get a list of policy groups, run the following command:

```
# /opt/OV/bin/OpC/utils/opcpolicy -list_groups
```

The following default policy groups are provided with the HP Operations management server:

- Correlation Composer
- Examples
- Examples/ECS
- Examples/Unix
- Examples/Windows
- Management Server
- SNMP
- SiteScope Integration/*<SiteScope Policy Group>*

Managing of Subagents in HPOM

Subagents are products that are not part of HPOM, but are partially manageable from the HP Operations management server. Some of the subagents are controlled by the OV Control Daemon.

Administering of subagents in HPOM includes the tasks outlined in the *HPOM Administrator's Reference*.

HPOM provides the mechanism for subagent deployment and undeployment, however actual subagent configuration details are provided in the manuals supplied with the subagent software packages.

Subagent Policies

Subagent policies are the policies that facilitate management of the subagent by means of a special policy type called `Subagent`. Policy bodies of `Subagent` type policies contain rules for the subagent installation and deinstallation on the managed node. These policies are provided by the subagent software supplier. For more information about their contents, see the manuals supplied with the subagent software packages.

Policies of the `Subagent` type are not distributed to the managed nodes, nor are they present in the node inventory, unlike other types of policies.

During the subagent software packages installation on the HP Operations management server, its subagent policy is registered with the server. The policy, as well as the policy group to which it belongs, is then loaded to the HPOM repository.

During the distribution of the agent configuration, the BBC Distribution Manager (`opcbbcdist`) checks the policy content and executes the subagent installation procedure provided in the subagent policy body. For more information about the installation process, see the *HPOM Administrator's Reference*.

Policies of the `Subagent` type comply with the rules for managing HPOM policies.

Upgrade Considerations

It is possible to have multiple versions of the same subagent installed on the HPOM 9.xx management server. However, only a single version of the subagent can be installed on the managed node.

By installing a new version of the subagent package on the management server, a new version of the subagent policy is delivered as well, which makes it easier to determine which version is installed on which managed node. Upgrading a subagent on the managed node is performed by assigning a new version of the subagent policy and deploying the subagent.

Note: Installing a new subagent version on the management server does not automatically mean that all managed nodes are upgraded with it. It is necessary to manually assign the desired subagent policy version to the nodes and trigger the installation.

For detailed information about the upgrade process, see the manuals supplied with the subagent software packages.

Organizing Message Groups

Message groups are a convenient way to categorize messages. Messages belonging to the same function or task can be collected into a single group. For example, the message group `Backup` can contain all messages that relate to the backing up and storing of data (for example, messages originating from a network backup program, pieces of hardware used in the backup or storage operation, and so on). Message groups are then assigned to operators, who see and manage only those groups assigned to them.

As an administrator, you can add, review, and delete message groups.

Adding Message Groups

Before you add a message group to your environment, make sure the group does not conflict with, or duplicate, one of the default message groups provided by HPOM. For a list of default message groups, see the *HPOM Administrator's Reference*. You can delete any of these default message groups, except the `OpC` and `Misc` message groups. You can also modify the description of existing groups, or add new groups.

Reviewing Message Groups

By reviewing message group status, operators get an overview of each function within the environment. You can configure a bank of message groups, then decide which groups you want to assign to operators.

Organizing Applications

An application can be a program, command, script, utility or service that the operator uses to maintain and control system and network services. For example, both a backup program and the process status command `ps` can be integrated as applications. You can integrate standard or custom applications, and applications that are already integrated into HPOM.

Applications and application groups integrated into HPOM can be managed by using the `opcappl` command-line tool. For detailed information on this tool, see the opcappl(1m) manual page. HPOM provides a selection of default applications and application groups. For details, see the *HPOM Administrator's Reference*.

Grouping Applications

You can group applications into application groups to create a hierarchical bank. For example, you can separate applications with multiple entry points so operators can access each entry point as a separate application. Other applications can be gathered into logical groups.

To build a hierarchical application bank, you can nest application groups into each other.

You can assign the applications or application groups to operators by using the `opcappl` command with the `-assign_app_to_grp` or `-assign_grp_to_grp` option.

Adding Applications

You can add applications to the application bank by using one of the following actions:

- **Add HPOM Applications**

  You can add HPOM applications by using the `opcappl` command with the `-add_app` option. You should specify the application name, application call, user name an password. You can also specify a list of target nodes, label and other parameters. For example:

  ```
  opcappl -add_app app_name=APP_X app_call=testCall user_name=John passwd=xyz
  ```

Note: Only applications of the type `Start on Target Node(s) selected by Operator` can be started from the message browser in the Java-based operator GUI.

- **Add Internal Applications**

  You can add internal applications of the type Broadcast by using the `opcappl` command with the `-add_app_inter` option. Define the general information of the application and the application type. You can also specify a user name different from that of the operator. You may want to change the user name (for example, if the user root is required to start the application).

  Note: The application names supplied in the `opcappl` command must be unique.

When creating an application (`Tool`) in the HPOM Tool Bank, make sure that you are familiar with how each presentation of the application affects the application itself. You can choose between the following three presentations:

- `Window (Output Only)`
- `Window (Input/Output)`
- `No Window`

For detailed information, see the *HPOM Administrator's Reference*.

Customizing the Application Startup

You can change the preconfigured start-up attributes for an application before it is started by using the Java GUI or the `opcappl` command with the `-chg_app` option.

The HPOM administrator defines the start-up attributes for an application. After that, operators can start the application directly from the Java GUI.

You can customize the following start-up attributes:

- Target nodes for an application
- Parameters for the application call
- User executing the application

To customize the start-up attributes for groups or individual applications, follow these steps:

1. In the Java GUI, right-click **Actions** and select **Start Customized**. The Start Tool - Customized Wizard displays.

2. In the Customized Wizard, select a tool (application) you want to customize, and then click **Next**.
3. Specify a list of the target nodes where you want to run a tool, and then click **Next**.
4. Specify additional information needed to run a tool:
   - *Additional Parameters*

     In the Additional Parameters field, you can specify nodes, if an application call accepts node names as an option. For example, for an application call with the option `-nodes`, you can use the option and argument `-nodes $OPC_NODES`. The variable `$OPC_NODES` expands to the list of nodes. For a list of available variables, see the *HPOM Administrator's Reference*.

   - *User Name and Password*

     The preconfigured user name and password for the target nodes are displayed. The password is displayed as a series of asterisks (*). You can change the user name and password, and execute the application as a different user.

Note: The HPOM administrator specifies a default user name and password for logging onto a node and starting the application. For example, `spooladm` is the default user name for HP OpenSpool.

For information on the `opcappl(1)` command-line tool, see the *opcappl(1)* manual page.

HPOM Licenses

Many HP Operations Manager product components, such as HP Operations management servers and HP Operations agents, require a license. If no license is installed for an HPOM component, its use might be locked.

Types of Licenses

- **Instant-On License**

  An Instant-On license is a temporary license that allows you to use the product without limitation for a period of 60 days for purposes of evaluation. It is installed and initialized during the installation of HP Operations Manager. The license expires after the 60-day evaluation period and new product license passwords must be installed to continue using the product.
- **Permanent Licenses**
  - Production licenses are licenses for normal product use. They are available for all product components.
  - Non-production licenses are licenses or license passwords for special purposes such as back-up systems.

License Verification

The HPOM license status is checked once a day. If the license status for any of the licensed objects is not OK or if the number of available licenses reaches a critical level, an internal HPOM message is generated and an e-mail is sent to the license administrator. The license status can be checked at any time using the license management tool.

Target Connector Count

The number of nodes from which messages are found in the database are counted once a day to determine how many Target Connector licenses are required. Nodes with an HP Operations Agent installed do not need a Target Connector license and are not counted.

History data values ranging between [today-31 days 00:00hrs] and [today 00:00hrs] are used to calculate the 30-day average, which is shown in the license report and is used for checking installed Target Connector licenses. If more than one value is stored for a particular day, the average value for that day will be used for the 30-day average.

You can use the `opcremsyschk -list` command to check which values have been calculated over the last 30 days.

License Notification

The HPOM licensing component has two possible license states - licensed and unlicensed - and three possible license violation notification (warning) levels - Warning, Major and Critical. License violation notification levels vary slightly, depending on the license type.

- **Instant-On License**

  Level 1 - Warning Notification: 6 to 14 days before the Instant-On license expires, a Warning Notification is sent to the HPOM Message Browser and an e-mail is sent to the license administrator.

  Level 2 - Major Notification: 0 to 5 days before the Instant-On license expires, a Major Notification is sent to the HPOM Message Browser and an e-mail is sent to the license administrator.

  Level 3 - Critical Notification: When the Instant-On license expires, a Critical Notification is sent to the HPOM Message Browser and an e-mail is sent to the license administrator. The license will be locked.

- **Permanent License**s

  If no licenses are installed and none are used, the license status will be regarded as Normal.

  Level 1 - Warning Notification: When the number of used licenses is greater than 90% and smaller than 100% of the number of installed licenses, a Warning Notification is sent to the HPOM Message Browser and an e-mail is sent to the license administrator.

  Level 2 - Major Notification: When the number of used licenses is greater than 100% and smaller than 110% of the number of installed licenses, a Major Notification is sent to the HPOM Message Browser and an e-mail is sent to the license administrator.

  Level 3 - Critical Notification: When the number of used licenses is greater than 110% of the number of installed licenses, a Critical Notification is sent to the HPOM Message Browser and an e-mail is sent to the license administrator. If the number of used licenses exceeds 110% of the number of installed licenses, the license will be locked.

Support for Small Environments

To better support small HPOM environments with a small number of managed nodes, the critical license threshold (Level 3) should set to 5 or greater.

For example, on an HP Operations server with 20 HP Operations Agent licenses, the critical notification level (Level 3) should effectively be set to 25 rather than 22. This avoids unnecessary

restrictions and provides the user with the flexibility to set-up additional systems before the license is locked.

License Availability

It might happen that no license or an insufficient number of licenses are available for one or more licensed product components.

- If no HP Operations management server license is available, server processes cannot be started. The missing license will be reported in the license report or the license status overview.
- If no HP Operations agent licenses are available, new nodes cannot be added to the HP Operations Manager data repository and configured. Nodes that have already been configured can be used, modified or removed from the data repository. The missing license will be reported in the license report or the license status overview.
- If there is an insufficient number of Target Connector licenses, the missing licenses will be reported but not enforced.

Setting Up Users and User Profiles

After you have set up all the aspects of the operations that you want to include in the HPOM environment, you can set up various users.

For example:

User Type	Task Overview
Operators	To observe and maintain the managed nodes and objects.

For more information about the different user responsibilities that can be configured with HPOM, see "HPOM Overview" on page 17.

You can add each user you set up directly to the database and configure the operators by using the `opccfguser` command-line tool.

For more information, see the *opccfguser(1m)* manual page.

Adding Users

You can use the `opccfguser` command-line tool to add a new user. You need to specify a user name and assign a password. For example, if you want to add a user “John,” run the following command:

```
opccfguser -add_user john -password secret -label John -real_name John Doe
```

For more information, see the *opccfguser(1m)* manual page.

Adding Operators

HPOM allows you to provide operators with extensive capabilities and very powerful tools. Operators can access systems throughout the environment, execute commands and scripts to be performed on all or selected systems, and perform critical corrective actions. They can also be responsible for the continued services offered throughout the computing environment. These responsibilities require that operators have an understanding of operator commands and network platforms, and that they can prioritize multiple tasks.

Operator Responsibilities

HPOM helps you to reduce the workload of operators while increasing their effectiveness. Operators should have the experience and skills which match the tools you want to assign. They should have knowledge of the systems they manage, know the responsible individuals within the environment, understand the applications, commands and scripts made available to them, and be able to apply troubleshooting procedures.

Configuring Operators

You need to define the following items for an operator's configuration:

Capabilities	Permissions to start and stop actions, to acknowledge and unacknowledge, to own and disown messages, and to modify message attributes.
Responsibilities	Responsibility for all events belonging to the assigned message groups on their assigned nodes.
Applications	Applications and tools available to the operators.
Profiles	Preconfigured user profiles that define the configuration of abstract HPOM users.
Node Hierarchy	Hierarchical layout of the operator's managed nodes.

The user name and password required to add an operator are not related to real UNIX/Linux users. For more information on HPOM users' file permissions and environment settings, see the *HPOM Administrator's Reference*.

Configuring Operator Profiles

One way to add a new operator to your HPOM environment is to configure abstract users, so-called user profiles, and assign these user profiles to the operator you are setting up. Or, copy and rename the

configuration settings of an existing user. If you choose this method to set up a new operator, note, that the browser settings saved by a particular operator are also copied along with all the other operator configuration data.

Default Operator Profile

HPOM provides a default operator profile, `opc_op`. The default operator profile may be used as a basis for creating new operator profiles that more accurately reflect the needs of a given organization or environment.

The `opc_op` operator controls system management functions. The `opc_op` operator works mostly in a system environment, and can access a limited selection of tools (for example, Processes, Disk Space, Print Status, and so on). By default, the `opc_op` operator is given all capabilities, and is responsible for all default message groups.

Note: The `opc_op` operator is not concerned with the management of network activities.

For more information on the default environments of the HPOM operators, see "Assigning Message and Node Groups" below.

Assigning Message and Node Groups

You can assign message groups and node groups to operators in a single step. When you assign a node group to an operator, the nodes in that group become the responsibility of that operator. You can select a node group and then assign the message groups for which the operator will be responsible. Alternatively, you can select a message group and then assign the node groups.

Use the `opccfguser` command-line tool to assign message and node groups to operators. For example, to assign responsibilities to user “john” for all node groups and all message groups, run the following command:

```
opccfguser -assign_respons_user -user john -node_group -all -msg_group -all
```

For more information, see the opccfguser (1m) manual page.

Guidelines for Assigning Groups to Operators

Use the following guidelines to assign groups to operators:

Job Function	You can assign the `Backup` message group, and set all node groups containing nodes that perform backup tasks.
Geographical Location	You can assign all node groups in a single building or facility, as well

	as the corresponding message groups.
Node Type	You can assign all IBM systems and the corresponding message groups.

Note: Complex nodes that provide multiple services to multiple users require more attention, and they often generate a lot of messages. Make sure you do not assign too many complex nodes to a single operator and, as a result, create a set of managed nodes that is too difficult for a single operator to manage.

Defining Different Layers of Operator Responsibility

If your environment includes multiple locations or a global network environment, you may want to define different layers of operator responsibility. For example, you can assign the node groups of each location to different operators, and assign the message groups for the connecting network to another operator. You can also set up two or more operators to manage the same group of managed nodes or message groups.

Assigning an Operator's Managed Node Hierarchy

In the operator's managed node hierarchy, the nodes accessible to the operator have already been determined when operator responsibilities were determined. You assign the hierarchy of the managed nodes for the operator by selecting the node hierarchy in the `Node Hierarchy Bank`, then clicking `[Get Map Selection]` in the Add/Modify User window. For more information about node hierarchies, see "HPOM Node Hierarchy" on page 41.

Assigning Tools to an Operator

The administrator makes sure that the operator has all the tools needed to do the job. The operator's job is to manage the assigned message groups of a collection of nodes. You decide which tools to include, so the operator can effectively manage those nodes. Commands, scripts, applications, the broadcast facility, and access to systems are all tools you can assign to the operator.

Each operator is responsible for a different set of managed nodes and services. The tasks that each operator performs can be different. Examine the nodes and message groups assigned to each operator.

Defining a Toolset for an Operator

To help define a toolset for an operator, consider the following questions about the node groups and message groups:

- Which services are provided?
- Which peripherals are connected?
- Which systems and intelligent devices are included?
- Which applications are operating?
- Does a node have a single, specific function?

Verifying the Operator Has the Proper Tools

To verify that you have provided the proper tools, you can review the following questions:

- **Access**

  Can the operator access the controlled systems?
- **Actions**

  Are there enough automatic and operator-initiated actions to respond to critical and warning situations that could arise on each managed object?
- **Permissions**

  Does the operator have permission to start all applications within the collection of managed nodes?
- **Scripts**

  Does the operator have access scripts or commands that can be used for troubleshooting or corrective actions?

Assigning Applications and Groups to Users

Use the `opccfguser` command-line tool to assign applications and application groups to an operator, to a list of operators, or to all operators. You can specify the list of applications in command-line string or refer to a file where you specified all applications you want to assign. For more information, see the opccfguser(1m) manual page.

For example, to assign application groups listed in the `system_groups.txt` file to the `opc_op` operator, run the following command:

```
opccfguser -assign_appgrp_user opc_op -appgrp -file system_groups.txt
```

Assigning User Profiles

After you have set up user profiles, you can quickly and easily set up operators in your environment. All you need to do is assign the selected profile to the operator you are configuring. For more information about setting up user profiles, see "Configuring User Profiles" on the next page.

To assign the user profiles use the `opccfguser` command-line tool. For example:

```
opccfguser -assign <profile name> -all
```

For more information, see the *opccfguser(1m)* manual page.

Tip: Generate a report about the new operator if you are interested in the sum of all node groups and message groups assigned to the operator, either directly or by way of user profiles. For details, see "Generating Reports" on page 87.

Configuring User Profiles

User profiles simplify user management in a complex environment. You can create a hierarchical set of abstract users with a default configuration, then assign this configuration to the real operators you are setting up.

Note: HPOM does not provide any default user profiles.

For example, a user profile for a database administrator would include application groups with the applications to configure and maintain a database. If you also set up a database node hierarchy containing all the managed nodes that have a database running on them, you can easily configure a new operator responsible for database servers in the HPOM environment. You only need to add the operator with the necessary capabilities, assign your database node hierarchy, and the user profile for database administrators. If the new operator requires any additional responsibilities or applications (that is, responsibilities or applications that are not already assigned by way of the user profile), assign them separately.

You can configure a user profile by using the Administration UI. You can list, assign, and deassign user profiles also by using the `opccfguser` command. For details, see the *opccfguser(1m)* manual page.

Non-root Operation

A non-root user is a user with the limited authority in comparison to the root user.

For the requirements that you must fulfill to operate as a non-root user, see "Requirements for Non-root Operation" on the next page.

Note: When the system that hosts the HP Operations management server is restarted when running under the non-root user, the HPOM processes are started under the `opc_op` user.

When HPOM is in the non-root operation mode, the `opcuiwww` process is always started under the `opc_op` user.

For the scope of the limitations of the non-root user, see "Limitations of Non-root User" on the next page.

Caution: When running HPOM as a non-root user, the HP Operations Agent is reconfigured in the non-root mixed mode. For the requirements and the limitations regarding non-root operation on the HP Operations Agent side, see the HP Operations Agent documentation.

Requirements for Non-root Operation

To be able to run HPOM as a non-root user, you must meet the following requirements:

- To be able to monitor a log file using a policy, you must have permission to read that log file.
- To be able to start a program using an automatic command, an operator-initiated command, a tool, or a scheduled task, you must have permission to start that program.
- To manage HPOM as a non-root user you must have the primary UNIX group set to `opcgrp` and the umask should be set to `002`.

Note: After you switch to the non-user operation, you can operate only as a non-root user. To start the processes as a root user, use the `opcsv` command. See the opcsv manual page for the usage details.

Limitations of Non-root User

The following limitations apply for the non-root operation:

- You cannot switch back to the root mode if you have configured HPOM to non-root operation.
- Non-root operation is not supported on RHEL 5.x.
- Some Smart Plug-ins may require additional configuration or user rights when they run under a non-root user. For more details, see the documentation supplied with the respective Smart Plug-ins.
- You can upgrade the HP Operations Agent running locally on the management server only as a root user.
- You can use the `opcdbinit` and `opcdbsetup` commands only as a root user. For usage details, see respective manual pages for these commands.
- You can use the `ecsmgr` utility only as a root user.
- The backup and restore scripts (`opcbackup_offline`, `opcrestore_offline`, `opcbackup_online`, and `opcrestore_online`) cannot be used with HPOM configured for running in the non-root operation mode.

Non-root User Configuration

You can configure HPOM to run under a non-root user during the HPOM installation/configuration or the upgrade process.

For more information, see the *HPOM Installation Guide for the Management Server*.

Non-root User Privileges and Capabilities

When running HPOM as a non-root user, you have limited authority in comparison to the user root.

However, some of functionality usually available only to the root user is enabled also for the non-root operation by assigning the particular privileges (HP-UX and Solaris) or capabilities (Linux) to the request sender process (`/opt/OV/bin/OpC/ovoareqsdr`) and to the `/opt/OV/bin/OpC/utils/opckilluiwww` binary.

These are the following:

- *HP-UX*
  - The `netrawaccess` privilege is assigned to the request sender. This enables the request sender process to send ICMP packets for the hearbeat polling.
  - The owner privilege is assigned to the `/opt/OV/bin/OpC/utils/opckilluiwww` binary.

  This enables the Java GUI to be stopped in case the HP Operations management server processes are stopped.
- *Linux*
  - The `cap_net_raw` capability is assigned to the request sender. This enables the request sender process to send ICMP packets for the hearbeat polling.
  - The `cap_kill` capability is assigned to the `/opt/OV/bin/OpC/utils/opckilluiwww` binary.

    This enables the Java GUI to be stopped in case the HP Operations management server processes are stopped.

    Caution: These capabilities cannot be used if the HPOM binaries are on an NFS-mounted file system. To be able to use non-root operation in such a case, run the following command:

    `ovconfchg -ovrg server -ns opc -set OPC_NON_ROOT_USE_SUID TRUE`

    Make sure to run this command after the HPOM installation, but before the HPOM configuration (that is, before the non-root configuration is performed). By doing so, the

`setuid` bit is set to the `ovoareqsdr` and `opckiilluiwww` binaries. The owner of the binary is root, so this process is started as user root.

- *Solaris*

  The `setuid` bit is set to the `ovoareqsdr` and `opckiilluiwww` binaries. The owner of the binary is root, so this process is started as user root.

  However, because the request sender is a privilege-aware application, it restricts its privileges to the least required ones.

  These are the following:

  - Effective privileges: `basic`, `net_icmpaccess`
  - Inheritable privileges: `basic`, `proc_owner`

Updating the HPOM Configuration

This section describes changes you make to the HPOM configuration after installation. For details about installing HPOM software, see the *HPOM Installation Guide for the Management Server*.

Distributing the Configuration

You always perform the initial software installation and configuration on the management server. The initial default configuration includes the management server as the only managed node. Each time you change the configuration (for example, adding nodes, monitor programs, or policies), you make the changes on the management server, then distribute these changes to the managed nodes.

Distributing Parts of the Configuration

HPOM lets you determine the parts of the configuration you want to distribute to the managed nodes and which HPOM nodes are permitted to receive configuration data. For example, when you add a new node, or want to update a configuration on a managed node, you distribute software from the management server.

This software includes:

- **Agent Software**

  Contains all of the HPOM software for the managed node, for example, the action agent, message agent, and monitor agent. Only one distribution is required, but each time you add a new managed node to the configuration, the agent software must be distributed to the new node. You must also select this software component when installing new versions of the agent software.

- **Node Configuration**

  Includes the following:

  - *Policies*

    Configured message and monitor sources, as well as MoM configuration policies. You distribute policies to the managed nodes on which they operate.

  - *Actions*

    Scripts, programs, or applications started when an automatic, action, an operator-initiated action, or a scheduled action is started. You distribute an action to each managed node from which the action will be started.

  - *Monitors*

    Scripts and programs used by the monitor agent to check monitored objects. These scripts and programs are located on the nodes from which they are started.

  - *Commands*

    Scripts, programs, or applications started with the Broadcast Command window, or other applications started from the Java GUI. You distribute these scripts, programs, or applications to the managed nodes from which they are started.

Preparing for Software Distribution

To prepare for the distribution, HPOM downloads a policy from the HPOM database to a local file. Since HPOM attempts to download as rarely as possible, HPOM saves the policy file on the management server after distribution. The file is saved so that it can be distributed to other managed nodes later.

Note: The file is downloaded before distribution only if the policy is changed in the database or if the local file does not exist.

To reduce network load and increase performance, HPOM updates only those parts of the configuration that have changed.

Distributing the Configuration to Managed Nodes

To install or update the configuration from a management server to managed nodes, follow these steps:

1. Define the configuration to be installed or updated.
2. Assign the policies to the nodes.
3. Define how the configuration will be installed or updated.

Use the `opcragt -distrib` option to install or update your configuration. Specify the parts of the configuration you want to distribute. If you want to replace the entire configuration, you can use the `-force` option (for example, **`opcragt -distrib -force`**).

Note: You can also use the command `inst.sh` to manually install the agent software on the managed nodes. And you can use the command `opcragt` to manually distribute policies, actions, monitors, or commands to the managed nodes. Both commands can be run in a non-interactive mode, which allows you to schedule the installation or update to take place at any time (for example overnight or over the weekend). For details, see the *inst.sh(1m)* and *opcragt(1m)* manual pages. For information about manually installing the HP Operations Agent software, see the HP Operations Agent documentation.

Forcing Updates

When you do a standard install or update without selecting `force` option, only the new information in a configuration is transferred. Any information that has not been changed is not transferred by default. This reduces the load on the network and decreases transfer time. The `force` option forces HPOM to install or update all specified configurations to the selected managed nodes.

Caution: Avoid the `-force` option. It defeats the performance improvements of HPOM.

Distributing Policies to Managed Nodes

You distribute policies only to the managed nodes that need them. And you keep the policy definitions on the management server from which you make changes. Then you can redistribute the definitions as needed. Any time you make changes to the attributes of a message policy, such as the message group or severity level, you need to install or update the policies.

For example, assume you have written a monitor script to check the number of processes on managed nodes. After assigning the policy to the managed nodes, you install or update the policy from the management server to the managed nodes from which you want the monitor to operate. For more information on policy body syntax, see "Policy Body Grammar" on page 253.

Note: In this example, you might also need to install or update the monitor script to the managed node. For details, see "Messages from Threshold Monitors" on page 188).

After you have completely defined a new message source policy, included the policy in a policy group, and assigned the policy or policy group to the managed nodes, you distribute it to the managed nodes.

Policies Installed by HPOM

HPOM installs the following policies on a node:

- All policies assigned to the node
- All policies in a policy group assigned to the node
- All policies assigned to a node group containing the node
- All policies in a policy group assigned to a node group containing the node

Note: If you delete or modify a policy or policy group, or remove a policy assignment for a specific node, you must redistribute the new configuration for the affected managed nodes to start using the changes.

Policies that are currently being modified are locked, and are not distributed to the managed nodes. The HPOM administrator can generate a distribution report, called the `Node Config Report`, to verify the assignment of policies to managed nodes, and to check which policies must be distributed. For more information, see "Generating Reports" on page 87.

Use the following command to generate a distribution report:

```
/opt/OV/bin/OpC/call_sqlplus.sh node_conf <node name>
```

Avoiding Duplicate Policy-Node Combinations Automatically

Before distribution begins, HPOM verifies that each policy is installed or updated only once to the managed nodes, and that no invalid policy-node combinations occur. Policies can be contained in more than one policy group, and can be assigned to more than one node. As a result, policies can be assigned twice to the same node. Also, some policy-to-node assignments might be invalid (for example, a platform-specific policy assigned to a node running on another platform).

If a policy is assigned more than once to a managed node, HPOM ignores the duplicate assignment and distributes the policy only once.

Distribution Tips

This section contains tips for distributing HPOM software, configurations, and policies quickly and easily.

Updating Only Those Nodes Requiring an Update

One of the easiest ways to smooth HPOM distributions is to update only those nodes requiring an update.

To update the nodes, use the `opcragt` command-line tool. By using this tool with the `-distrib` option, you can apply updates to a single node, node groups, or all nodes. Example how to apply updates to node group:

`opcragt -distrib -nodegrp <group>`

Where *<group>* is a name of the node group.

For more information, see the *opcragt(1m)* manual page.

Note: If you are distributing to a UNIX cluster, you must distribute monitors, actions, and commands to each cluster client.

Reducing Nodes and Priorities During Distribution

The performance of some HPOM services, such as the `Message Browser`, can be slowed down if you distribute new configuration data to many nodes at the same time.

To avoid this problem, do the following:

- **Minimize Nodes**

  Minimize the number of managed nodes receiving new configuration data at one time by using the `OPC_MAX_DIST_REQS` configuration variable.
- **Reduce Priority**

  Reduce the priority of the `opcbbcdist` process on the management server by using the `nice(1)` command.
- **Use Category-based Distribution Method or Selective Distribution Feature**

  Prevent distribution of the particular configuration files that are not needed on a specific node by choosing the category-based distribution method or the Selective Distribution feature of `opcbbcdist`.

  For more information on the category-based distribution method or the Selective Distribution feature, see the *HPOM Administrator's Reference*.

Manually Distributing Policies

In some cases, it may be desirable to distribute policies to managed nodes manually. For example, if you want to schedule the distribution to take place over night, or if your security standards do not allow

you to transfer the policies through the network. HPOM supports the following types of manual distribution:

- **Over the Network**

  Use the `opcragt` command-line tool with the `-distrib` option to distribute policies, action, commands, and monitors to managed nodes. For more information, see the opcragt(1m) manual page.

- **Alternative Transport Medium**

  If you do not want to distribute the configuration over the network, use the `opctmpldwn` command-line tool to download (and encrypt) the policies. You can then decide how to transfer the downloaded (and encrypted) configuration to the managed nodes. For more information, see the opctmpldwn(1m) manual page.

Distributing Without Operator Configurations

When you add a new operator, you define a set of managed nodes and message groups for which the operator is responsible. The operator's configuration is stored on the management server, and it is not distributed to the managed nodes. If the new operator has special logon, customized application start, or broadcast command capabilities, this information is verified by the managed node each time the operator uses those capabilities. You do not distribute this information with the software and configuration.

Synchronization of the Configuration Changes

The data-synchronization feature of HPOM provides an automatic update of configuration data within the HP Operations server components, that is, the GUI, HP Operations management server processes, APIs, and so on.

Update of configuration can involve changes, such as adding, deleting, assigning, deassigning, or regrouping the following configuration objects: node, node group, applications, application group, policy, policy group, message group, and user profile.

Configuration change can be a simple operation with one configuration object, for example, updating a policy. Or it can be a complex operation where a configuration object of one type is assigned or deassigned to configuration object of another type, for example, assigning a node to a node group. HPOM provides an online configuration update, which means that the configuration changes are applied without restart of server processes and GUI.

HPOM configuration is stored in the database, configuration files, and configuration variables. Each time the configuration is changed with the `opccfgupld` tool, the database changes are uploaded. Each time the `ovconfchg` tool is used, the content of most configuration variables used in server processes

is updated. For more information on the configuration variables, see the *HPOM Administrator's Reference*.

The following HPOM configuration files are reloaded when the `ovconfchg` tool is used:

- Outage policy
- Message forward policy
- MSI conf. file
- Internal name resolution conf. file

Configuration Stream Interface (CSI) is an extension of the Message Stream Interface (MSI) for synchronizing the configuration changes. CSI provides registration for the configuration changes to the internal (server processes, Java GUI) and external (API clients) configuration consumers. Java GUI and server processes and are registered for the configuration changes by default. You can register for configuration change notifications by using the `opcif_*` APIs.

Synchronizing the GUI After Reconfiguration

When the update of the HPOM configuration takes place, this leads to the relevant changes of the Java GUI configuration objects, such as nodes, node groups, applications, and message groups.

The HPOM provides the synchronization mechanism between the server and Java GUI during the configuration update. Most configuration changes are automatically reflected in the Java GUI and there is no need for a restart. You can see the relevant configuration changes in the GUI immediately.

If, for example, the node hierarchy is modified, or an application is added to the application group, the synchronization mechanism allows you to see these changes without restarting the GUI, logging off, and logging back on again.

However, in case of massive configuration changes (for example, uploading a configuration with `opccfgupld`, or assigning and deassigning profiles to users), you need to reload your new configuration from the HP Operations management server. The reload is also needed for the update of filters definition or change of services with the disabled `Automatic Service Reload` option. You do not need to log out of the Java GUI.

Note: Synchronization events are forwarded to external users (for example, users of configuration APIs), if they are connected to the Configuration Stream Interface (CSI).

When the configuration update tool `opccfgupld` is running, new logons to the Java GUI are blocked and the following error message is displayed: `Cannot login while opccfgupld is running`.

Accessing Up-to-Date Content Automatically

Just before opening for the first time in an HPOM administrator GUI session, a new window reads the most recent data directly from the HPOM database. In this way, both you and your applications are able to access the most up-to-date contents of single, modified objects.

Restarting Sessions Manually

Selecting the `Restart Session` option closes the current sessions as normal. But, unlike the `Close` and `Exit` menu options, the `Restart Session` option does not ask you for any confirmation. The `Restart Session` option closes all open windows, then starts a new HPOM session using the settings stored in the most recent `Save Settings/Home Session`, including the contents and configuration of node, message, and application groups. However, windows, such as the Application Output window and the Report Output window, do not reopen. And windows sometimes reopen in different positions and with different geometry settings to those in the session that has been closed (unless, of course, you saved settings and details during the session that is being closed). This system behavior is identical to what you experience when you log out and log back in again.

Locking Components During Transactions Automatically

Since multiple processes are able to manipulate HPOM configuration data, there is an obvious need to control the configuration data as well as any attempt to modify it. In HPOM, applications lock data while they are modifying. This data locking prevents concurrent modifications by other applications or users. After modification, both the server processes and the user interfaces are synchronized automatically, so that you can see and subsequently make use of the updated configuration data.

HPOM uses a **transaction** concept with locking to synchronize components after changes have been made to configuration data. The transaction concept supports API functions to start, commit, and roll back user transactions.

Backing Up and Restoring Data

This section describes how to back up and restore data on the HP Operations management server.

Backing Up Data

HPOM provides two scripts to back up data on the management server:

- **opcbackup_offline**

  Manual offline backup. If you do not have the resources or need for an automated backup, you can use the `opcbackup_offline` and `opcrestore_offline` scripts to perform a full offline backup.
- **opcbackup_online**

  Automated online backup. If you use `opcbackup_online` to carry out an automated backup, any task that cannot be completed before the backup starts remains idle until the backup is complete and then resumes.

Comparison of Backup Methods

When planning and executing a backup, it is important to remember that the HPOM configuration encompasses the managed nodes as well as the management server. Consequently, if the restored configuration on the management server does not match the current configuration on a managed node, errors relating to missing instructions or incorrectly assigned policies may occur.

Table 3 gives an idea of the advantages and disadvantages of the manual (offline) and automatic backup methods.

Table 3: Comparison of Backup Methods

Backup Method	Backup Type	Advantages	Disadvantages

`opcbackup_offline`	Offline	• Archive-log mode does not need to be enabled:
		  • Better overall system performance
		  • Less disk space needed
		• Backs up binaries too (if full mode is used).
		• Performs a complete offline backup.
			• All Java GUI instances must be closed.
			• Stops all HP Software services, including the HP Operations server processes.
			• Can only recover to the state of the most recent full backup.

`opcbackup_online`	Automated	• HPOM Java-based operator GUI, trouble ticket, and notification services are fully operational during the backup. • Partial recovery of the	• Backup script pauses some HP Software services. • Archive log or WAL must be enabled: • Lower overall system performance

Comparison of Backup Methods, continued

Backup Method	Backup Type	Advantages	Disadvantages
		database is possible, for example: • Up to a given time • Individual tables that have been damaged • All HP Software processes run during the backup.	• More disk space needed • Does not back up binaries or temporary files (for example, queues).

Restoring Data

Restoring an HPOM Database

Data backed up with one tool must be recovered with the corresponding restore tool. For example, use `opcrestore_offline` to restore data backed up with `opcbackup_offline`. Similarly, use `opcrestore_online` to recover data backed up with `opcbackup_online`.

The `opcrestore_online` script enables you to restore the complete Oracle or PostgreSQL database to the state of the backup.

If you use the Oracle database, you can also restore the database to the most recent state (a roll forward is done based on the offline redo logs).

In addition, the Oracle archive log mode offers more possibilities. For example, you can use the Oracle archive log mode to do the following:

- **Retrieve Single Files**

  You can retrieve individual corrupt Oracle data files from the backup and recover them by using offline redo logs.
- **Recover Up to a Specified Time**

  You can recover Oracle data up to a specified point in time by using offline redo logs.

Note: Archive log mode is a term used by Oracle to denote a state in which data is periodically and automatically saved. Any changes to data files are stored in redo log files. These redo log files are subsequently archived.

For details, see the Oracle documentation.

Message Ownership

This section describes the effect ownership has on operations performed to solve problems.

The administrator determines ownership policy by selecting one of the ownership modes allowed in HPOM. For more information about configuring the ownership mode, see the *HPOM Administrator's Reference*.

Marking and Owning a Message

The concept of marking and owning a message is fundamental to your understanding of your system environment and the responsibility you have for its maintenance.

In HPOM, marking and owning are defined as follows:

- **Marking**

  Indicates that a user has taken note of a message. Informational ownership mode.
- **Owning**

  Indicates that, depending on how the environment has been configured, a user either wishes (optional ownership mode) or has been forced (enforced ownership mode) to take charge of a message to carry out actions associated with that message.

Ownership Display Modes

HPOM provides the following ownership display modes:

- **No Status Propagation** (default)

  The moment a message is owned or marked, the color indicating the severity of the message changes, a flag displays in the own-state column in the Java GUI `Message Browser`, and the own-state color bar in the `Message Browser` reflects the new number of messages owned. The status of a message that is owned or marked is ignored for the purposes of status propagation in the `Object Pane`; in the operator Message Group and Node Group; and in the administrator's message group bank.
- **Status Propagation**

  The status of all messages, whether owned or not, is used to determine the status of the related symbols of other submap windows. Consequently, in the example above, the managed node to which the single critical message relates will continue to assume the critical, severity-level color (by default, red) even after the message has been owned. In this display mode, the only indication that the message is owned is a flag in the own-state column in the `Message Browser`.

For example, if a user takes ownership of the one and only message with critical severity level relating to a given managed node, the managed node to which that critical message relates will no longer assume the critical, severity-level color (by default, red). Instead, it will reflect the status of the next unowned message with the highest severity level in the `Message Browser` relating to the same managed node.

For further information and instructions on how to change from one ownership display mode to another, see the *HPOM Administrator's Reference*.

Ownership Modes

HPOM provides the following default message ownership modes:

- **Optional**

  User has explicit permission to take ownership of a message. The owner of a message has exclusive read-write access to the message. With the exception of the HPOM administrator, all users who find this message in their `Message Browser` have only limited access to it.

  In this mode, only the owner of a message may do the following:

  - Start or stop operator-initiated actions related to the message.
  - Stop or restart automatic or operator-initiated actions related to the message.
  - Acknowledge the message.
  - Unacknowledge the message.

- **Enforced** (default mode)

  Operator either chooses explicitly to take ownership of an unowned message or automatically owns the message when performing operations on the message.

  In this mode, an operator automatically owns a message when attempting to do the following:

  - Start or stop operator-initiated actions relating to the message.
  - Stop or restart automatic or operator-initiated actions related to the message.
  - Unacknowledge the message.

- **Informational**

  Concept of ownership is replaced with that of marking and unmarking. A **marked** message indicates that an operator has taken note of the message. Marking a message is purely for informational purposes. It does not restrict or alter operations on the message in the way optional or enforced mode do. Operators may only unmark messages they themselves have marked. The administrator can unmark any marked message.

Generating Reports

HPOM meets the general need for more complex and comprehensive reporting with a selection of powerful report-writing tools and a wide range of informative reports on areas that range from low-level network elements to service availability. The reports are automated and viewable in different formats. Generally speaking, the range of reports and the configuration possibilities vary according to the type of user. For example, the reporting possibilities available to the HPOM administrator are greater than those available to other HPOM users.

Reporting Tools

The following tools are available to help write and generate reports:

- **SQL-based Reports**

  Internal predefined HPOM reports, based on SQL.
- **HPOM-specific Reports**

  HP Reporter />, with HPOM-specific reports.
- **Database Access**

  Direct database access with self-written scripts.

In addition, the *HPOM Reporting and Database Schema* provides essential information the HPOM administrator needs to define and create reports by using any external report-writing tool to access the HPOM database.

HPOM Reports

Although HPOM reports can be divided into administrator and operator reports, both administrators and operators can generate a number of different types of internal reports in the HPOM environment.

Administrator and Operator Reports

In general terms, there are two types of HPOM reports:

- **Administrator Reports**

  Report on all aspects of the HPOM working environment. The HPOM administrator could, for example, generate a report on all the actions taken by either all or selected operators to investigate the success rate of these actions. Alternatively, reports are available showing, amongst other things, node or node-group configuration.
- **Operator Reports**

Contains all operator instructions and message annotations. If you choose to generate reports on all active messages or all history messages, and the message buffer contains a large number of messages, generation of the report output can take several minutes.

You can select the type of report you want to generate and a report output media. The selection of reports for different users differs according to the scope and responsibility of the user.

Message, Error, and Configuration Reports

Specifically, you can select from the following report types:

- **Message Reports**

  You can prepare reports for a single message, or all messages displayed in the Browser windows. For more detailed information, see the HPOM Administrator's Reference.

- **HPOM Error Report**

  Contains the HP Operations management server error messages.

  These messages are written to the following files:

  `/var/opt/OV/log/System.txt` (Plain text)

  `/var/opt/OV/log/System.bin` (Binary)

  Note: The HTTPS agent and the management server use the same location.

- **Configuration Reports** (administrators only)

  Contains configuration information, for example on nodes, node groups, policies, operators, actions, and so on.

For a detailed list of available reports, and a brief description of their scope, see the *HPOM Administrator's Reference*.

Service Reports

You can get an overview of the status of services in the HPOM environment at any moment or over a specified period of time by using the HP Reporter. The HP Reporter provides preconfigured service reports for the HPOM managed environment. These reports include overviews of message throughput, problem response times, trends, and configuration issues.

For example, you can use the HPOM service reports bundled with the HP Reporter to provide information in both graphical and statistical form on the following:

- General state of the HPOM managed environment
- HPOM operators and their work load
- Status of automatic actions and operator-initiated actions

- Configuration problems in the HPOM managed environment
- Trend analysis for a wide variety of HPOM operational areas

For more detailed information on the contents of these reports and a description of their scope, see the *HPOM Administrator's Reference* and the HP Reporter documentation.

You can schedule service reports at specified times, and display the retrieved information in the form of graphs, diagrams, or statistical tables. In addition, if a web server is running and so configured, reports may be updated and published automatically on the web. For information on the HP Reporter, see the product-specific documentation. For more information on installing and configuring a web server for HPOM, see the *HPOM Installation Guide for the Management Server*.

Generating Reports

Administrators and operators can generate a number of different types of internal reports in the HPOM environment. In so doing, they can choose between a concise or detailed format for the reports. The resulting output may be sent to a printer or a file, or displayed online.

Creating PGM and INT Reports

All reports have the type PGM (program). PGM reports can be created by using a program or script or the type INT (internal). INT reports are produced by the internal functionality of the server, and are restricted to a set of selected messages. The PGM report type is used for the database reports that are generated by an SQL script. The PGM report type is also used for other programs and scripts.

Format of Internal Reports

For operators, the short form of the internal report contains the following information:

- Report date, time, and type
- All message attributes, including message text
- Original message text

Defining Your Own Reports

The HPOM reporting facilities enable you to design and generate your own reports with information retrieved directly from the database by using either HPOM applications, third-party tools, or even scripts that you write yourself.

HPOM also allows the administrator to define additional reports. For more information, see the *HPOM Administrator's Reference* and the *HPOM Reporting and Database Schema*.

The configuration of a new report is done by either creating a new script or program, or by creating a new SQL file. Then you edit existing plain text files to integrate the new reports. These configuration files define which reports are for the administrator and which are for the operator. You can configure an HPOM scheduled action policy to schedule these reports at convenient times and then display the retrieved information by using tools provided with HPOM.

For detailed information on how the HPOM database works as well as instructions on how to access the database and use the contents, see the *HPOM Reporting and Database Schema*.

Integrating Your Own Reports

You can integrate your own reports. For more information about creating and integrating your own reports, see the *HPOM Administrator's Reference* and the *HPOM Reporting and Database Schema*.

Chapter 3: HPOM Managed Nodes Concepts

In this Chapter

This chapter describes HPOM managed nodes concepts. The information in this chapter covers the following topics:

- "HTTPS Agent Overview" below
- "HTTPS Communication in HPOM" on page 94
- "Security Concepts" on page 99
- "Managing HTTPS Nodes" on page 106
- "Creating and Distributing Certificates " on page 111
- "Virtual Nodes in HPOM" on page 111
- "Proxies in HPOM" on page 114
- "Tracing HPOM" on page 116
- "Firewalls and HTTPS Communication" on page 123
- "Configuring HTTPS-Based Communication" on page 125

For more detailed information about HPOM managed nodes, see the *HPOM Administrator's Reference* and the HP Operations Agent documentation.

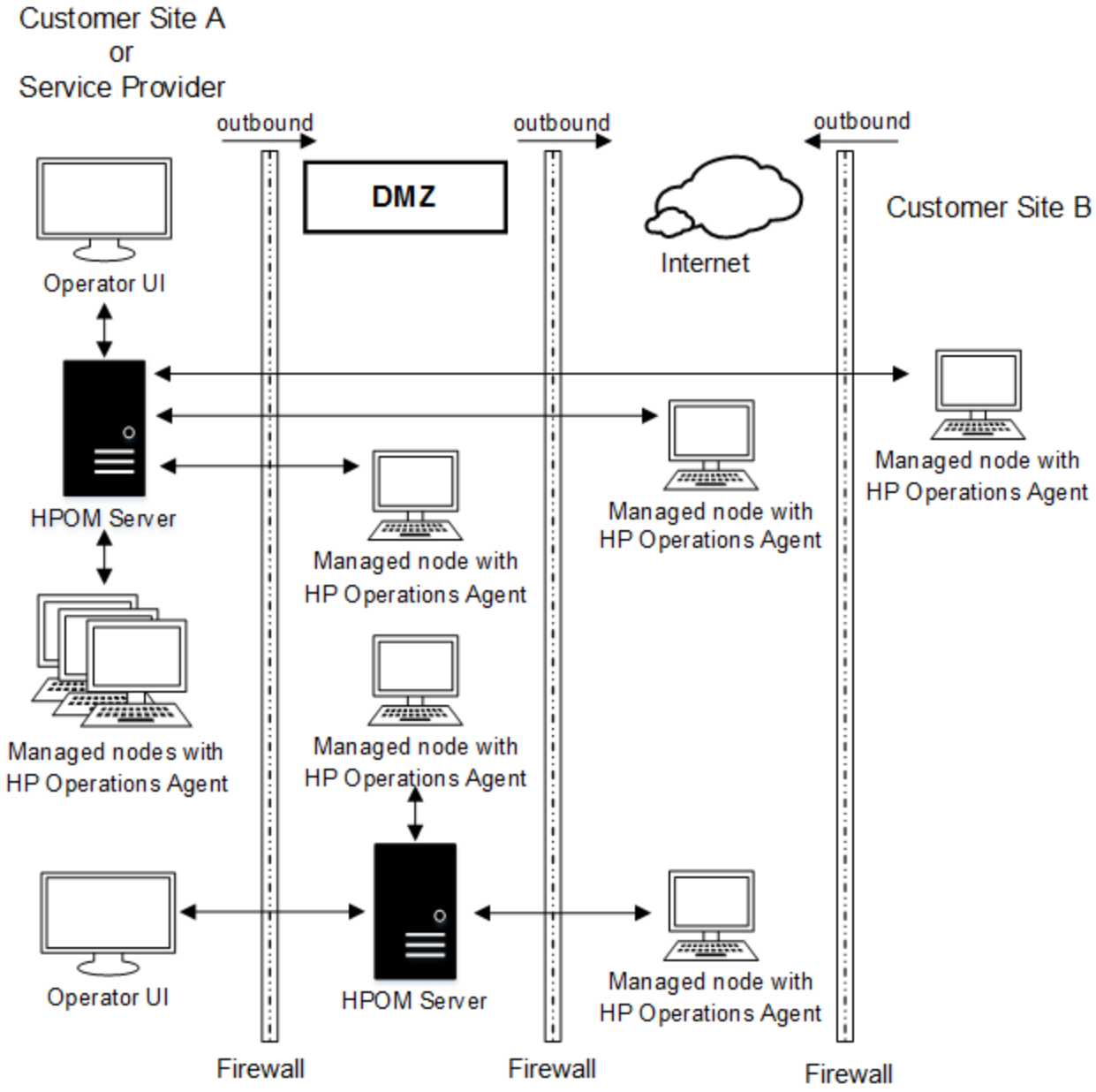
HTTPS Agent Overview

HTTPS agent software provides highly secure communication between HP Operations management servers and their managed nodes.

Figure 7 illustrates a typical environment managed by HP Operations Manager.

Advantages and benefits of using the HTTPS agents are described in the following chapters.

Figure 7: A Typical HPOM Managed Environment



HTTPS-based communication provides you with the following major advantages:

- Simple management through firewalls with configurable, single-port, secure communication using, open, HTTPS-based communication techniques. Restrict outside access to dedicated HTTP proxies and reduce port usage by multiplexing over HTTP proxies.
- Out-of-the-box Internet Secure Communication using SSL/PKI encryption with server and client certificates for authentication.
- Communication is based on standard Web technologies (HTTP, SOAP, Proxies, SSL, and others),

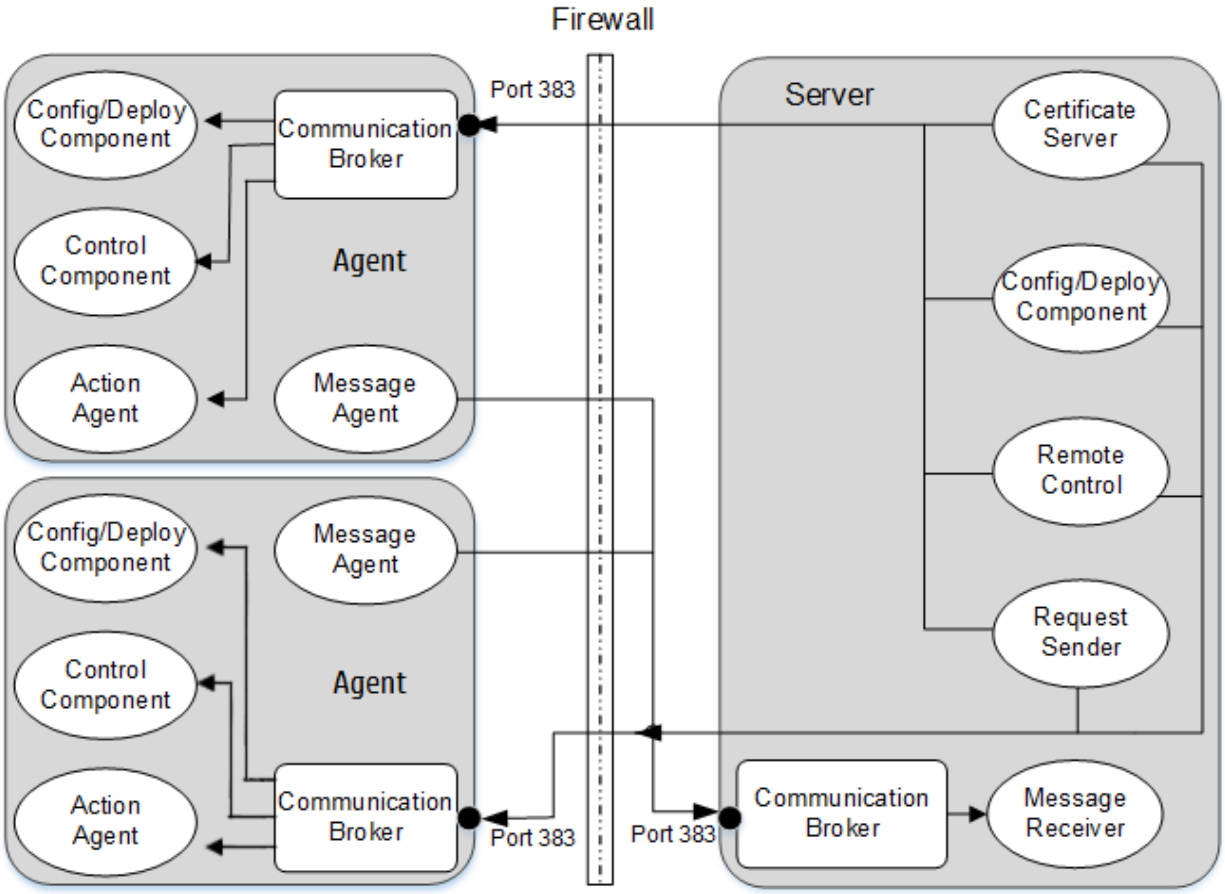
available in every environment today, and familiar to every IT administrator.

- HPOM message format based on XML and SOAP used for message security from the HTTPS agent to the HP Operations management server.
- IP independence or dynamic IP (DHCP). Managed nodes can be identified by their unique `OvCoreID` and not necessarily by their IP addresses.
- No need for additional investments (training, additional software).
- HP Operations standard control and deployment mechanism.
- HP Operations standard logging capability.
- HP Operations standard tracing capability.

HP Operations HTTPS Agent Architecture

The following graphics illustrate the architecture of the HTTPS communication in HPOM.

Figure 8: HTTPS Agent Components and Responsibilities

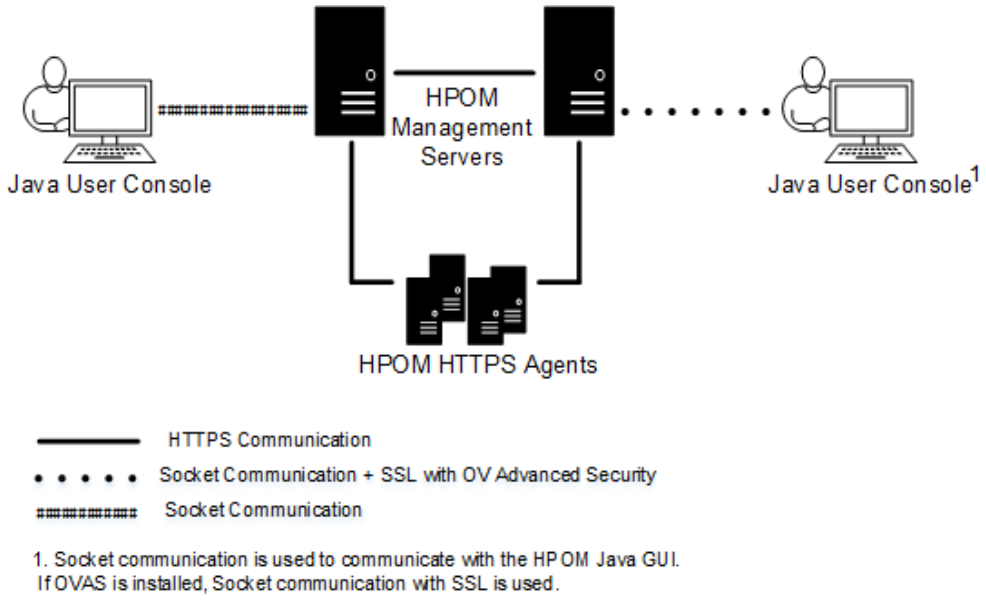


HTTPS Communication in HPOM

HTTPS 1.1 based communications is the latest communication technology used by HP BTO Software products and allows applications to exchange data between heterogeneous systems.

HP BTO Software products using HTTPS communication can easily communicate with each other, as well as with other industry-standard products. It is also now easier to create new products that can communicate with existing products on your network and easily integrate with your firewalls and HTTP-proxies. Figure 9 illustrates an example of HTTPS communication.

Figure 9: Communication Overview in HP Operations Manager



Advantages

HTTPS communication provides the following major advantages:

- "Firewall Friendly" below
- "Secure" on the next page
- "Open" on page 97
- "Scalable" on page 97

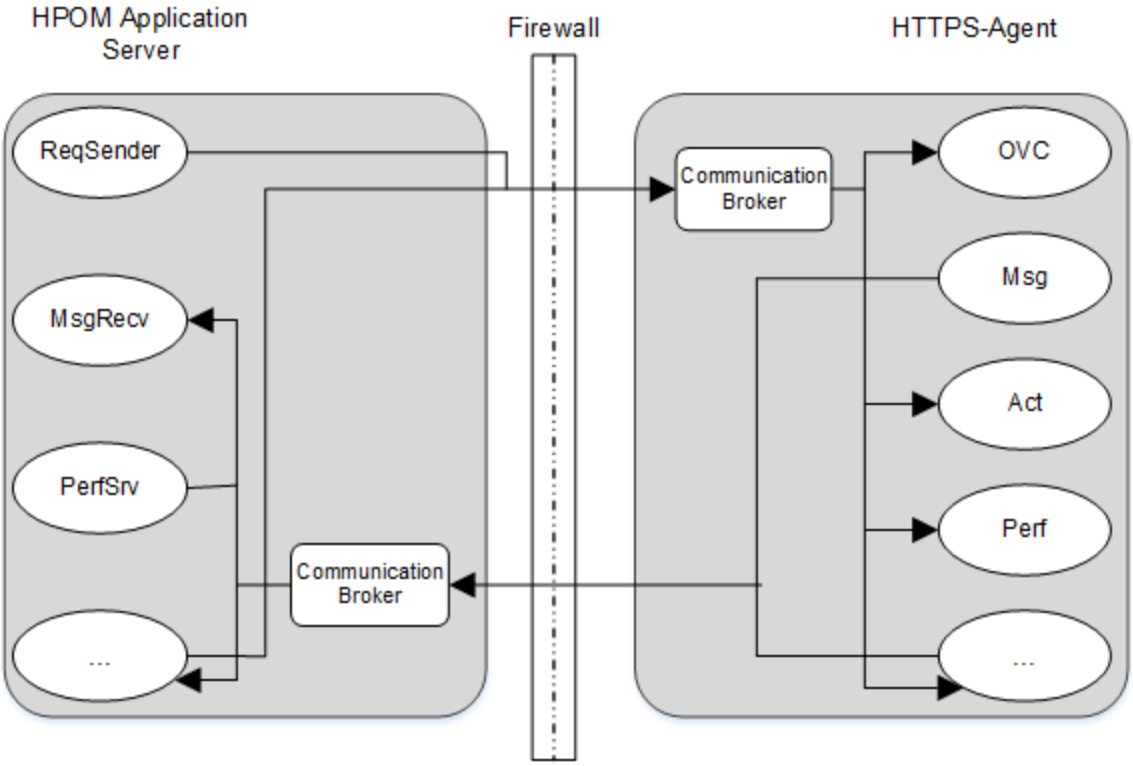
Firewall Friendly

More and more organizations need to cross firewalls in a safe, secure, and easily manageable way. Most of these organizations are very familiar and comfortable with HTTP, HTTP proxies, and firewalls.

Their IT environments are already configured to allow communication through HTTP proxies and firewalls. By focusing on technology that is already a part of most IT infrastructures, it helps you to be more efficient and effective, without the need for new training. The end result reduces support and maintenance costs, while simultaneously creating a highly secure environment without significant effort.

Figure 10 illustrates crossing a firewall using HTTPS-communication.

Figure 10: Crossing a Firewall with HTTPS Communication



Secure

HP Operations HTTPS communication is based on the TCP/IP protocol, the industry standard for reliable networking. Using the Secure Socket Layer (SSL) protocol, HTTPS communication uses authentication to validate who can access data, and encryption to secure data exchange. Now that businesses are sending and receiving more transactions across the Internet and private intranets than ever before, security and authentication assume an especially important role.

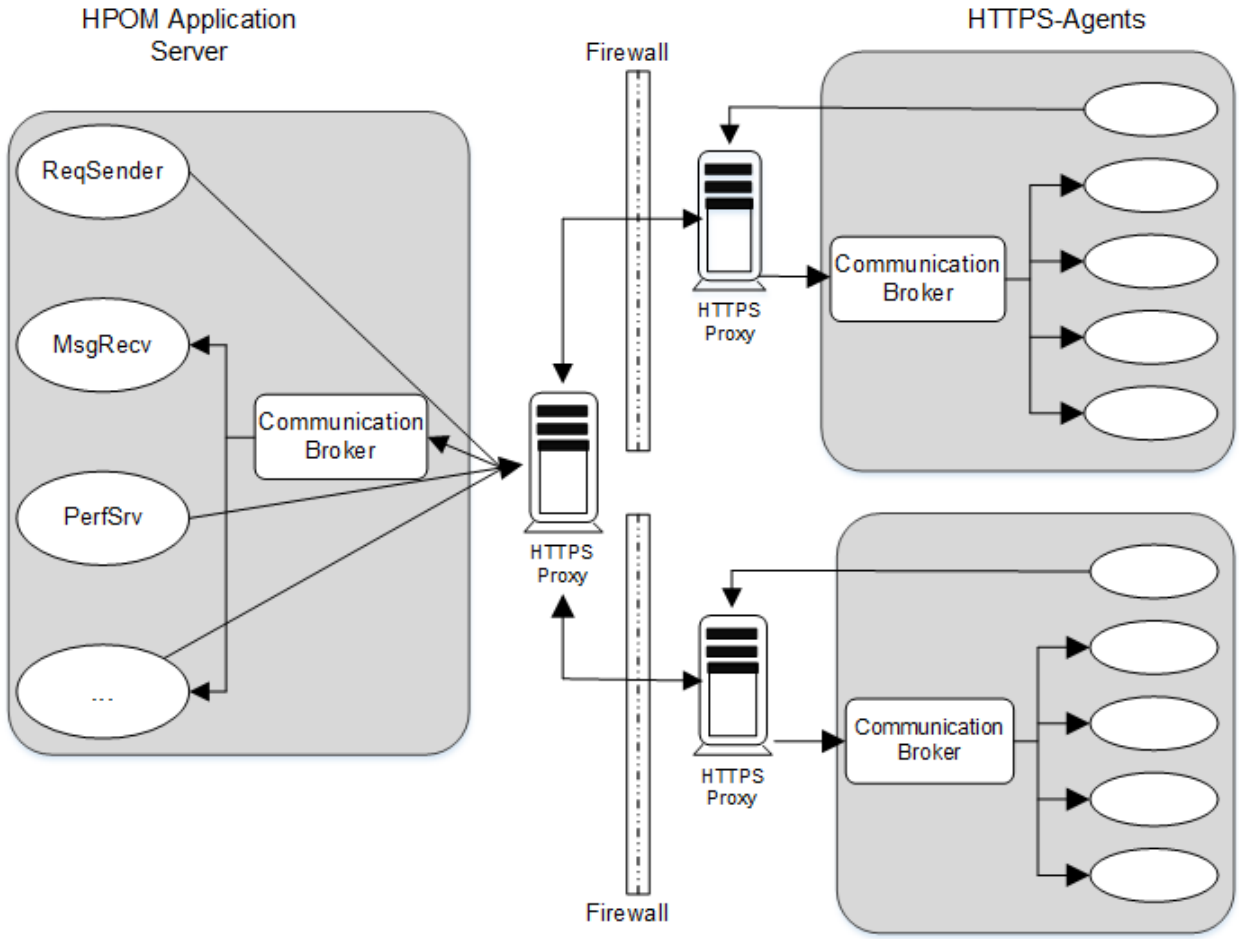
HP Operations HTTPS communication meets this goal through established industry standards. HTTP protocol and SSL encryption and authentication insure data integrity and privacy. By default, data is compressed, ensuring that data is not transmitted in clear text format, even for non-SSL connections.

In addition:

- All remote messages and requests arrive through the Communication Broker, providing a single port entry to the node.
- Restricted bind port range can be used when configuring firewalls.
- Configure one or more standard HTTP proxies to cross a firewall or reach a remote system when sending messages, files or objects.

Figure 11 illustrates crossing firewalls using standard HTTPS proxies.

Figure 11: Crossing a Firewall using External HTTPS Proxies



To work with HTTPS communication and proxies, you will need to:

- Configure HTTP proxy servers.
- Implement SSL encryption.
- Establish server side authentication with server certificates.
- Establish client side authentication with client certificates.

How you do this in HP Operations is described in the following sections.

Open

HP Operations HTTPS communication is built on the industry standard HTTP 1.1 protocol and SSL sockets. HP Operations adherence to open standards, such as HTTP, SSL and SOAP, allows you to maximize the use of your current HTTP infrastructure.

Note: Content filtering for HPOM agents is not supported.

HTTP proxies are widely used in today's networks. They are workhorses to help safely bridge private networks to the Internet. The use of HTTP allows HP Operations to slot into and take advantage of current infrastructures.

Scalable

HP Operations HTTPS communication is designed to perform well, independent of the size of the environment and the number of messages sent and received. HP Operations HTTPS communication can be configured to suit the environment within which it is to work. Large applications are able to handle many simultaneous connections while consuming the minimum of resources. If the maximum number of configured connections is exceeded, an entry in a logfile is created from which a warning message can also be raised.

Communication (Broker) Architecture

The Communication Broker acts as a proxy on the local managed node and provides a central point of entry to the managed node for all applications on that managed node. Applications that want to receive data register an address with the Communication Broker. The registration defines the port number, protocol, bind address, and base path the application wants to receive data on. Other applications, local or remote, either query the Communication Broker for the location of the application or use the Communication Broker as a proxy to forward the request to registered applications. The Communication Broker loads configuration data from the standard HP Operations Configuration File.

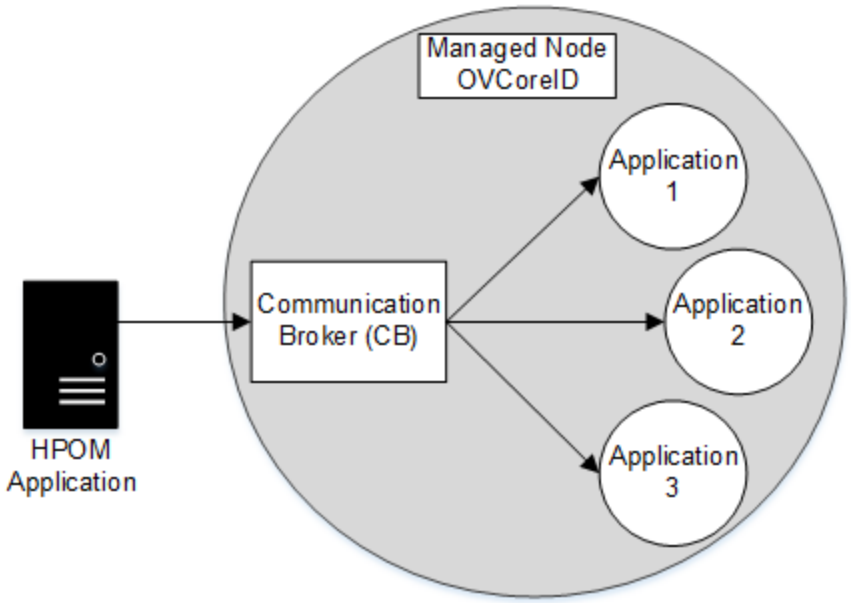
The Communication Broker has the following characteristics:

- The Communication Broker provides a single port solution for the managed node. Requests for all registered servers on this managed node can be directed through the Communication Broker. The Communication Broker transparently forwards the request to the registered server in the same way as an HTTP proxy forwards an HTTP request. The default port for the Communication Broker is 383 but can be changed.
- For higher security on UNIX systems, `chroot` can be used at start up of the Communication Broker. `chroot` restricts the part of the file system visible to the Communication Broker process by making

the specified path act as the root directory, thus reducing exposure to hackers.

- The Communication Broker can be run as non-root on UNIX systems if its port number is greater than 1024.
- The Communication Broker can be configured to run as root-only on UNIX systems to open its port and then switch to a non-root user for all other operations.
- The Communication Broker can be:
  - Started as a daemon on UNIX systems.
  - Installed as a Windows Service on Windows systems.
- Control commands for the Communication Broker can be restricted to the local managed node only.
- The Communication Broker applies SSL encryption of data transmission over the network.
- The Communication Broker applies SSL authentication through guaranteed identity of senders and receivers.

Figure 12: Communication Broker Architecture



A Communication Broker configures a minimum of one port for accepting incoming data to a managed node. The port is associated with an OVCoreID to identify the managed node. The Communication Broker can be configured to open multiple ports for high availability managed nodes. Each port can have a different identity associated with it. If SSL is enabled, the port is configured with X509 certificates. These certificates allow connecting applications to verify the identity of both message senders and receivers.

All applications on the current managed node that register with the Communication Broker are automatically registered for all active incoming ports opened by the Communication Broker.

The port associated with the default namespace, `bbc.cb`, is automatically activated on startup of the Communication Broker. Other ports can be activated or deactivated dynamically after startup. See the command line interface parameters for the Communication Broker for details.

Security Concepts

This section describes the following HPOM security concepts:

- "HTTPS-based Security Components" below
- "Remote Actions" on page 102
- "Roles and Access Rights" on page 103

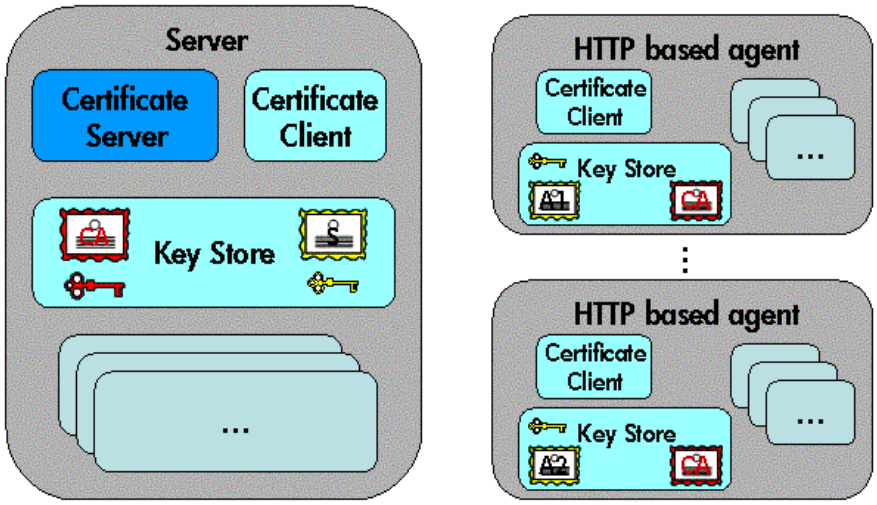
HTTPS-based Security Components

Managed nodes must have a valid, industry standard, X509 certificate issued by the HP Certificate Server to be able to communicate with HP Operations management servers. Certificates, signed by 1024 bit keys, are required to identify managed nodes in a managed environment using the Secure Socket Layer (SSL) protocol. The “SSL handshake” between two managed nodes only succeeds if the issuing authority of the certificate presented by the incoming managed node is a trusted authority of the receiving managed node. The main communication security components responsible for creating and managing certificates are:

- HP Certificate Server
- HP Key Store
- HP Certificate Client

Figure 13 illustrates these components:

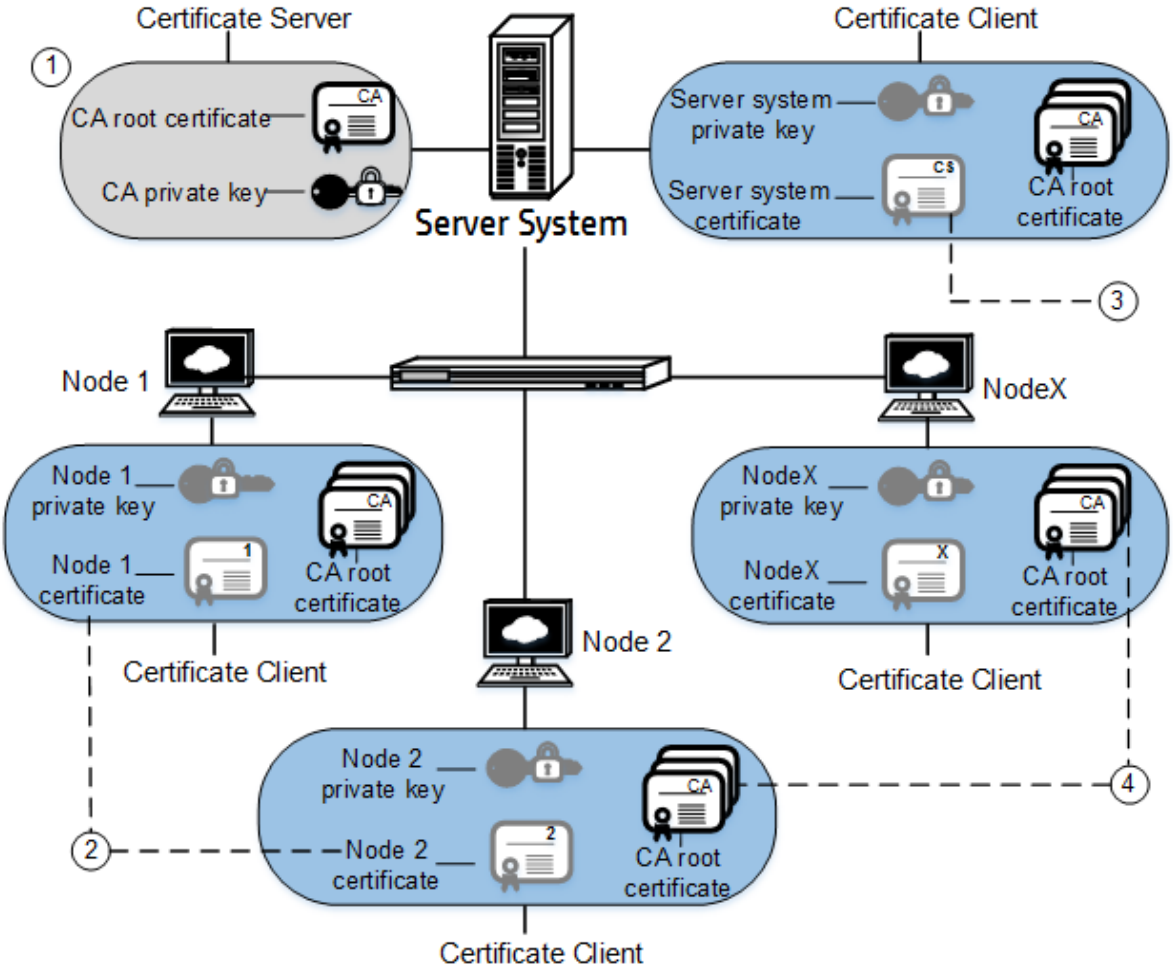
Figure 13: Components of Authenticated Communication



For each HP Operations system (managed node or server) `OvCoreId` is used as a unique identifier and is contained in the corresponding managed node certificate. `OvCoreId` is allocated its value during installation.

Figure 14 illustrates an environment for authenticated communication:

Figure 14: Environment for Authenticated Communication



1. A server system hosts the Certificate Server, which contains the needed certification authority (CA) functionality.
2. Every system has a certificate that was signed by the Certificate Server with the certification authority private key.
3. The server system also needs a certificate to prove its identity.
4. Every system has a list of trusted root certificates, which must contain at least one certificate. The trusted root (CA) certificates are used to verify the identity of the communication partners; a communication partner is only trusted if the presented certificate can be validated using the list of trusted certificates.

A list of trusted root certificates is required, when the certificate client is being managed by more than one HP Operations management server. For instance, when a managed node is managed simultaneously by multiple HP Operations management servers.

Certificates

There are two types of certificates:

- Root certificates
- Managed node certificates

A root certificate is a self-signed certificate, containing the identity of the certification authority of the certificate server. The private key belonging to the root certificate is stored on the certificate server system and protected from unauthorized access. The certification authority uses its root certificate to digitally sign all certificates.

Every HTTPS managed node in the managed environment receives a managed node certificate issued by a certificate server, a corresponding private key stored in the file system and the root certificates valid in its environment. The certificate client running on the managed node ensures this. For more information about the certificate client, see the HP Operations Agent documentation.

HP Certificate Server

The certificate server is responsible for the following:

- Creating and installing self-signed root certificates.
- Importing self-signed root certificates from the file system.
- Storing the private keys of root certificates.
- Granting or denying certification requests.
- Creating a new certificate and a corresponding private key or creating an installation key for manual certificate installation.
- Offering a service for clients to automatically retrieve trusted root certificates.

Certification Authority

Note: Every HP Operations management server is automatically configured as a Certificate Authority. The default setting for `sec.cm.client:CERTIFICATE_SERVER` for every agent is its own HP Operations management server.

The certification authority is part of the certificate server and is the center of trust in certificate management. Certificates signed by this certification authority will be regarded as valid certificates and

therefore be trustworthy. The certification authority must be hosted in a highly secure location. By default, it is installed on the system hosting the HP Operations management server.

Since the certification authority is the root of trust, it operates with a self-signed root certificate. This root certificate and the corresponding private key are created and stored on the file system with the level of protection to allow the certification authority to operate. After the certification authority is successfully initialized, it is responsible for signing granted certificate requests using its root certificate.

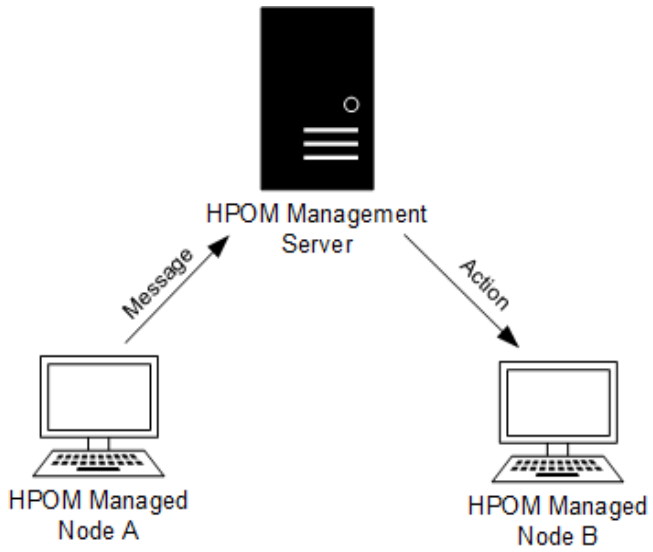
Remote Actions

A remote action is an automatic action or an operator-initiated action that is attached to an HPOM message sent by managed node A and configured to run on managed node B. The execution of such actions can be controlled by using the `remactconf.xml` file, which you can find in the following location:

`/etc/opt/OV/share/conf/OpC/mgmt_sv/remactconf.xml`

Figure 15 shows how managed node A sends a message to the HP Operations management server, which then executes the action on managed node B.

Figure 15: Example of Remote Actions



One HP Operations management server of a service provider must be able to manage the environments of several of its customers, while ensuring that a system located in one customer's segment cannot trigger any actions in any other customer's segment. Therefore, on the HP Operations management server, it is possible to configure on which systems the HP Operations management server is allowed to execute an action, so that remote actions designed for the managed nodes of one customer are not executed on the managed nodes of another customer.

For detailed information about configuring the management server for remote action authorization and the security mechanisms that prevent the misuse of remote actions, see the *HPOM Administrator's Reference*.

Roles and Access Rights

In general, a role grants the right to perform a certain task; for example, in HPOM environments, the rights to execute actions, deploy files, or configure settings. Each preconfigured HPOM role described below has a default set of access rights that can be changed as explained in the *HPOM Administrator's Reference* and HP Operations Agent documentation.

About Roles

An HP Operations management server can assume a preconfigured HPOM role. The mapping between management servers and roles is defined in the `sec.core.auth` namespace and, in MoM environments, in the responsible manager policy.

An HPOM environment includes the following preconfigured roles:

- **Local User Role**

  The local user has all rights, assuming appropriate system rights are given, for example `root`.
- **Initial or Authorized Manager Role**

  This manager has all rights and is set up at install time. This role is defined by the `MANAGER` and `MANAGER_ID` settings in the security namespace `sec.core.auth`. There can be only one initial manager.
- **Secondary Manager Role** (MoM environments only)

  A secondary manager has all rights including action execution and configuration deployment. There can be multiple secondary managers defined in the responsible manager policy. The initial manager and the secondary managers make up the group of possible configuration servers.
- **Action-allowed Manager Role** (MoM environments only)

  An action-allowed manager has no other rights than the action execution right. There can be multiple action-allowed managers defined in the responsible manager policy.

About Access Rights

Access rights are the rights to, for example, execute actions, deploy files, and configure settings. The rights are mapped to the HP Operations management server roles described in "About Roles" above.

For more information about avoiding unattended configuration deployment and denying remote access, see the *HPOM Administrator's Reference*.

For more information about restricting access rights, see the *HPOM Administrator's Reference* and the HP Operations Agent documentation.

Authorization Mappings

The following table lists the individual default access rights for each HPOM management server role.

Table 4: Authorization Mapping

Component		**Value**	**Initial Manager**	**Secondary Manager**	**Action-allowed Manager**
<comp_name>	**Right**	*<dec_value>*	*<HPOM_mgr_role>*		
Control (ctrl)	Start	1	yes	yes	no
	Stop	2	yes	yes	yes
	Status	4	yes	yes	no
	Notify	8	yes	yes	no
	Default value:	**15**	**15**	**15**	**2**

Authorization Mapping, continued

Component	Right	Value	Initial Manager	Secondary Manager	Action-allowed Manager
<comp_name>	**Right**	*<dec_value>*	*<HPOM_mgr_role>*		
Config (conf)	Install policy	1	yes	yes	no
	Remove policy	2	yes	yes	no
	Enable policy	4	yes	yes	no
	Disable policy	8	yes	yes	no
	List policies	16	yes	yes	yes
	Update policy header	32	yes	yes	no
	Read configuration setting	64	yes	yes	yes
	Write configuration setting	128	yes	yes	no
	Sign policy	256	yes	yes	no
	Default value:	**511**	**511**	**511**	**80**

Authorization Mapping, continued

Component	Right	Value	Initial Manager	Secondary Manager	Action-allowed Manager
<comp_name>		*<dec_value>*	*<HPOM_mgr_role>*		
Deploy (depl)	Deploy file	1	yes	yes	no
	Remove file or directory	2	yes	yes	no
	Get file	4	yes	yes	no
	Execute file	8	yes	yes	no
	Deploy package	16	yes	yes	no
	Remove package	32	yes	yes	no
	Upload package	64	yes	yes	no
	Download package	128	yes	yes	no
	Get inventory	256	yes	yes	yes
	Modify inventory	512	yes	yes	no
	Get node information	1024	yes	yes	yes
	Default value:	**2047**	**2047**	**2047**	**1280**
Action agent (eaagt.actr)	Execute action	1	yes	yes	yes
	Default value:	**1**	**1**	**1**	**1**

Managing HTTPS Nodes

The HP Operations management server can perform the following functions on HTTPS nodes:

- Remote control of HTTPS agents.
- Remote and manual installation of HTTPS agents.
- Remote and manual patch installation and agent upgrade.

- Remote and manual configuration deployment.
- Support of multiple parallel configuration servers for HTTPS agents.
- Heartbeat polling.
- Security management of HTTPS nodes.
- Support of HTTPS nodes through the HP Operations management server APIs and utilities.

The following sections explain some of these concepts for HTTPS nodes:

- "Configuration Deployment to HTTPS Nodes" below
- "Remote Control of HTTPS Nodes" on page 110

For more information on managing the HTTPS nodes, see the HP Operations Agent documentation.

Configuration Deployment to HTTPS Nodes

The following sections explain the configuration management concepts introduced with the HTTPS agents.

Policy Management

The HPOM policies are managed in a way which allows the generic policies to be registered in the database, assigned to managed nodes, and distributed to them. Policies can have multiple versions on the HPOM 9.xx management server, and are organized in a tree-like structure. Policies can also contain category assignments. **Categories** unify the related instrumentation files and make their distribution to the managed nodes easier.

To learn about policies and policy types, see "HPOM Policies" on page 49.

For information on the administration tasks related to policies, managing the multiple versions and the distribution of the HPOM configuration to managed nodes, see the *HPOM Administrator's Reference*.

Instrumentation Management

The directory for executables on the HP Operations management server is located under:

```
/var/opt/OV/share/databases/OpC/mgd_node/
```

No instrumentation directory is created and the directories actions, commands, and monitors are used, unless the categories for the instrumentation data are created.

Typically, action, command, and monitor executables are referenced in HPOM policies. As long as these executables are not referred with their full path in policies, this change is transparent, because the new locations of the binaries is also added to the path variables of utilities like the HPOM action agent, monitor agent and logfile encapsulator.

Files from the monitor directory on the HP Operations management server are installed on the agent with the rights `744`, all others with the rights `755`.

The configuration management process can also update running executables. Scripts and binaries of running executables are renamed and allowed to complete their tasks. Subsequent execution of these programs use the newly installed files.

For more information on the distribution and management of multiple versions of the instrumentation data, see the *HPOM Administrator's Reference*.

Manual Installation of Policies and Instrumentation

It is not possible to copy policy data directly to a managed node because the agent must receive the configuration data in a secured format. This is required to avoid illegal manipulation of configuration data by unauthorized persons on the managed nodes.

The `opctmpldwn` tool is used to prepare the manual installation of policies on the HP Operations management server. The output data is stored in a directory on the management server system dedicated to the managed node.

`opctmpldwn` handles HTTPS nodes in the following way:

- The `nodeinfo` and `mgrconf` data are regarded as policies and therefore contained in the directory mentioned above.
- A policy is encrypted with a node-specific key.

HTTPS Agent Distribution Manager

`opcbbcdist` is the configuration management adapter between the HP Operations management server and the HTTPS agents. Its main functions are:

- Convert templates into policies.
- Create instrumentation from existing actions, commands, and monitors.
- Convert ECS templates into policies and their associated circuits.
- Switch `nodeinfo` settings into the XPL format used on HTTPS nodes.

`opcbbcdist` uses the internal file system interface, `/var/opt/OV/share/tmp/OpC/distrib`, to get the information about what data should be deployed. It distinguishes between the four configuration categories:

- Policies/templates
- Instrumentation actions/commands/monitors
- nodeinfo
- mgrconf

`opcbbcdist` only accepts requests from other HP Operations management server components of the form `deploy configuration types xyz to node abc`. These requests may be issued by a configuration API or by `opcragt -update` and `opcragt -distrib`.

`opcbbcdist` possesses an automatic retry mechanism which is started if it was not possible to reach a node and new data is present for it. You can also manually trigger a retry by calling `opcragt -update`.

When `opcbbcdist` completes a task for a certain node, you get a message in the browser confirming correct distribution of configuration data. If tasks are not completed, messages, such as `Node Unreachable`, are displayed.

`opcbbcdist` transfers instrumentation data first, then policies. This is done to avoid synchronization issues when an executable is referenced in a template. In addition `opcbbcdist` follows a simple transaction model: only if all data of a certain configuration type is successfully deployed, is the next category processed. The distribution of one configuration type is regarded as one transaction. If a transaction fails, it is rolled back and retried later. This schema is also applied when `opcbbcdist` is stopped due to HP Operations server shutdown.

Configuration Push

The HP Operations management server triggers all configuration deployment tasks to HTTPS nodes. The HP Operations server pushes configuration data down to the agent and there is only out-bound communication. The more secure HP Operations management server triggers the managed nodes.

A disadvantage is that a managed node must run with old data in the case of the system not being reachable when new configuration was distributed. The HP Operations management server must poll all nodes for which configuration is present but could not be delivered. The HP Operations management server does this task:

- at least once an hour per pending node.
- when the server is restarted.
- when the configuration push is explicitly triggered by `opcragt -update`, `opcragt -distrib`, or by directly calling the API associated with the command.

A monitor called `dist_mon.sh` checks for pending distributions. If any data in the configuration transfer directory:

`/var/opt/OV/share/tmp/OpC/distrib`

is older than 30 minutes, a message is displayed that specifies the managed node where a distribution is pending.

Delta Distribution

By default in HPOM, the distribution process, known as delta-distribution, only deploys data which has been modified or added since the last configuration transfer. This minimizes the amount of data transferred and reduces the number of reconfiguration requests for interceptors and other sub agents. If required, the complete configuration can be re-deployed to the managed node.

In the delta-distribution mode, the HP Operations management server requests the policy inventory of the managed node and time stamps of the last instrumentation distribution. The policy inventory is compared with the policy assignment list and `opcbbcdist` computes and executes the required policy removal and installation tasks for the node. For instrumentation deployment, the time stamp of the last deployment is compared with the time stamps in the management server instrumentation directories. All files on the HP Operations management server that are newer than the corresponding file on the managed node are distributed. No instrumentation data is ever removed from the managed node, except if the `opcragt -purge` command line command and option is applied.

Remote Control of HTTPS Nodes

The `opcragt` utility is used to control agents from the HP Operations management server. The operations includes start, stop, get status, primary manager switch, get and set configuration variables, as well as configuration distribution. There is a wrapper called `opcagt` on HTTPS nodes. This utility can be used to perform remote control tasks by application launch from the operator's desktop. It allows to setup a common action definition for any kind of managed nodes.

Subagents are identified by names on HTTPS nodes. Therefore, you can specify aliases of the form:

<alias> <maps_to>

in the configuration file:

`/etc/opt/OV/share/conf/OpC/mgmt_sv/subagt_aliases`

The entries `1 EA` and `12 CODA` are pre-defined. To transform the `-id 1` into `-id EA` for HTTPS managed nodes, type the following command:

opcragt -status -id 1 *<node_list>*

Heartbeat Polling

The `opchbp` tool is used to enable or disable HPOM heartbeat polling for one or more managed nodes. It is also used to set the type of heartbeat polling and define if the agent should send alive packets.

To check the heartbeat polling status of all managed nodes, run the following command:

/opt/OV/bin/OpC/opchbp -status

Heartbeat polling can be performed on all managed nodes that are in the HPOM node bank except external nodes. The HP Operations management server performs heartbeat polling for those managed nodes on which the HP Operations Agent is installed.

Disabling heartbeat polling is useful when the managed node system is shut down, when HPOM is completely stopped (including the control agent), or during the manual HPOM software upgrade on the managed node.

If heartbeat polling is enabled, HPOM generates a message that the control agent is not responding (that is, is not reachable), and succeeding warning messages are suppressed automatically. When the control agent is running again, the message is automatically acknowledged by HPOM.

For more information about the `opchbp` tool and its options, see the *opchbp* manual page.

Creating and Distributing Certificates

Certificates are needed for network communication using the Secure Socket Layer (SSL) protocol with encryption. Server and client authentication are enabled. Managed nodes of the managed environment are identified using certificates. The “SSL handshake” between two managed nodes only succeeds if the issuing authority of the certificate presented by the incoming managed node is a trusted authority of the receiving managed node.

You can install certificates automatically, and manually. Certificate installation is monitored with HPOM messages. After a certificate request has been granted automatically, a notification message confirming the successful deployment of a certificate is sent to the message browser. If a certificate request is not automatically granted, a message in the message browser indicates the reasons for request denial and the steps that an administrator must take to solve the problem.

Certificates are managed with the `ovcm` and `opccsa` command line utilities. You can either grant, deny, list, or delete certificate requests, or map certificate requests with the corresponding node from the node bank.

For information on how to work with certificates, see the *HPOM Administrator’s Reference* and the HP Operations Agent documentation.

Virtual Nodes in HPOM

Clusters are multiple systems, or nodes, that operate as a unit to provide applications, system resources, and data to users. In modern cluster environments such as Veritas Cluster, Sun Cluster, or TruCluster, applications are represented as compounds of resources. Those resources construct a resource group, which represents the application running in cluster environment. Each resource has a special function in this compound.

There is a common mechanism to model applications running in cluster environments.

Terminology

The following High Availability terms and abbreviations are used in HPOM.

General High Availability Terms

HA (High Availability)	High availability is a general term used to characterize environments that are business critical and which are therefore protected against downtime through redundant resources. Very often, cluster systems are used to reach high availability.
HA Cluster (High Availability Cluster)	High availability clusters are hardware resources grouped together by a cluster management application such as HP ServiceGuard (HP/SG), Veritas Cluster, and Sun Cluster. Redundant resources are used to guarantee high availability through, for example, multiple computers, redundant network connections, and mirrored storage devices.
HA package | HA resource group | Cluster package | HARG	These terms are all used to denote a resource defined in the 'cluster world' which can be linked to an application instance. It runs on a cluster and can be switched from one cluster node to another. A cluster package is usually also linked to an element from the 'networking world' known as a virtual node.
Virtual Node	A virtual node is the network representation of an application package running on an HA cluster. A virtual node typically has a hostname and an IP address, is known to the name resolution and can be addressed like an ordinary system.
Physical Node | Cluster Node	This is one single system belonging to the cluster hardware and acting as a potential host for the HARG. A set of physical nodes makes up the cluster.
Switch-over	Controlled switch of a cluster package from one cluster node to another, for example, due to load balancing.
Fail-over	Unplanned switch of a cluster package from one cluster node to another, for example, due to an application error.

Cluster Terms Used in HPOM

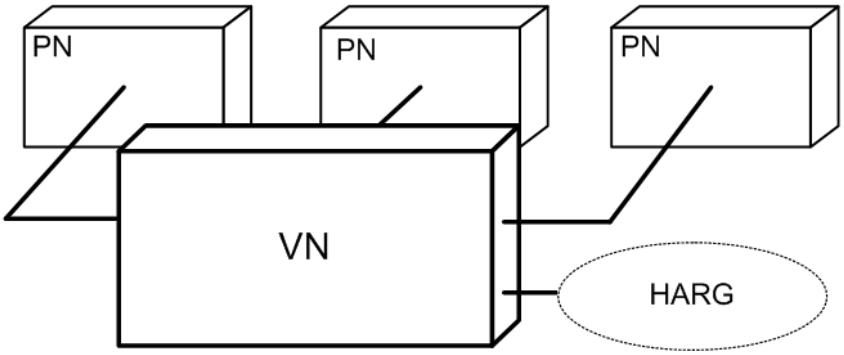
HPOM Virtual Node	An HPOM virtual node is a concept to represent HA packages in the HPOM database. A virtual node is assigned the hostname and IP address

	belonging to the HA package. An HPOM virtual node has an `HARG Name` attribute. Typically, the value of this attribute is the HA resource group name. An HPOM virtual node is comprised of the physical nodes where the HA resource group can run on the cluster.
CIAw (Cluster Awareness)	Cluster awareness is HPOM functionality which is used to monitor start and stop events of cluster packages. The CIAw module must be installed on each physical node of a cluster that is to be monitored, as the cluster awareness software only monitors start and stop events on the LOCAL node. The CIAw module is part of the HPOM HTTPS agent and the functionality is located in the `ovconfd` process.
HARG Name (High Availability Resource Group Name)	HARG Name is a string attribute which can be assigned to an HPOM virtual node in the HPOM database. A HARG name in HPOM must be identical to the HA resource group's name in a cluster. This name is the link between the HPOM world (HPOM database) and the cluster world.

Virtual Node Concepts

An HPOM virtual node can be regarded a group of physical nodes linked by a common `HA Resource Group` name. The Cluster Awareness (CIAw) extension of the agents on these physical nodes can switch the policies on a physical node as the package itself switches within the virtual node.

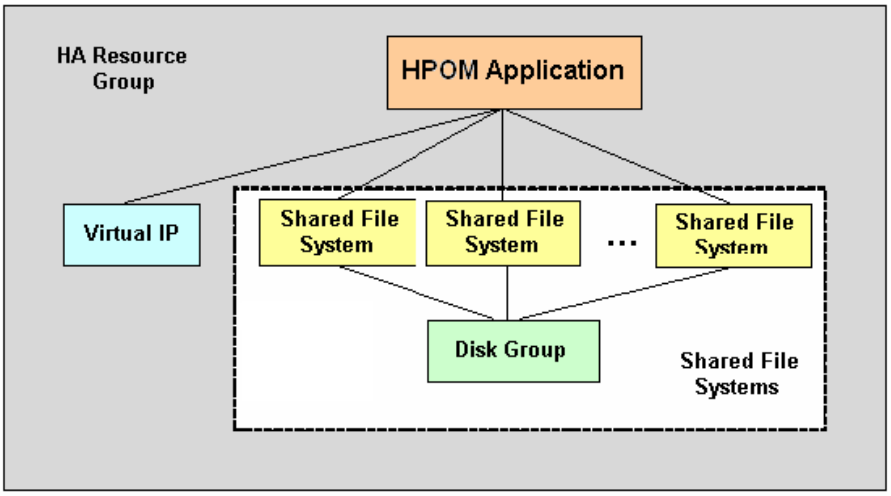
Figure 16: Virtual Nodes



The `HA Resource Group` name linking the managed node provides the following advantages:

- Events detected in the scope of the `HA Resource Group`, for example, by policies assigned to the virtual node, may receive that name as the originating node.
- Correct filtering and highlighting on the management station GUI.
- Provide appropriate service names and message key correlations for true management of the cluster.

Figure 17: HA Resource Group



Note: This functionality is only available for HTTPS nodes.

A virtual node can be associated with just one HA resource group name.

An HA resource group name can be assigned to more than one virtual node, but these virtual nodes should not share any common physical nodes. This is because any policy assigned to both virtual nodes would receive the same HARG a second time and the cluster awareness of the agent would not be able to distinguish the virtual nodes.

For information on how to work with virtual nodes, see the *HPOM Administrator's Reference*.

Proxies in HPOM

Firewall programs and their associated policies, located at a network gateway server, are gateways that are used to protect the resources of a private network from external users. Users of an intranet are usually able to access the approved parts of the Internet while the firewall controls external access to the organization's internal resources.

There are two basic categories of firewalls:

- IP packet filters that work on the network level.
- Proxy servers that work on the application level, for example, a web proxy.

A proxy is a software application that examines the header and contents of Internet data packets and takes necessary action required to protect the systems to which the data is directed. In conjunction with security policies, proxies can remove unacceptable information or completely discard requests.

There are significant security-related advantages of using Application Proxies. These include:

- A fine granularity of security and access control can be achieved as proxies examine packets at the application level. For example, it is possible to restrict specific types of file transfer such as `.exe` files.
- Proxies can provide protection against “Denial of Service” attacks against the firewall.

There are two commonly cited disadvantages of using proxies:

- Proxies require large amounts of computing resources in the hose system but this is no longer a practical issue as powerful computers are now relatively inexpensive.
- Proxies must be written for specific application programs and there may be programs for which proxies are not easily available.

A proxy server stops and inspects all information before letting it access the internal network. Therefore, by using a proxy, there is no direct connection between an internal network and the “outside” world. Users must authenticate to the proxy to be able to send out information.

When a client within the intranet attempts to make a request to the Internet, the proxy actually receives that request. Using Network Address Translation (NAT), the proxy changes the source IP address of the packet to that of the proxy server, which hides the identity of the users on the internal network from the outside. If the request meets the requirements of any established policies, the proxy server forwards this request to the desired address. When a response is received, the process is reversed. As long as the incoming request is deemed to be safe, the request is forwarded to the target client on the network. The source address of the response remains unchanged but the destination address is changed back to that of the requesting machine within the firewall. This confers a dramatic increase in security for the network because there is no direct, uncontrolled route to any network systems.

There are two basic types of proxy servers:

- **Single-Homed Host**

  The proxy server has only one network card and address, and it is the responsibility of the Internet router to forward requests to the proxy server and block all other information to the network.
- **Dual-Homed or Multi-Homed Host**

  The proxy server is associated with more than one network card. Requests from the internal network are directed to one of the network cards. Information that comes from the Internet is received by the other network card. There is no routing setup between the network cards, so there is no direct connection between the incoming and outgoing information. The proxy server is responsible for deciding what is sent and to where it is sent.

For more information on configuring proxies in HPOM and setting proxies on HPOM management server, see the *HPOM Administrator’s Reference*.

Tracing HPOM

This section describes the concepts of tracing HPOM in the following sections:

- "Tracing HPOM" below
- "HPOM Trace-Enabled Applications" on the next page
- "Server and Agent Applications" on page 119

Tracing HPOM

To help you investigate the cause of problems, HPOM provides problem tracing. Trace logfiles can help you pinpoint when and where problems occur, for example, if processes or programs abort, performance is greatly reduced, or unexpected results appear.

The following tracing mechanisms can be used with HPOM:

- HP tracing is the mechanism for tracing the latest HP BTO Software products and will be incorporated into all future products. HP tracing can be used to help solve problems with HTTPS agents and the HP Operations management server.

  Tracing allows remote access using a proprietary format. SSL encryption is not used. By default, the communication port is 5053.

  For more information about HP-style tracing, see "HP-Style Tracing Overview" below.
- HPOM-style tracing, using configuration settings, can also be used to problem solve HTTPS agents as well as the HP Operations management server. The configuration settings set with the `ovconfchg` command. For more details, see the *HPOM Administrator's Reference*.

HP-Style Tracing Overview

HP tracing implements a hierarchy of elements: `Applications`, `Components`, `Categories` and `Attributes`. By specifying a combination of these in the trace GUI or in a trace configuration file, the area of interest can be traced.

Table 5 illustrates how these elements relate to HPOM components, processes, and areas.

Table 5: Tracing Terminology

Name	HPOM-Style Name	Examples
Application	Process, `OPC_TRC_PROCS` and `OPC_DBG_PROCS`	`opcmsga`, `ovpolicy`

Tracing Terminology, continued

Name	HPOM-Style Name	Examples
Component	n.a.	`opc`, `eaagt`
Subcomponent	Trace Areas, `OPC_TRACE_AREA`	`actn`, `msg`, `init`, `debug`
Category	`OPC_TRACE TRUE`	`Trace`
Attribute	n.a.	`Info`, `Warn`, `Error`, `Developer`, `Verbose`

There are two ways to trace HPOM using HP tracing:

- Configure Remote Tracing Using the Windows Tracing GUI. For more information, see the HP Operations Agent documentation.
- Configure Manual Tracing Using Trace Configuration Files. For more information, see the *HPOM Administrator's Reference*.

HPOM Trace-Enabled Applications

All HPOM processes use HP Tracing. The HPOM Trace Enabled processes can be divided into three groups:

- The Server processes
- The Agent processes
- The processes that link with a lower level component which implemented XPL Tracing.

There are no pre-configuration steps required to enable tracing within HPOM. This is accomplished by either adding XPL Tracing into the HPOM code base or by incorporating core functionality from a foundation component and linking with the corresponding library. In the case where XPL Tracing was added to the HPOM code base, the existing tracing was converted to XPL Tracing. In cases where functionality from a foundation component was added, the XPL Tracing incorporated into these foundation components is pulled into HPOM.

Table 6: HPOM Trace-enabled Applications on Management Server and Managed Nodes

Platform	Application Name		
UNIX/Linux	coda	ovas	ovconfget
	codautil	ovbbccb	ovcoreid
	ctrlconfupd	ovc	ovcreg
	logdump	ovcd	ovcs
	opc_getmsg	ovcert	ovdeploy
	opc_ip_addr	ovcm	ovpolicy
	opccrpt	ovconfchg	
	opcnls	ovconfd	

Table 7: HPOM Trace-enabled Applications on Management Server

Platform	Application Name		
UNIX/Linux	opc	opcdbck	opcsvcm
	opc_dbinit	opcdbinst	opcsw
	opc_dflt_lang	opcdbmsgmv	opcttnsm
	opc_rexec	opcdbpwd	opcuiadm
	opcactm	opcdispm	opcuiopadm
	opcagtdbcfg	opcforwm	opcuiwww
	opcagtutil	opchbp	ovoareqsdr
	opcauddwn	opchistdwn	
	opcbbcdist	opcmsgm	
	opccfgupld	opcmsgrb	
	opccsacm	opcnode	
	opccsad	opcragt	
	ovcd	opcservice	

Table 8: HPOM Trace-enabled Applications on Managed Nodes

Platform	Application Name		
UNIX/Linux	opcacta	opcmon	opcmsgi
	opceca	opcmona	opctrapi
	opcecaas	opcmsg	
	opcle	opcmsga	

Server and Agent Applications

The following sections describe the server and agent applications.

HP BTO Software and HPOM Specific Components

There are many components and sub-components defined for each application. The most important are `eaagt` and `opc`. Table 9 lists the Tracing Components which are defined for the server and agent processes.

Table 9: HPOM Server and Agent Components

HPOM Component Name	Component Description
`eaagt`	Event Action Agent
`opc`	Management Server Control

Table 10 lists the components defined for the shared components which have been incorporated into the product.

Table 10: HP BTO Software Shared Components

Application with Component and Subcomponent Names	
Black Box Communication	
bbc.cb	bbc.http.output
bbc.fx	bbc.http.server
bbc.fx.client	bbc.messenger
bbc.fx.server	bbc.rpc
bbc.http	bbc.rpc.server

HP BTO Software Shared Components, continued

Application with Component and Subcomponent Names	
bbc.http.client	bbc.soap
bbc.http.dispatcher	
Control Component	
ctrl.action	ctrl.ovc
ctrl.autoshutdown	ctrl.process
ctrl.component	ctrl.rpcclient
ctrl.controller	ctrl.rpcserver
ctrl.main	ctrl.soap
ctrl.monitor	ctrl.xml
ctrl.monitorproxy	
Configuration Management Component	
conf.cluster	conf.ovconfd
conf.cluster.clioutputs	conf.ovpolicy
conf.config	conf.policy
conf.message	
Certificate Server Adapter	
CSA-CertRequestImpl	Csa-Main
CSA-CertReqContainer	csa.ovcmwrap
CSA-Database	Csa-RpcServer
Csa-Log	CSA-UpdateHandler
Security Core Component	
sec.cm.client	sec.core.base
sec.cm.server	sec.core.ssl
sec.core.auth	

HP BTO Software Shared Components, continued

Application with Component and Subcomponent Names	
Cross Platform Library	
xpl.cfgfile	xpl.net
xpl.config	xpl.runtime
xpl.io	xpl.thread
xpl.log	xpl.thread.mutex
xpl.msg	
Embedded Performance Agent	
coda	coda.mesa
coda.dataaccess	coda.mesainstances
coda.kmdatamatrix	coda_mesametricrdr
coda.localmesa	coda.mesarea
coda.logger	coda.prospector
Deployment Component	
depl	

HPOM Specific and XPL Standard Categories

HPOM trace areas are designated by HP BTO Software categories. In addition, a number of the standard categories are used by both HPOM processes and the lower level HP BTO Software components used by HPOM.

Table 11 TBD lists the tracing categories which are defined for the `eaagt` and `opc` components.

Table 11: HPOM opc and eaagt Sub-components

Sub-Component Name	Sub-Component Description
HPOM Specific Tracing Categories	
actn	Actions
agtid	IP independence using AgentID

HPOM opc and eaagt Sub-components, continued

Sub-Component Name	Sub-Component Description
alive	Agent alive checking
api	Configuration API
apm	Cluster APM
audit	Auditing
db	Database (dblib)
debug	Debug
dist	Distribution
fct	Function (control flow)
init	Initialization (e.g. err init, conf init)
inst	Installation
int	Internal
lic	Licensing
memerr	Problems with Memory allocation
memory	Rest of memory allocation
misc	Miscellaneous
mon	Monitor
msg	Message flow
name	Name resolution
nls	National Language Support (character set conversion,…)
ntprf	NT Performance trace
pdh	Performance data helper
perf	Performance
pstate	Policy and Source state changes

HPOM opc and eaagt Sub-components, continued

Sub-Component Name	Sub-Component Description
sec	Security
srvc	Service
wmi	Conversion of LE-Templates to WMI-Templates
Generic XPL Tracing Categories	
Trace	Generic traces
Proc	Procedure traces
Operation	Operational traces
Init	Initialization
Cleanup	Cleanup operation
Event	Event
Parms	Parameters
ResMgmt	Resource Management

Firewalls and HTTPS Communication

Firewalls are used to protect a company's networked systems from external attack. They usually separate the Internet from a company's private intranet. It is also quite common to implement multiple levels of firewalls to restrict access to the more trusted environments from those of lower sensitivity. For example, the research and finance departments may be contained in the environment of highest security, while direct sales may need to be easily accessible from the outside. Systems on the intranet are allowed, under certain conditions, to cross the firewall to access systems on the internet, for example located in the DMZ. The firewall can also allow systems on the Internet to cross the firewall and access systems on the private intranet. For either of these situations, the firewall must be configured to allow that operation. HP Operations HTTPS communication provides features that allow firewall administrators to configure HP Operations applications to communicate through firewalls.

Contacting an Application on the Internet from an Intranet Using an HTTP Proxy

An HP Operations HTTPS-based application on a private intranet wants to contact an application outside of the firewall on the public Internet or Demilitarized Zone (DMZ). The HP Operations application initiates the transaction and acts as a client contacting a server application on the Internet. The server application could be another HP Operations application acting as an HTTP server or any other HTTP server application. A common example of a client is a web browser on the private intranet wanting to contact a web server on the Internet. An HTTP proxy must be configured in the browser which forwards the request across the firewall and contacts the web server in the Internet. The firewall is configured to allow the HTTP proxy to cross the firewall. The firewall does not allow the web browser to directly cross the firewall. In the same way, HP Operations HTTPS communication applications can also be configured to use HTTP proxies to cross firewalls.

Contacting an Application on the Internet from an Intranet Without an HTTP Proxy

An HP Operations HTTPS-based application on a private intranet wants to contact an application outside of the firewall on the Internet without using an HTTP proxy. The firewall must be configured to allow the HP Operations application on the private intranet to cross the firewall. This is very similar to configuring a firewall to allow an HTTP proxy to cross the firewall. The firewall administrator may want to set source and target ports for the transaction to restrict communication across the firewall. The `CLIENT_PORT` configuration parameter specifying the source ports can be set from the HP Operations application when initiating the transaction. The target or destination port is defined in the URL (Uniform Resource Locator or Identifiers) address used to contact the HTTP server on the Intranet. This is the communication broker port on the target node.

Contacting an Application Within a Private Intranet from an HP Operations Application on the Internet

An HP Operations HTTPS-based application on the Internet wants to contact an application on a private intranet. This means that a firewall must be crossed from the outside and is usually only allowed by organizations under very restricted conditions set by the firewall administrator. The initiating or client application may do this using an HTTP proxy or go directly through the firewall. The HTTP proxy is outside the firewall and the firewall must be configured to allow the HTTP proxy to cross it. The HTTP proxy could either directly contact the server on the private intranet or go through another proxy, in a

cascading proxies arrangement. In either case, the HP Operations HTTPS communication client application is configured in the same way. However, the HTTP proxies must be configured differently.

Contacting an Application Within a Private Intranet from an HP Operations Application on the Internet Without Using HTTP Proxies

An HP Operations HTTPS-based application on the Internet wants to contact an application on the private intranet, but there is no HTTP proxy. The firewall must be configured to allow the HP Operations client application to cross the firewall. The firewall administrators may want to set the source and target ports for the transaction to restrict communication across the firewall. The `CLIENT_PORT` configuration parameter specifying the source port can be set from the HP Operations application when initiating the transaction. The target or destination port used to contact the HTTP server on the Intranet is defined in the URL address and is the Communication Broker port on the target node.If the target server is registered with the Communication Broker, the target port will always have the port number of the Communication Broker. This makes it easier when configuring firewalls. It can greatly reduce the number of target ports an administrator must configure at the firewall. For information on configuring HPOM with firewalls, refer to the *HPOM Firewall Concepts and Configuration Guide*.

Configuring HTTPS-Based Communication

HP applications may be customized for installation using configuration parameters. The communication broker configuration parameters are contained in the `bbc.ini` file located at:

`<OVDataDir>/conf/confpar/bbc.ini`

The parameters used for communication are described in the bbc.ini(4) file, which is described in the HP Operations Agent documentation.

The Communication Broker uses the namespace `bbc.cb`. An additional namespace, `bbc.cb.ports`, has been defined to specify the Communication Broker port number for all managed nodes. This enables different Communication Brokers to have different port numbers. This configuration takes precedence over the `SERVER_PORT` parameter defined in the namespace `bbc.cb`.

Note: A namespace is a unique URL, for example:

`www.anyco.com` or `abc.xyz`

Namespaces provide a simple method for qualifying element and attribute names used in Extensible Markup Language documents by associating them with namespaces identified by URL references.

The name/value pairs in the `bbc.cb.ports` namespace define the port numbers for the Communication Brokers within the network. The syntax of the name/value pairs is:

`NAME=<host>:<port>` or `NAME=<domain>:<port>`

Multiple host/port or domain/port combinations may be defined per line. Each is separated by a comma or semicolon.

A domain takes the form `*.domainname`. All entries for this domain will use the specified port. More specific entries take precedence. The name of the name/value pair is ignored, although the names must be unique within this namespace. The following are entry examples:

- `HP=jago.sales.hp.com:1383, *.sales.hp.com:1384; *.hp.com:1385`
- `SUN= *.sun.com:1500`

In this example, the Communication Broker running on the host `jago.sales.hp.com` will have the port number `1383`.

All other hosts within the domain `sales.hp.com` use the port number 1384. All other hosts within the domain `hp.com` use the port number 1385. Hosts in the domain `sun.com` use the port number 1500. All other hosts use the default port number 383.

Chapter 4: Implementing Message Policies

In this Chapter

This chapter explains how to implement message policies and distribute configurations in a HP Operations Manager (HPOM) environment.

Who Should Read this Chapter

This chapter is designed for HPOM *administrators*.

What this Chapter Does

This chapter explains the following to HPOM administrators:

- "Message Management" on the next page
- "Managing Message Source Policies" on page 129
- "Evaluating Message Sources " on page 135
- "Collecting Messages" on page 136
- "Processing Messages " on page 138
- "Filtering Messages with Conditions " on page 144
- "Strategies for Optimal Message Filtering" on page 162
- "Logging Messages " on page 180
- "Log File Messages " on page 183
- "HPOM Message Interface " on page 187
- "Messages from Threshold Monitors" on page 188
- "SNMP Traps and Events" on page 202
- "Filtering Internal HPOM Error Messages" on page 204
- "Event Correlation in HPOM" on page 204
- "Scheduled Outages" on page 223

- "Configuring Service Hours and Scheduled Outages" on page 224
- "Setting Custom Message Attributes Based on Message Selection Criteria " on page 224

Message Management

With HP Operations Manager, (HPOM), you can set up a centralized management point for your message sources. Because messages are generally intercepted at managed nodes, network traffic to the management server is reduced. By distributing message source information from the management server to specific managed nodes, you have only the necessary configuration on each node. Additions and changes to message source information are made on the management server once, and distributed only to those nodes requiring the information.

Centralizing Actions

From the management server, you can start automatic actions and operator-initiated actions on all systems, thereby minimizing or even eliminating operator intervention at remote sites.

Detecting Problems Early

When operators in your environment observe node activities with HPOM, they can detect problems early, and take corrective actions before problems become critical for end users.

Improving Productivity

You can also improve the productivity of operators by delegating simple, repetitive tasks to HPOM, by providing instructions to help operators solve more complex tasks, and by reducing the number of messages operators receive in the message browser. HPOM lets you match operator skill sets and responsibilities with the corresponding toolsets.

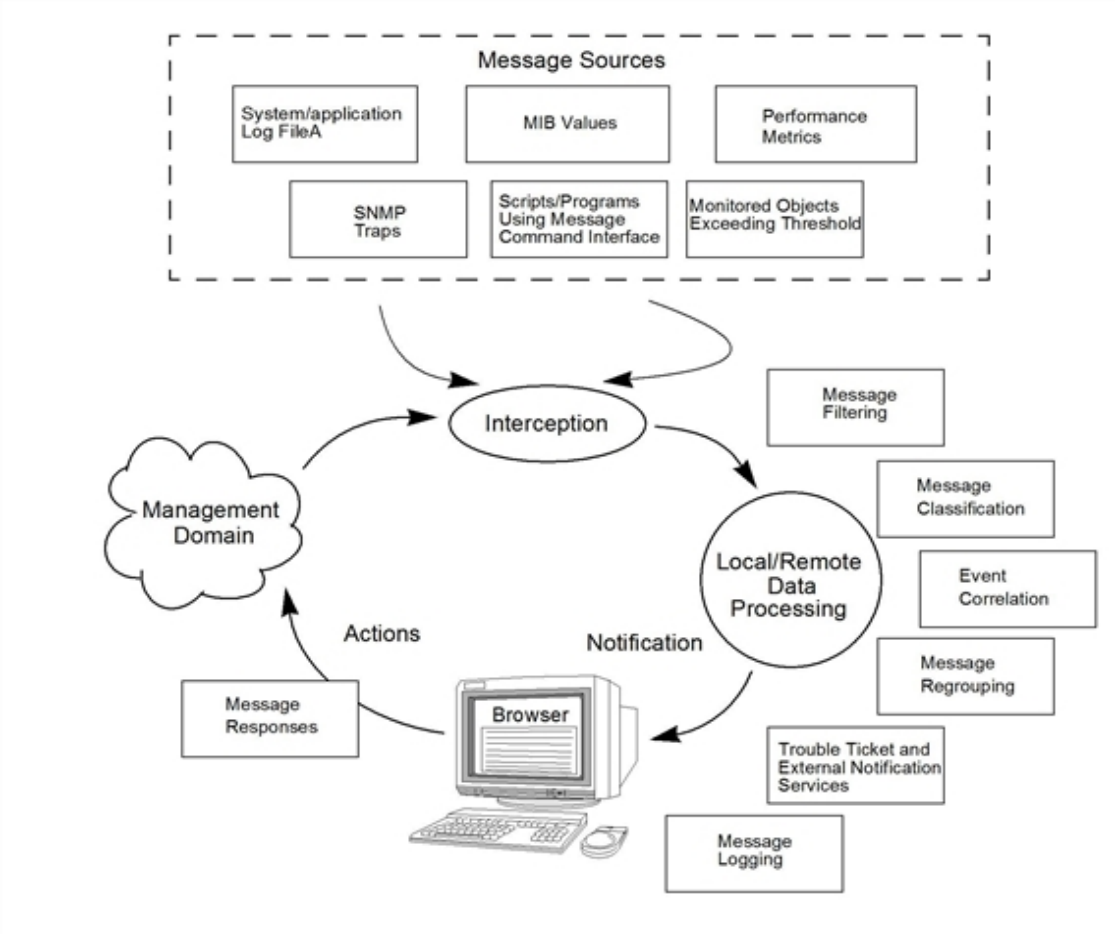
Distributing Policies

Policies ensure that you collect the same kind of information from sources throughout your environment. You distribute the policies to the managed nodes from which you want to collect information.

Consolidating Messages in the Browser

Figure 18 shows how HPOM intercepts, processes, and displays messages.

Figure 18: Consolidating Relevant Messages in the Browser



Managing Message Source Policies

The central element of message interception is the **message source policy**, which you set up on the management server. Message source policies specify the messages and values that are to be collected or monitored. They also specify the routine actions to be scheduled, the filters (conditions) that integrate or suppress messages, and the logging options to be used after interception.

Elements of a Message Source Policy

A message source policy is made up of the following elements:

- **Type of Message Source**

  Defines the sources from which you want to collect messages and assigns default attributes for all messages:

- Log files (`Logfile`)
- SNMP trap (`Trap`)
- HPOM message interface (`opcmsg(1|3)`)
- Threshold monitor (`Monitor`)
- Event correlation circuit (`EC`)
- Event correlation composer (`EC`)
- Scheduled action (`Schedule`)

- **Message Conditions**

  Filter messages that match a set of attributes into HPOM. Message conditions also define responses to received messages.
- **Suppress Conditions**

  Exclude messages from HPOM that exactly match a set of attributes.
- **Options**

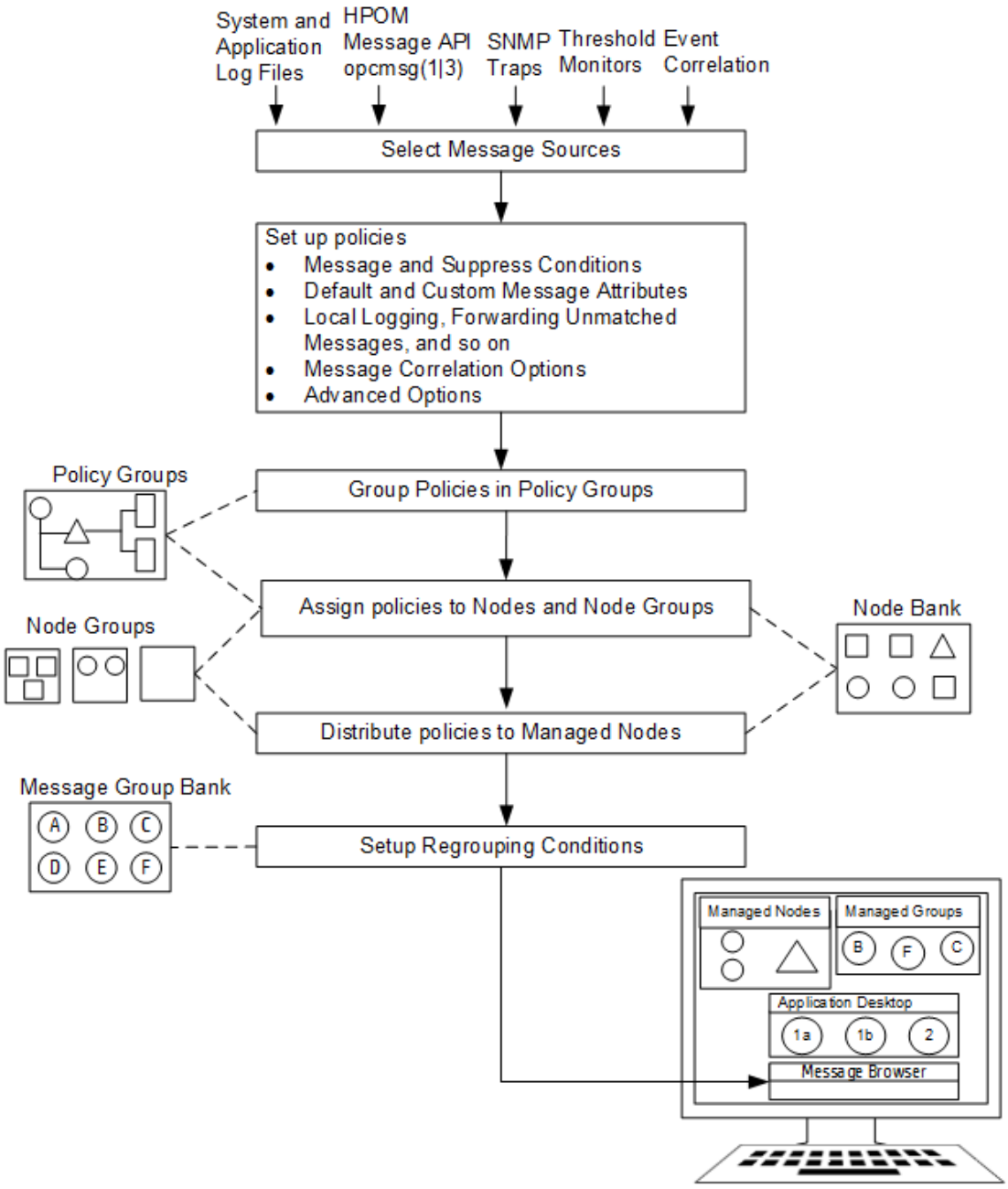
  Specify default logging of messages, and set forward unmatched message options.

Configuring Message Source Policies

Message source policies enable you to integrate messages from different message sources into HPOM. By configuring policies, you can determine whether a message is forwarded to the Java GUI message browser, with which attributes the message is displayed, and whether actions are to be performed.

Figure 19 shows the task flow from selecting message sources to setting up regroup conditions for message source policies.

Figure 19: Configuring Message Source Policies



Message Source Policies

The administrator can create, edit, and delete policies, as well as assign the policies to a policy group. Conditions are defined for each policy, and further options are specified.

For detailed information about setting up a message source policy, see the *HPOM Administrator's Reference*.

Caution: If you do not define any conditions for a message source policy, and if the `FORWARDUNMATCHED` keyword is present in the policy body, HPOM intercepts all messages from that message source. This interception of all messages could lead to a large number of unmatched messages reaching the message browser.

Creating Policies for Message Sources

HPOM enables you to create multiple policies for the same message source. Create your own policies and conditions for all message sources, rather than modifying the preconfigured policies.

Caution: When you upgrade HPOM to a higher version, all modified policies are lost.

Organizing Policy Groups

Policy groups are collections of policies or other policy groups. As the administrator, you can group policies to increase the performance of configuration and management tasks.

For example, you could group policies sharing the following characteristics:

- Common message source
- Common managed node platform

Advantages of Policy Groups

Organizing policies into policy groups has the following advantages:

- **Meaningful Overview**

  Policy groups let you organize your policies into logical units. For example, you might combine all policies relating to spool servers in one group. This grouping reduces the number of policies displayed in the policies list, and gives you a better overview of available policies.
- **Clear Hierarchy**

  You can place your policy groups in a policy group hierarchy. This hierarchy enables you to improve the organization of policies, making it easier for operators to concentrate on one type of policy when editing.
- **Simplified Assignment**

  Assigning policies to a managed node or a node group is straightforward. You can assign a specific policy group to a specific type of node. When adding a new node, you can assign all required groups, rather than collecting single policies for assignment. This ensures that all needed policies are selected.

Listing Policy Groups

You can list policy groups by using the `opcpolicy -list_groups` command. The operations with policy groups include creating and deleting policy groups, assigning policies to groups, and deassigning policies from groups, as well as listing all policy groups and their contents. More information can be obtained if you increase the verbosity level or use different values of the `level` parameter. For more information, see the *opcpolicy (1M)* manpage.

Creating Policy Groups

You can create and delete your policy groups, as well as assign and deassign policies to them. A policy can be assigned to more than one policy group. This multiple assignment capability provides you with the flexibility to form policy groups that meet the precise needs of your organization. To ensure system efficiency, HPOM ensures that the policy is distributed only once to the managed node.

When you create policy groups, make sure they simplify the policy assignment. For example, create a policy group for all policies that monitor database servers. When assigning the policies to your managed nodes, you can then assign the policy group “Database Monitoring” to the node group “Database Servers”. For a list of the default policy groups that HPOM provides for each supported agent platform, see the HP Operations Agent documentation.

It is important to remember that assigning a policy to a group that is a member of another group does not automatically assign the policy to both groups. For example, assigning a policy X to group /a does not automatically assign the same policy to group /b/a.

There are three different types of policy-to-policy-group assignments that relate to the policy version:

`FIX`	Exact version of the policy is assigned to the group and deployed to the node.
`LATEST`	Latest version of the policy existing at the time of deployment is deployed to the node.
`MINOR_TO_ LATEST`	Latest version of the policy existing at the time of deployment that has the same major version as the assigned one is deployed to the node.

Regrouping Messages

To reorganize messages into other message groups, you can establish regroup conditions. If HPOM Event Correlation Services (ECS) is installed, you can also consolidate similar messages into fewer and more meaningful messages by configuring event correlation policies.

Assigning Policies

After you have configured policies and policy groups, you need to determine which node or node groups should receive the new policies. You can assign policies to nodes or node groups by using the `opcnode` command-line tool. Or you can assign policy groups to nodes or node groups where the message interception should be performed. Then you can distribute the new configuration.

Assigning Policies to Managed Nodes

You can assign policies and policy groups to managed nodes and node groups by using the `opcnode` command-line tool. The process of assigning a policy group to a node is the same as the process you use to assign a single policy. All nodes within a node group automatically inherit the policies and policy groups assigned to the node group. This inheritance simplifies assigning policies to new nodes. For example, you can assign a policy group to a node group by using the following command:

`opcnode -assign_pol_group pol_group=<policy_group_name> group_name=<node_group_name>`

In this command, *`<policy_group_name>`* is the name of the specified policy group and *`<node_group_name>`* is the name of the node group.

For more information, see the *opcnode(1M)* manpage.

Note: HPOM assigns and distributes policies individually. If only one policy in a policy group is changed, HPOM redistributes only the changed policy. If a policy is assigned several times to the same node through different policy groups, the policy is distributed and handled only once on the managed node. While policies are being modified, they are locked and cannot be distributed. You can verify the status of policies, and their assignment to managed nodes, by generating a report. For details about available reports, see the *HPOM Administrator's Reference*.

Distributing Assigned Policies

After you have defined a new message source policy and assigned it to the managed nodes, you must distribute it to the managed nodes. To find out how, see "Updating the HPOM Configuration" on page 75.

Note: If you delete a policy, or if you remove a policy assignment for a specific node, you must distribute the new configuration to the affected managed nodes. Otherwise, your changes will not become active.

If you want to temporarily disable a policy on a managed node, use the `opcpolicy` command-line tool. For details, see the *opcpolicy(1M)* manpage.

Distributing Message Source Policies

After defining a message source policy, you distribute it to the managed nodes, where it intercepts messages and monitor values. You can distribute message source policies by using the following command:

```
opcragt -distrib -policies
```

For more information, see the *opcragt(1M)* manpage.

Evaluating Message Sources

The first step in implementing a message policy is to review existing message sources.

Where to Look for Messages

Evaluate the following message sources:

- Application and system log files
- Applications using the HPOM message API `opcmsg(3)`
- Applications using the HPOM command interface `opcmsg(1)`
- Monitored objects
- Performance metrics
- Monitored SNMP MIB values
- Applications sending SNMP traps

How to Evaluate Messages

Evaluate the messages by using the following criteria:

- Effect of events on end user
- Level of severity
- Frequency of occurrence
- File format in terms of readability (that is, binary or plain text format)
- Language and character set
- Network traffic and performance

Determine which messages require operator attention and which do not. Many messages are not significant because they do not affect system performance or the ability of users to perform their daily tasks. Other messages represent potential or current problems. These problem messages tell you that, without preventive counter-measures, a problem will occur or could occur again.

Evaluating the Severity of Messages

Evaluate the severity of each message. Many messages contain a severity level as part of the message. Decide if these severity levels reflect the true significance of the message in your environment. An object can issue a message with a severity of critical, but the message might not represent a critical situation within your environment.

Starting With Message Catalogs

If an application provides a message catalog, its contents can be used as a starting point when considering possible messages.

Collecting Messages

A message is a structured piece of information about the status of an **object** in your operating environment. This object can be any component of your operating environment, from the operating system to an application, or even a peripheral device.

Creating Message Status

A message is created as a result of an event or change of status. You can specify the severity and general characteristics of an event.

Depending on which severity level you assign to the message, the message is either intercepted by HPOM or filtered out of HPOM. If the message is intercepted by HPOM, it is then processed and displayed in a Java GUI operator's browser window.

Note: HPOM collects messages from the various message sources at regular intervals in the message source policies. For example, you can define polling intervals for the log file encapsulator or the HPOM monitor agent. When you define intervals, make sure the interval is not too small. If the interval is to small, it can cause unnecessary system overload. You may also consider adapting the HPOM default message source policies.

Intercepting Messages

HPOM intercepts messages from the following sources:

- **Log Files**

  Application and system log files.
- **Applications Integrated into HPOM**

  Applications that have been integrated into HPOM send messages through the `opcmsg (3)` application programming interface (API) or the `opcmsg(1)` command-line tool. You can integrate your own applications by writing a program that sends messages through `opcmsg(3)` or `opcmsg(1)`.
- **SNMP Traps**

  Applications and network devices sending SNMP traps.
- **Threshold Monitors**
  - *Application and System Values*

    Many application or system values can be compared with expected values.
  - *Database Values*

    Use the SQL database language and database administration tool to monitor specific values (for example, table sizes and number of locks). Compare these values with expected values.
  - *Processes*

    Use scripts to check if important processes are operating, for example, daemons. Check the process values, for example, the number of processes running.
  - *Files and File Systems*

    Check the existence and size of important files or file systems. The script can return the used or available disk space, which is checked against a predefined limit.
  - *Performance Metrics*

    The embedded performance component collects performance counter and instance data from the operating system.
  - *SNMP MIB Values*

    Check dynamic Management Information Base (MIB), parameters. These parameters are set and updated by applications, which may or may not be outside of HPOM. HPOM checks the current values against the configured thresholds using the SNMP API.
- **Scheduled Action Messages**

  Schedule automatic actions for routine tasks (for example, for policy distribution). If so configured, HPOM generates messages informing you about the success or failure of your scheduled action.

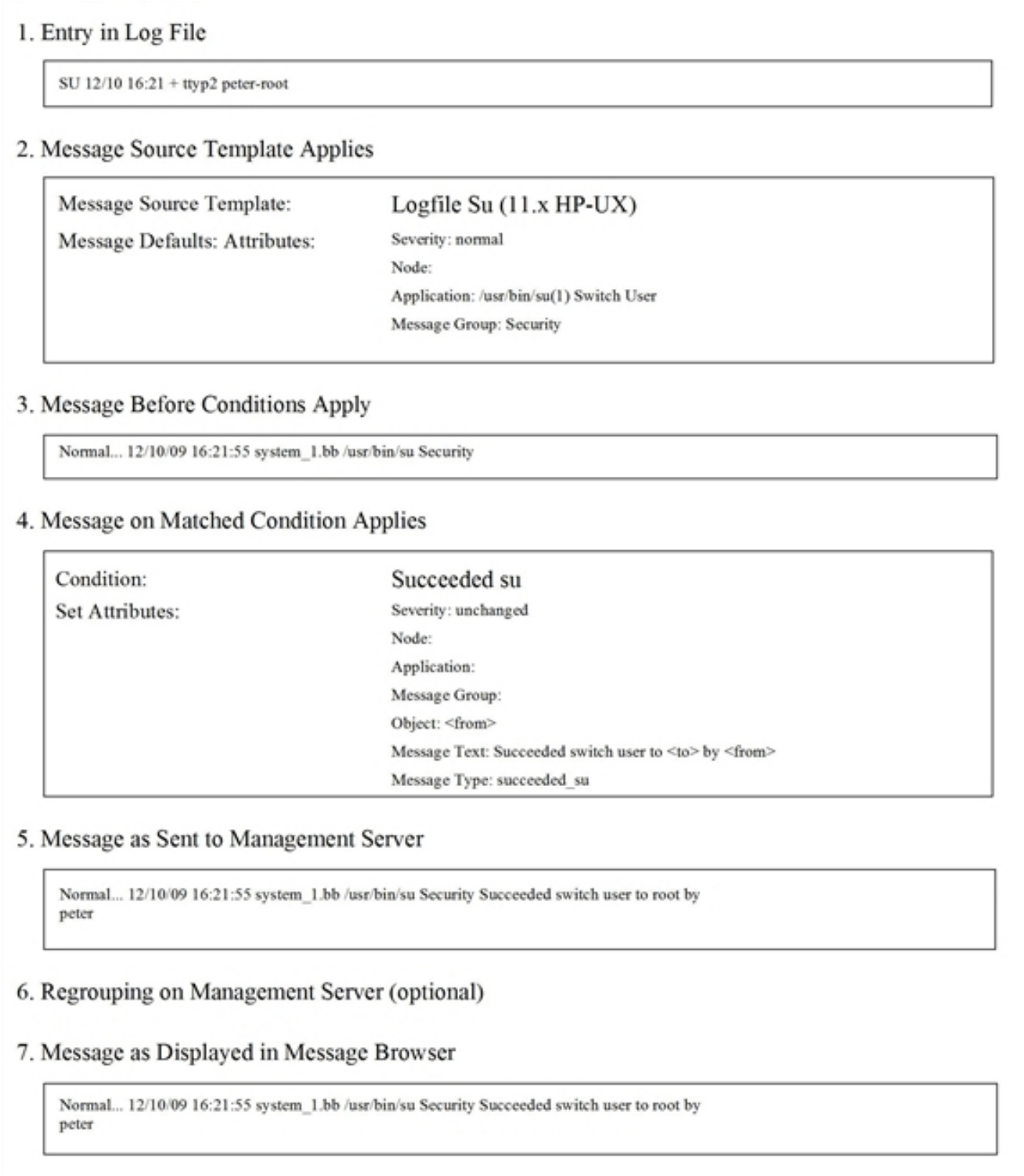
For more information about configuring scheduled action policies, see the *HPOM Administrator's Reference*.

Processing Messages

After you have configured policies to intercept messages from their sources, you must establish conditions to filter the messages. HPOM lets you set up **conditions** to filter messages into HPOM or to exclude messages from being forwarded to the management server.

Figure 20 shows the path of a message through policy filters, condition filters, and regroup conditions before it reaches the browser.

Figure 20: Resolving Message Attributes



How Messages Are Processed by Policies

You can use message source policies to set global defaults for message attributes, message correlation options, pattern-matching options, and message output options.

Setting Message Defaults

A message source policy assigns the following default settings to a message:

- **Message Attributes**

  Message attributes are characteristics that the HPOM administrator can use to classify a message when it is received by the management server. Message attributes are, for example, the severity of a message, the node where the message is generated, the application or object related to the event, and the message group for the message. These default attributes can be set in the message source policies, but are overwritten by values set in message conditions. HPOM displays message attributes in the Java GUI browser.

- **Custom Message Attributes**

  Custom message attributes allow you to extend HPOM messages by adding additional information to the message, for example, the customer name, the type of Service Level Agreement, a device type, and so on.

  HPOM displays custom message attributes in the Java GUI browser where the operator can sort and filter messages based on these attributes.

  Use the `opccmachg` command-line tool to assign attributes of your choice to a message. For usage details, see the *opccmachg(1m)* manpage.

  Custom message attributes can only be set for message conditions for log file, HPOM interface, threshold monitor policies, SNMP Trap Interceptor (`trapi`) and Scheduled tasks. For more information on custom message attributes, see the HP Operations Agent documentation.

- **Message Correlation Options**

  You can assign a message key to the message, decide which messages should be automatically acknowledged by this message key (state-based browser), and choose how HPOM suppresses duplicate messages. The assigning of message keys prevents identical messages from filling up the Java GUI message browser. Use keyword `SUPP_DUPL_IDENT` to suppress messages matching the message key.

  For the Duplicate Message Suppression method, you can do the following:

  - Specify an interval of time over which HPOM suppresses duplicate messages. HPOM retransmits the messages when this time interval elapses.

- Specify a threshold for a duplicate message counter. HPOM increments the counter until it crosses the threshold, and then allows the transmission of the duplicate message.

For more information, see "Policy Body Grammar" on page 253.

- **Pattern-matching Options**

  You can specify field separators and case-sensitive checks used by policies when scanning messages. Use keywords `ICASE` and `SEPARATORS` to define case-sensitive checks and field separators respectively. For more information, see "Policy Body Grammar" on page 253.

- **Output Options for a Message Stream Interface**

  You can choose whether HPOM should output messages to an external message stream interface, and if so, how HPOM should transfer the messages.

  Use the following keywords to define defaults:

`MPI_SV_COPY_MSG`	Send the message to MSI as well as the server processes.
`MPI_SV_DIVERT_MSG`	Divert the message to MSI and do not send it to server processes.
`MPI_SV_NO_OUTPUT`	Do not send message to MSI (default).

These default attributes are set globally at the policy level, but can be overridden by a message condition. For more information about policy body syntax, see "Policy Body Grammar" on page 253.

Configuring Multiple Policies

HPOM allows the administrator to configure multiple policies with different message and suppress conditions for each message source. When an event occurs, it is filtered in parallel by all policies assigned to that managed node. As soon as a condition applies, the message is handled according to the options specified in the policy. It is therefore important to understand how messages are filtered by HPOM to avoid filling the browser with unimportant messages or losing important messages.

Processing Multiple Policies at the Same Time

HPOM is capable of handling, in parallel, several policies of the same type for one node. In this process, no priority is given to any policy, rather each policy is handled as an independent entity. A message that matches the `suppress` or `suppress unmatched` condition in one policy will be suppressed only for the processing in that policy. However, a message could still match a message condition in another policy and create an HPOM message for the responsible operator. For more information about how to improve performance in a multiple-policy configuration, see "Optimizing Performance" on page 163.

Figure 21 shows how messages are processed in parallel by policies:

- **Filtering Messages**

  Events create messages that are intercepted by HPOM, and filtered by the message source policies.

- **Assigning Default Settings**

  Policies assign default settings to the message.

- **Checking Message Conditions**

  The message is checked against a list of conditions. The first matched condition determines further processing.

- **Forwarding Messages**

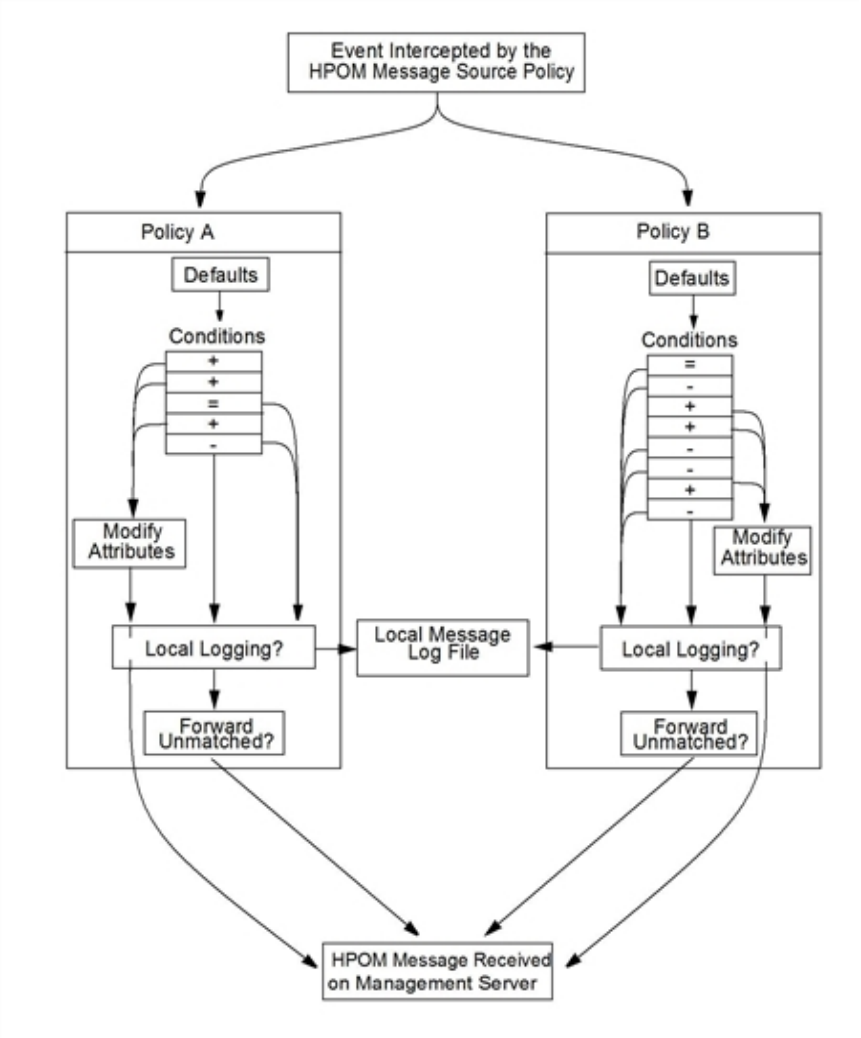
  If the message does not match any condition, and the Forward Unmatched attribute has been set in the policy, the unmatched message (containing the default values of the policy) is forwarded.

- **Logging Messages**

  If you have configured local logging or log-only for unmatched messages, HPOM logs the message accordingly.

One event can generate several HPOM messages as each policy is configured differently and reacts to the problem in its unique way.

Figure 21: Messages Filtering Through Multiple Policies



Forwarding Unmatched Messages

Using the `FORWARDUNMATCHED` keyword in multiple policies can lead to you receiving multiple messages from a single event. Each policy with the `FORWARDUNMATCHED` keyword creates a message with the default values of the policy.

Application-specific policies and generic policies handle multiple messages differently:

- **Application-specific Policies**

  Use the `SUPP_UNM_CONDITIONS` keyword to receive relevant messages only.

- **Generic Policies**

  Use the `FORWARDUNMATCHED` keyword to receive also unmatched messages beside relevant ones.

Only policies of the following type do *not* generate multiple messages from a single event:

- Logfile Entry policies
- SNMP Trap policies
- HPOM Interface Message policies

They process all assigned policies before forwarding an unmatched message to the management server. If the message is matched by a suppress condition, but Forward Unmatched is set in another policy, the message is suppressed. Suppress Unmatched conditions only suppress a message for that policy but forward messages that are unmatched by other policies to the management server.

Configuring Your Own Application-Specific Policies

When you modify preconfigured HPOM policies and conditions, make sure they are saved under the different version.

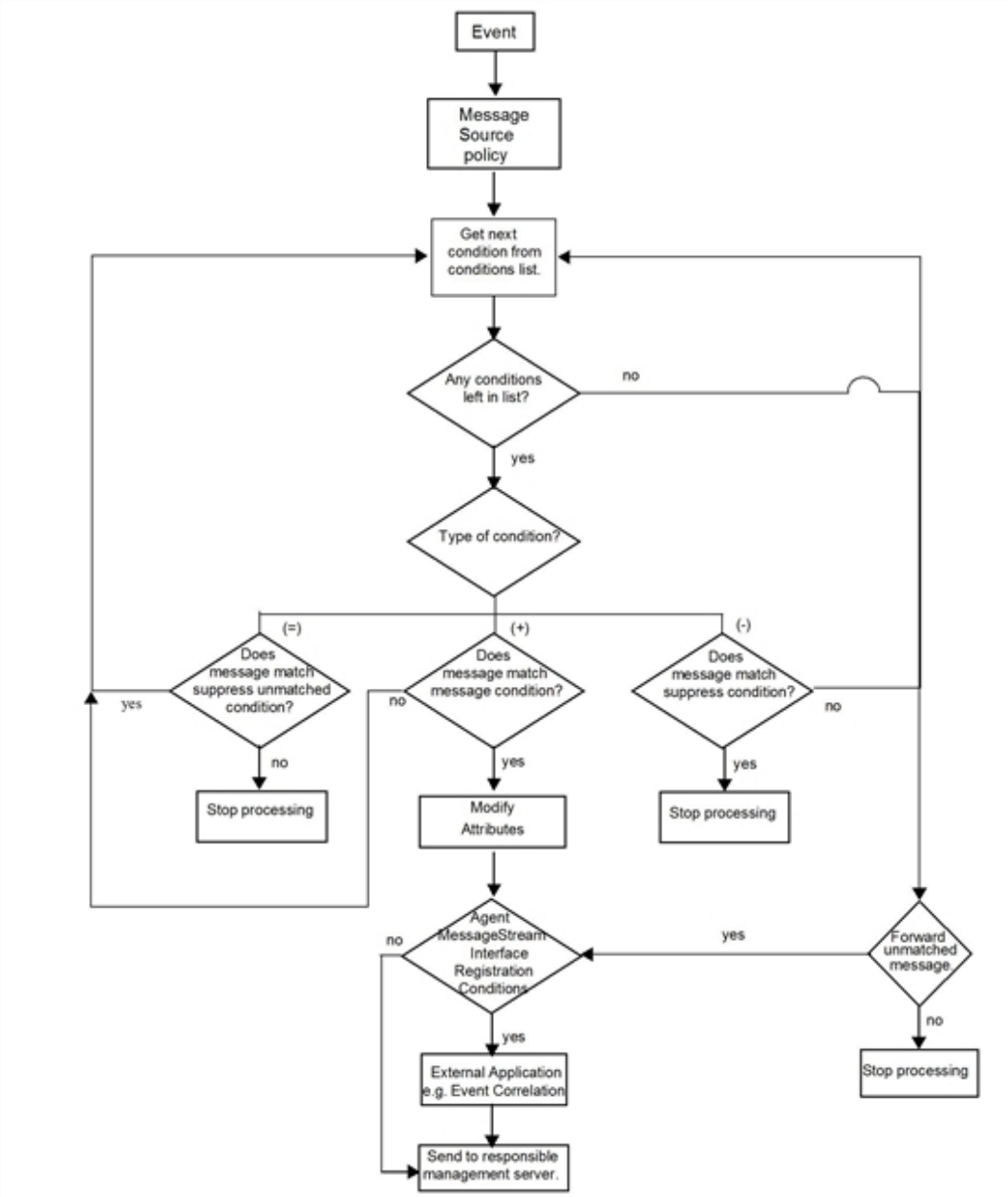
Filtering Messages with Conditions

The cornerstone of HPOM is its message processing mechanism. By controlling the exact messages from a source that you want operators to receive, **conditions** help reduce the amount of data and provide a common format for diverse message sources.

Filtering Message Sources

Figure 22 shows the process through which messages are checked against conditions on the agent, and how matched and unmatched messages are handled. This process is called “message source filtering”.

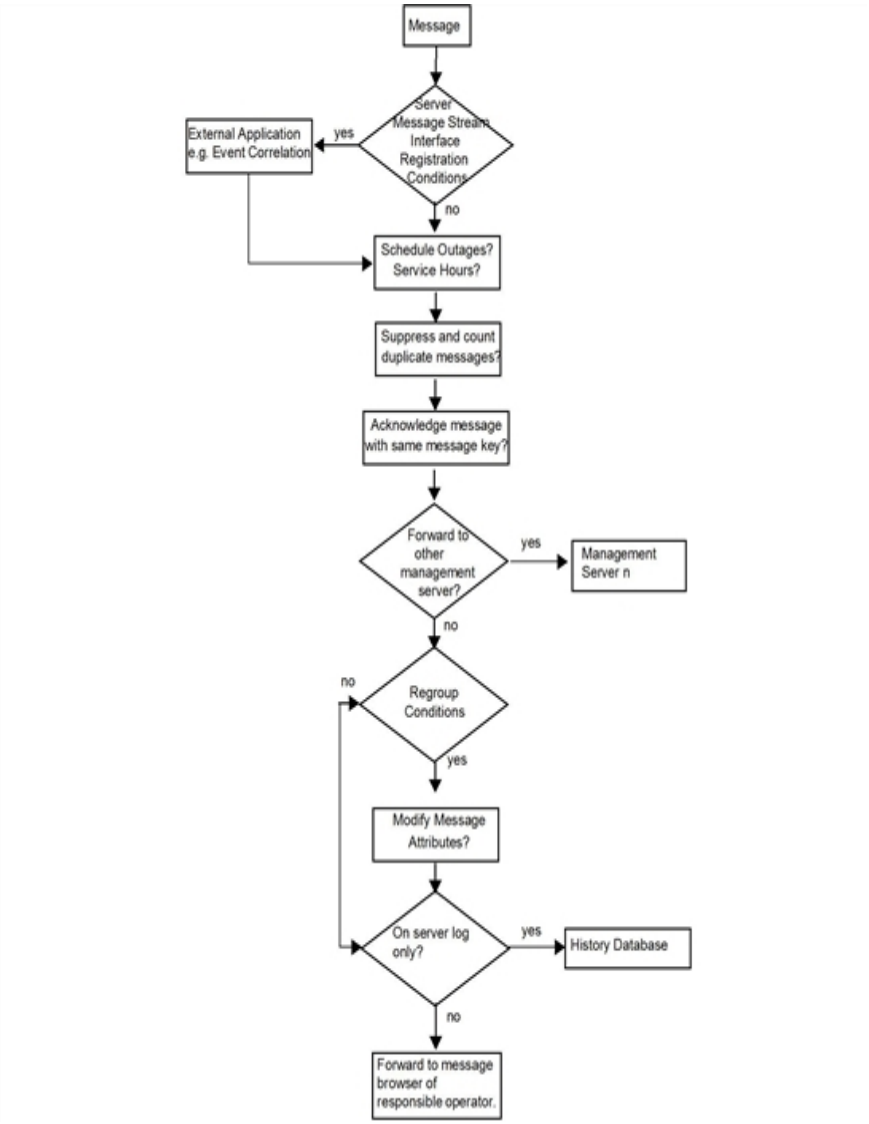
Figure 22: Flow of Messages Passing Through HPOM Filters on the Agent



Processing Messages on the Management Server

Figure 23 shows how messages are processed on the management server before they arrive in the browser of the responsible operator.

Figure 23: Flow of Messages Passing Through HPOM Filters on the Server



To Set Up Message Conditions

When setting up message conditions, use the policy body syntax described in "Policy Body Grammar" on page 253.

To set up message conditions, follow these steps:

1. **Define Match Conditions**

   Define a matching pattern called the message condition or suppress condition.
2. **Test Pattern Matching**

   Test to make sure that the pattern matching of one condition works as expected.

3. **Set Up Message Correlation Options**

   Set up message correlation options to automatically acknowledge messages with a specific message key, and to stop frequently repeated messages from cluttering up the Java GUI message browser.

4. **Configure Operator-initiated Actions**

   Configure operator-initiated actions so that, every time a selected message is matched, HPOM allows a selected operator to run a script or program that you have configured.

5. **Configure Automatic Actions**

   Configure automatic actions so that, every time a message is matched, HPOM runs a script or program automatically.

6. **Configure Messages**

   Configure messages to be output to an external notification or to a trouble ticket service.

7. **Define Message Attributes**

   Define the attributes of the message to be displayed in the Java GUI message browser. These attributes are not necessarily the same as those for the original text matched from the message source.

8. **Define Custom Message Attributes**

   Define your own message attributes of the message to be displayed in the Java GUI message browser to provide operators with more relevant information about the message.

9. **Write Instructions**

   Write instructions to accompany the message displayed in the Java GUI message browser.

For more information about setting up message conditions, see "Policy Body Grammar" on page 253 and *HPOM Administrator's Reference*.

If you do not define any filters for a message source, all messages from that source are brought into HPOM for processing, provided you have chosen to forward unmatched messages to the management server.

Message and Suppress Conditions

Conditions consist of various attributes (for example, node name, application name, message key, or a text or object pattern) that can be compared with the event. HPOM compares each incoming message with the message and suppress conditions in the order they are listed in the policy body.

You can set up as many message, suppress, and suppress unmatched conditions as you need to filter messages from a single message source policy either into or away from HPOM.

Conditions That Can Be Applied to Events

The following conditions can be applied to events on the managed node:

- **Message on Matched Condition**

  If a message matches *all* attributes set for a message condition, it is brought into HPOM for further processing.

  Message conditions enable you to set **message attributes**, and forward messages from the managed nodes to the HP Operations management server, where they can be assigned to specific operators.

- **Suppress Matched Condition**

  If a message matches *all* attributes for a suppress condition, it is excluded from HPOM. The message is not processed further.

  Suppress conditions enable you to reduce the number of messages that HPOM must process and that are displayed in the Java GUI message browser.

- **Suppress Unmatched Condition**

  If a message does *not* match the attributes for a suppress unmatched condition, it is excluded from HPOM. The message is not processed further.

  If a message matches *all* attributes for a suppress unmatched condition, it is processed further by the succeeding conditions in the conditions list.

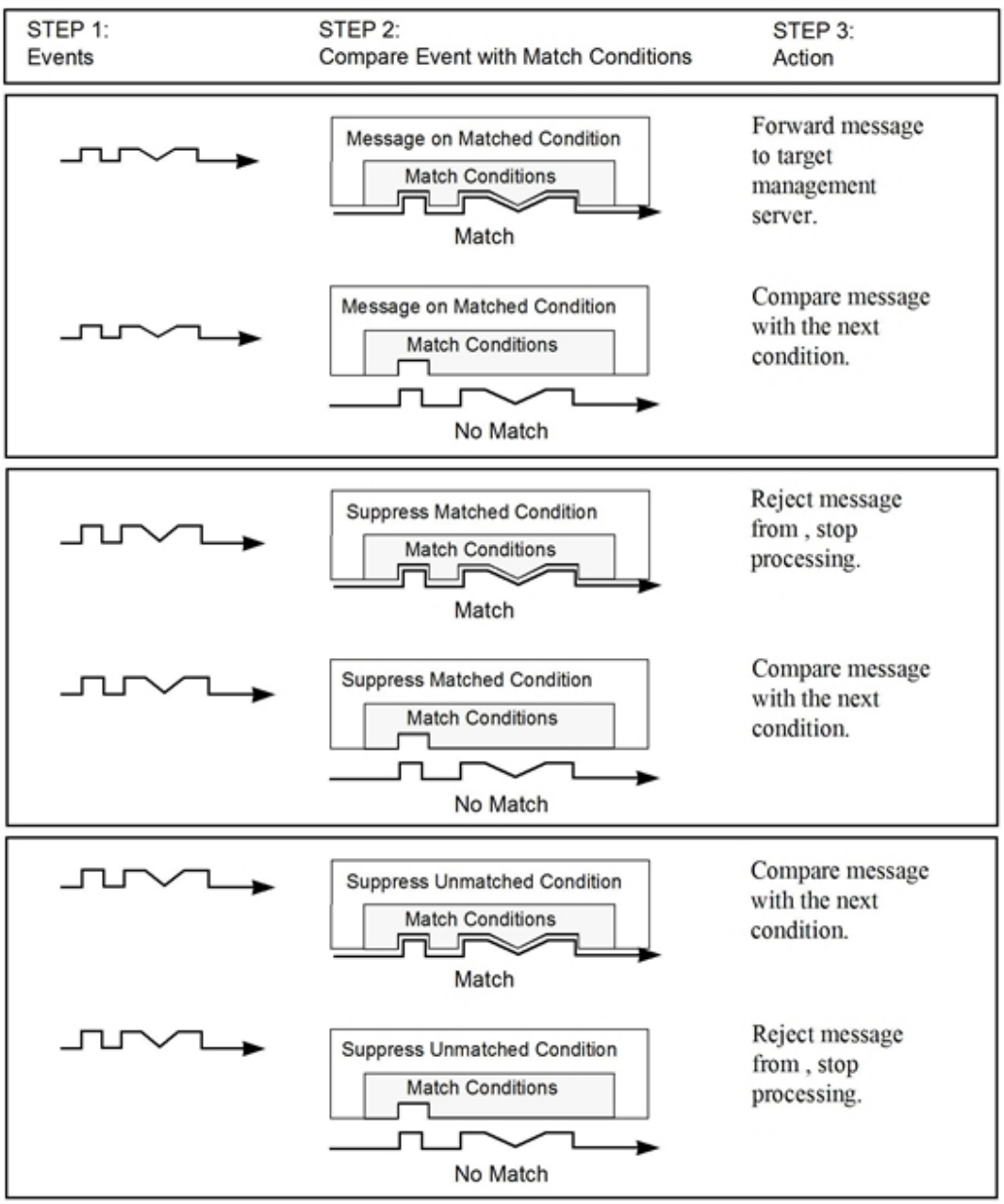
  Suppressing unmatched conditions enables you to improve HPOM performance by filtering out (at the condition level) all messages that are irrelevant to the policy, thus reducing the number of messages processed by the policy condition list.

  When the `FORWARDUNMATCHED` keyword is used in a policy body, the set of unmatched messages that are forwarded to the server are restricted to only those that directly relate to the policy.

Comparing Incoming Messages with Match Conditions

Figure 24 shows how incoming messages are compared with the match conditions specified for message, suppress, and suppress unmatched conditions. For details about how to set up message conditions, see "Filtering Messages with Conditions " on page 144.

Figure 24: Using Conditions to Filter Messages



Note: Though both refer to unmatched messages, the `SUPP_UNM_CONDITIONS` keyword filters messages at the condition level while the `FORWARDUNMATCHED` keyword filters messages on the policy level.

Pattern Matching in Messages

HPOM provides a powerful pattern-matching language that reduces the number of conditions you must enter to a minimum. Selected, dynamic parts of messages can be extracted, assigned to variables, and used as parameters to build new message text or to set other attributes. These parameters can also be used for automatic and operator-initiated action commands. For a full list of HPOM and SNMP variables, see the *HPOM Administrator's Reference*.

Pattern Matching with Mathematical Operators

In most cases, pattern matching involves simply scanning for a specific string in a message. However, a number of mathematical operators are available to enhance the precision of the search. For example, if you type `ERROR` along with the `TEXT` keyword in `MSGCONDITIONS` block of the policy body, only message text containing the string "ERROR" (in any position) is matched.

Similarly, if you want to match text that does not contain a specific string (for example, "WARNING"), you might enter the following:

```
<![WARNING]>
```

In this example, you use the **not operator (!)**, together with the angle brackets that must enclose all operators, and the square brackets that isolate sub-patterns.

Because messages that match a suppress condition are excluded from HPOM, you do not need to reformat messages or specify actions for matching messages. For an illustration of the message flow through HPOM, see Figure 22.

Pattern Matching Without Case-Sensitivity

If you used the `ICASE` keyword in your policy body, any combination of uppercase and lowercase characters composing the word, "warning", can be matched. For more information about policy body syntax, see "Policy Body Grammar" on page 253.

Examples of Pattern-Matching Conditions

Here are a few examples of the many conditions you can use in the HPOM pattern-matching language:

- `Error`

  Recognizes any message containing the keyword `Error` at any place in the message. This condition is case-sensitive by default.

- `panic`

When case-sensitive mode is not set, this condition matches all messages containing `panic`, `Panic`, or `PANIC`.

- `logon|logoff`

  Uses the **OR operator (|)** to recognize any message containing the keyword `logon` or `logoff`.

- `^getty:<*.msg> errno<*><#.errnum>$`

  Matches messages such as:

  ```
  getty: cannot open ttyxx errno: 6
  getty: can’t open ttyop3; errno 16
  ```

  In the first example, the `cannot open ttyxx` string is assigned to the **msg** variable. The digit 6 is assigned to the **errnum** variable. The anchoring symbol is used to specify that the digit 6 will be matched only if it is at the end of the line.

- `^errno[ |=]<#.errnum> <*.errtext>`

  Matches messages such as:

  ```
  errno 6 - no such device or address
  errno=12 not enough core.
  ```

  The blank space before the `OR` operator is critical. The expression in square brackets matches either this blank space or the equal sign (=). The blank space between <#.**errnum**> and <*.**errtext**> is used as a delimiter. Although not strictly required for assignments to the variables shown here, this blank space helps improve performance.

- `^hugo:<*>:<*.uid>:`

  Matches any `/etc/passwd` entry for user `hugo` and returns the user id to variable **uid**. The colon (:) in the middle of the pattern is used to delimit the string passed to **uid** from the preceding string. The colon at the end of the pattern is used to delimit the string passed to **uid** from the succeeding group ID in the input pattern. The colon is necessary not only to enhance performance, but also as a logical string separator.

- `^Warning:<*.text>on node<@.node>$`

  Matches any message, such as `Warning: too many users on node hpbbx`, and assigns `too many users` to the variable **text** and `hpbbx` to the variable **node**.

Details of Pattern-Matching Expressions

Here are details about the expressions you can use in the HPOM pattern-matching language:

- **Standard Characters**

Ordinary characters are expressions that represent themselves. Any character of the supported character set may be used. However, if any of the following special characters are used, they must be prefaced with a backslash (\), which masks their usual function:

```
[ ] < > | ^ $
```

If a caret (^) and a dollar sign ($) are not used as anchoring characters (that is, not as first or last characters), they are considered standard characters and, as a consequence, do not need to be masked.

- **Expression Anchoring Characters (^ and $)**

  If a caret (^) is used as the first character of the pattern, only expressions discovered at the beginning of lines are matched. For example, `^ab` matches the string `ab` in the line `abcde`, but not in the line `xabcde`.

  If the dollar sign ($) is used as the last character of a pattern, only expressions at the end of lines are matched. For example, `de$` matches `de` in the line `abcde`, but not in the line `abcdex`.

- **Expressions Matching Multiple Characters**

  Patterns used to match strings consisting of an arbitrary number of characters require one or more of the following expressions:

<*>	Matches any string of zero or more arbitrary characters (including separators).
<n*>	Matches a string of *n* arbitrary characters (including separators).
<#>	Matches a sequence of one or more digits.
<n#>	Matches a number composed of *n* digits.
<_>	Matches a sequence of one or more separator characters.
<n_>	Matches a string of *n* separators.
<@>	Matches any string that contains no separator characters. In other words, it matches a sequence of one or more non-separators. This pattern can be used for matching words.

  Separator characters are configurable for each condition. By default, separators are the space and the tab characters.

- **Square Bracket Expressions**

  The square brackets ([]) are used as a delimiter to group expressions. To increase performance, square brackets should be avoided wherever they are superfluous.

  In the following pattern, all square brackets are unnecessary, the string `abcdefgh` is equivalent:

  ```
  ab[cd[ef]gh]
  ```

Bracketed expressions are used frequently with the **OR operator**, the **NOT operator**, and when using **sub-patterns** to assign strings to variables.

- **OR (|) Operator**

  Two expressions separated by the special character vertical bar (|) matches a string that is matched by either expression.

  For example, the following pattern matches the string `abd` and the string `cd`:

  ```
  [ab|c]d
  ```

- **NOT (!) Operator**

  The **NOT operator (!)** must be used with delimiting square brackets.

  For example, the following pattern matches all text which does not contain the string “WARNING”:

  ```
  <![WARNING]>
  ```

  The **NOT operator** may also be used with complex sub-patterns:

  ```
  SU <*> + <@.tty> <![root|[user[1|2]]].from>-<*.to>
  ```

  This pattern makes it possible to generate a switch user message for anyone who is not `user1`, `user2`, or `root`.

  Therefore, the following string would be matched:

  ```
  SU 03/25 08:14 + ttyp2 user11-root
  ```

  However, the following line would not be matched because it contains an entry concerning `user2`:

  ```
  SU 09/25 08:14 + ttyp2 user2-root
  ```

  If the sub-pattern including the **not operator** does not find a match, the **not operator** behaves like a `<*>`. It matches zero or more arbitrary characters. For this reason, there is a difference between the UNIX `[!123]` expression, and the corresponding HPOM pattern-matching expression `<![1|2|3]>`. The HPOM expression matches any character or any number of characters, except 1, 2, or 3. The UNIX operator matches any *one* character, except 1, 2, or 3.

- **Mask (\) Operator**

  The backslash (\) is used to mask the special function of the following characters:

  ```
  [ ] < > | ^ $
  ```

  A special character preceded by a backslash (\) results in an expression that matches the special character itself.

  Because a caret (^) and a dollar sign ($) have a special meaning only when placed at the beginning and end of a pattern, you do not need to mask them when they are used within the pattern (that is, not at beginning or end of the pattern).

  The only exception to this rule is the tab character, which is specified by entering `\t` into the pattern string.

The OR operator (|) can be used in the following fields of the match condition:

- Node
- Application
- Message group
- Object

Numeric Range Operators

The basic pattern for constructing complex expressions with these operators is:

```
<number--operator--[sub-pattern]--operator--number>
```

The sub-pattern can be a simple numeric operator, for instance `<#>` or `<2#>`. Such a simple operator does not require delimiting brackets. Alternatively, it may be a complex sub-pattern, using delimiting brackets:

```
<120 -gt [<#>1] -gt 20>
```

It is also possible to construct a pattern by using only one operator:

```
Error <<#> -eq 1004>
```

The 6 Numeric Range Operators:

`-le`	Less than or equal to
`-lt`	Less than
`-ge`	Greater than or equal to
`-gt`	Greater than
`-eq`	Equal to
`-ne`	Not equal to

Less Than or Equal To (-le) Operator

Example of use:

```
<<#> -le 45>
```

This pattern matches all messages containing a number that is less than or equal to 45. For example, the following message would be matched:

```
ATTENTION: Error 40 has occurred
```

Note that the number 45 in the pattern is a true numeric value and not a string. Numbers higher than 45, for instance, 4545 will not be matched even if they contain the combination, 45.

Less Than (-lt) Operator

Example of use:

```
<15 -lt <2#> -le 87>
```

This pattern matches any message in which the first two digits of a number are within the range 16-87. For instance, the message:

`Error Message 3299` would be matched.
The string: `Error Message 9932` would not be matched.

Greater Than or Equal To (-ge) Operator

Example of use:

```
^ERROR_<57 -ge <#.err>>
```

This pattern matches any text starting with the `ERROR_` string immediately followed by a number less than or equal to 57. For example, the following message would be matched:

```
ERROR_34: processing stopped
```

The string 34 would be assigned to the variable, `err`.

Note the use of the caret (^) expression anchor.

Greater Than (-gt) Operator

Example of use:

```
<120 -gt [<#>1] -gt 20>
```

Matches all numbers between 21 and 119 that have 1 as their last digit. For instance, messages containing the following numbers would be matched: 21, 31, 41...101... 111, and so on.

Second example:

```
Temperature <*> <@.plant>: <<#> -gt 100> F$
```

This pattern matches strings such as: ``Actual Temperature in Building A: 128 F''. The letter A would be assigned to the variable, plant. Note the use of the $ expression anchor. A “greater than” operator is also referred to as an Open Interval. Using “greater than equal to” operators creates a Closed Interval.

Equal To (-eq) Operator

Example of use:

```
Error <<#> -eq 1004>
```

This pattern matches any message containing the string `Error` followed by the sequence of digits, `1004`. For example, the following message would be matched by this pattern:

```
Warning: Error 1004 has occurred
```

However, `Error 10041` would not be matched by this pattern.

Not Equal To (-ne) Operator

Example of use:

```
WARNING <<#> -ne 107>
```

This pattern matches any message containing the string `WARNING` followed by a blank and any sequence of one or more digits, except 107. For example, the following message would be matched:

```
Application Enterprise (94/12/45 14:03): WARNING 3877
```

To Insert Expression Symbols in Pattern-matching Expressions

Any time you work with pattern matching, you can use the left and right mouse buttons to insert expression symbols.

To insert expression symbols, follow these steps:

1. Select the text you want to replace with an expression.
2. Right-click the selected text for a list of replacement symbol choices.
3. Select the symbol from the list.

Note that you *cannot* use this feature to supply variable names, which must be typed into the expression individually.

Variables and Parameters of Pattern-matching Expressions

Any matched string can be assigned to a variable, which can then be used to recompose messages or be used as a parameter for action calls. To define a parameter, add *.parametername* before the closing bracket. The `^errno: <#.number> - <*.error_text>` pattern matches a message such as the following:

```
errno: 125 - device does not exist
```

and assigns `125` to **number** and `device does not exist` to **error_text**.

Variable names may only contain alphanumeric characters as well as underscores (_) and hyphens (-). The following syntax rules apply:

```
(Letter | '_'){ Letter | Digit | '_' | '-' }
```

In the syntax above, `Letter` allows letters and ideographic characters from all alphabets, and `Digit` allows digit characters from all alphabets.

Rules Used by HPOM to Assign Strings to Variables

In matching the pattern `<*.var1><*.var2>` against the string `abcdef`, it is not immediately clear which substring of the input string will be assigned to each variable. For example, it is possible to assign an empty string to var1 and the whole input string to **var2**, as well as assigning `a` to **var1** and `bcdef` to **var2**, and so on.

The pattern-matching algorithm always scans both the input line and the pattern definition (including alternative expressions) from left to right. Expressions such as `<*>` are assigned as few characters as possible. In contrast, expressions such as `<#>`, `<@>`, and `<_>` are assigned as many characters as possible. The variable will therefore be assigned an empty string in the above example. For example, to match the `this is error 100: big bug` input string, use the following pattern:

```
error<#.errnumber>:<*.errtext>
```

In this example:

- `100` is assigned to **errnumber**.
- `big bug` is assigned to **errtext**.

For performance and pattern readability purposes, you can specify a delimiting substring between two expressions. In the above example, a colon (:) is used to delimit `<#>` and `<*>`.

Matching **<@.word><#.num>** against `abc123` assigns `abc12` to **word** and `3` to **num**, as digits are permitted for both `<#>` and `<@>`, and the left expression takes as many characters as possible.

Patterns without expression anchoring can match any substring within the input line. For example, the `this is number<#.num>` pattern is treated in the same way as the following:

```
<*>this is number<#.num><*>
```

Using Sub-patterns to Assign Strings to Variables

In addition to being able to use a single operator, such as a star (*) or a number sign (#), to assign a string to a variable, you can also build up a complex sub-pattern composed of a number of operators, according to the following pattern: `<[sub-pattern].var>`

For example:

```
<[<@>file.tmp].fname>
```

Note that in the example above, the period (`.`) between `file` and `tmp` matches a similar dot character, while the dot between "]" and "`fname`" is necessary syntax. This pattern would match a string such as `Logfile.tmp` and assigns the complete string to `fname`.

Other examples of sub-patterns are:

```
<[Error|Warning].sev>
```

```
<[Error[<#.n><*.msg>]].complete>
```

In the first example above, any line with either the word `Error` or the word `Warning` is assigned to the variable, `sev`. In the second example, any line containing the word `Error` has the error number assigned to the variable, `n`, and any further text assigned to `msg`. Finally, both number and text are assigned to `complete`.

Configuring a Message Policy Condition to Match a Multiline Message

To configure a message policy condition to match a multiline message sent by using the `opcmsg` command, use the "`</>`" or "`<n/>`" pattern that matches exactly `n` line breaks (for example, "`<1/>`" to match exactly one line break).

When configuring the Set Attributes: Message Text field to display a multiline message in the message browser, use "`\n`". For example:

Message Text field (Condition):

```
^First line: <*.text1><1/>Second line: <*.text2>$
```

Message Text field (Set Attributes):

```
Message with two lines: \n First line: <text1>\n
Second line: <text2>
```

To send a multiline message from the command line, place the message text name within quotation marks and press ENTER after each line. For example:

```
# opcmsg a=a o=o msg_t="First line: Hi
Second line: there"
```

If the previous policy condition is active, the following message text appears in the message browser:

```
Message with two lines:
  First line: Hi
  Second line: there
```

Displaying Matched Messages

After a message matches a message condition, you can assign certain settings to the message before it is displayed in a browser.

Assigning Message Settings

You can assign new values for the following settings:

- Severity level
- Node
- Application
- Message group
- Object
- Message text
- Message type
- Message key
- Service name

Any attribute set at the condition level overrides the value of the same attribute set by the policy defaults. You can also use part of the message text as a parameter to redefine the message text before the message is forwarded to an operator's browser.

Adding Custom Message Attributes to Your Message

Custom message attributes allow you to add your own attributes to a message. This means that in addition to the default message attributes listed in "Implementing Message Policies" on page 127, you can extend HPOM messages with attributes of your choice, for example, the attribute "Customer" or the attribute "SLA" for service level agreements.

Custom message attributes can only be set for message conditions and are only available for log file, HPOM interface, and threshold monitor policies.

Use the `opccmachg` command-line tool to assign attributes of your choice to a message. For more information, see the *opccmachg(1m)* manpage.

When creating and assigning custom message attributes, you can specify attribute name and value, for example:

```
# opccmachg -user opc_op -id
55d3604a-536f-71db-08c0-0a1108c90000 CUSTOMER=VIP SLA=none
Device=Device1 Source=Node1
```

A message matching the following condition would display with four additional columns in the Java GUI browser:

- `Customer`
- `Device`
- `SLA`
- `Source`

The values can contain one or more of the following:

- Hard-coded text
- Variables returned by the HPOM pattern-matching mechanism

  For more information, see "Implementing Message Policies" on page 127.
- Predefined HPOM variables

  For more information, see the *HPOM Administrator's Reference*.

Note: Custom message attributes are only displayed in the browser and message properties windows of the Java GUI.

If so configured, custom message attributes are passed to the Message Stream Interface (MSI) on the agent, the management server, or both. Custom message attributes are also passed to the trouble ticket system, the notification service, or both.

Adding Instructions to Your Message

You can add instructions to your message. Typically, these instructions describe an automatic action, provide details of how an operator should perform an operator-initiated action, or describe other manual steps for resolving a problem.

To add instructions to your message, use one of the following methods:

- **Write Instructions**

  Write instructions for the message condition. All messages matching the condition then have the instructions associated with them. The text can be viewed in the Message Properties window of the Java GUI message browser. The advantage of using simple, text-based instructions is that they are stored in the database.

  This method has the following advantages:

  - Instructions are highly reliable.
  - Instructions are resolved at a very high speed.
- **Call an External Application to Send Instructions**

  Use the instruction text interface to call an external application to present instructions to an operator. The advantage of this method is that parameters of many kinds can be passed to the interface.

  This method is potentially very flexible:

  - *Variables*

    Variables may be included in the text, which among other things enables the complete localization of the instructions.

- *Message-specific*

  Instructions can be message-specific, rather than just message-condition-specific.

For the Java GUI, special HPOM variables are available to call an external application or to open a web browser on the client where the Java GUI is currently running. For details, see the *HPOM Administrator's Reference*.

Responding to a Message

HPOM provides several options for responding to messages that match conditions. Operators use some of these options in message browser to respond to messages. Some of these responses are transparent to operators.

Responses

You can choose from the following response types:

- **Logging Messages on the Management Server Only**

  If you choose this option, messages that match a message condition are logged on the management server and stored in the history database. They are not processed further, but operators can view them in the Java GUI `History Message` browser.

  If you log the messages on the management server only, the other actions are ignored.

- **Defining an Automatic Action**

  An **automatic action** starts immediately when the message is received. For each action, you must define the node on which it is performed, and the command to be executed (shell script, program, application start, or other response). Operators can stop currently running actions, and restart automatic actions, if necessary.

  You can also specify that an automatic action provides an annotation, and that the annotation be acknowledged automatically if successful. If you set up an automatic action with automatic acknowledgment, the operator might not see the message in the Java GUI message browser.

- **Defining an Operator-Initiated Action**

  Operators can start an **operator-initiated action** after reviewing the message in a Java GUI message browser. As is the case with automatic actions, operators can stop currently running actions, and restart them if necessary. You can define the node and command. You can also specify that an annotation and acknowledgment be provided automatically.

  As a general rule, in instructions, you enter details about the operator-initiated actions, so operators know what exactly will be executed when the operator-initiated action is started. Normally, an operator-initiated action requires some kind of operator interaction. Or operators must set up or verify some type of prerequisite.

Examples:

- Stopping the database before starting a backup.
- Informing users that, before print spooling, the subsystem will go into maintenance mode.

- **Forwarding Messages**

  You can forward messages to a trouble ticket system or external notification service. In addition, you can configure automatic acknowledgments after forwarding a message.

Configuring Automatic Annotations and Acknowledgments

For both automatic and operator-initiated actions, you can configure automatic annotations and automatic acknowledgments.

An automatic annotation logs the following:

- Start and stop time of action
- Exit value of action
- Action information written to `stdout` and `stderr`

If an action fails, an annotation is automatically written. When you configure an automatic acknowledgment for an action, the message is acknowledged automatically if processing of the action was successful. Without automatic acknowledgment, operators must manually acknowledge messages in the Java GUI Browser.

Strategies for Optimal Message Filtering

This section suggests ways to optimize your message filters to improve system performance and ensure that operator browsers are free of duplicate or unimportant messages.

Filtering Messages

You can filter messages on the managed node and on the management server:

- **Managed Node**

  Filtering out as many messages as possible on the managed node minimizes network traffic and reduces the load on the management server.

- **Management Server**

  Filtering on the management server enables you to compare and correlate messages from several nodes. If so configured, the management server can maintain a counter of suppressed messages. **Regroup conditions** let you customize the grouping of messages that operators see in the Java

GUI message browser. Regroup conditions are not used to filter out messages. For more information about regroup conditions, see "Regrouping Messages " on page 181.

Optimizing Performance

Optimal processing performance can be achieved easily by putting the conditions into a sequence and by deploying Suppress Unmatched conditions.

Organizing Conditions into a Sequence

The sequence in which conditions display within a policy determines the type and number of messages processed through the system. In principle, a policy that begins with suppress unmatched or suppress conditions (thus filtering out unwanted messages from the start) demands less processing than a policy in which the message conditions are placed first. Fewer messages to match reduces processing and increases performance.

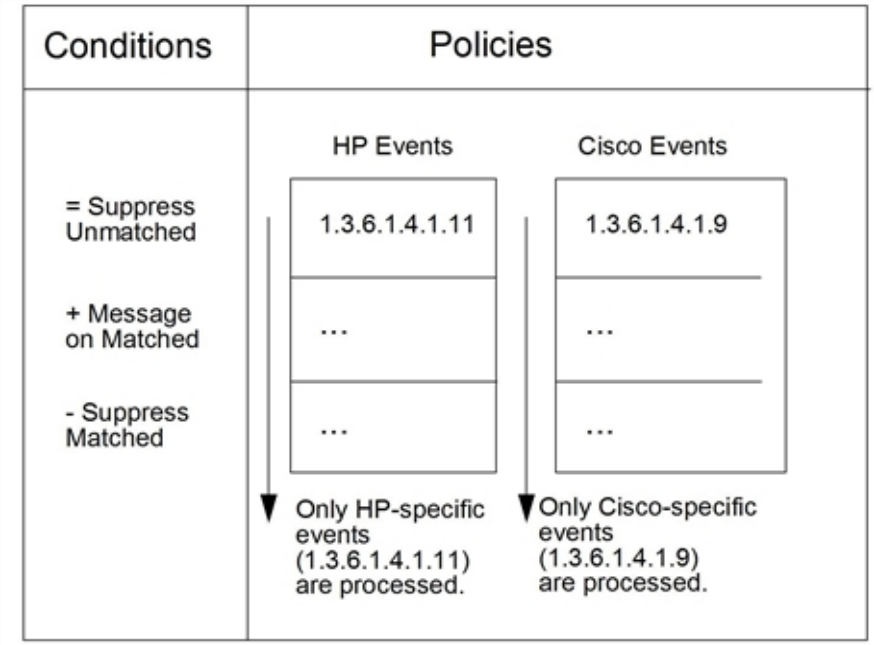
Deploying Suppress Unmatched Conditions

Suppress Unmatched conditions filter events on the managed nodes. They enable you to stop the matching process for events not “intended” for a particular policy. Suppress Unmatched conditions suppress events that do *not* match the specified pattern, but allow events that match to be processed by the conditions in the list. Message suppression improves performance on the managed node because HPOM processes only those events intended for the policy.

For example, to improve performance of SNMP trap filtering, create one policy for each enterprise-specific event occurring in your environment. Then place a Suppress Unmatched condition first in the list of conditions. HPOM suppresses SNMP traps from MIB objects not matching the condition, and only processes the SNMP traps intended for that policy.

In Figure 25, the policy for HP-specific SNMP traps suppresses SNMP traps from Cisco MIB objects, and only processes HP-specific events.

Figure 25: Channeling Enterprise-specific SNMP Traps



Reducing the Number of Messages

Operators are often overwhelmed by the number of messages HPOM sends to their message browsers:

- **Related Messages**

  Some messages are related to each other (for example, an application stops and starts up again).
- **Similar or Identical Events**

  Some messages report similar or identical events (for example, a user switches to user root three times).
- **Problem Deterioration**

  Some messages report how a problem situation deteriorates (for example, the amount of free disk space decreases on a managed node).

HPOM can be configured so operators receive only important and relevant messages. Messages that relate to the same or to a similar problem can be suppressed, or correlated and replaced with a new, more meaningful message.

Correlating Messages and Events

Message replacement is achieved through message correlation and event correlation.

- **Message Correlation**

  Message correlation can be achieved with built-in HPOM mechanisms, but offers only basic correlation techniques. Message correlation is recommended if you want to get started with correlation techniques and then proceed to a more sophisticated solution.

- **Event Correlation**

  For details about event correlation, see "Event Correlation in HPOM" on page 204.

Table 12 clarifies the differences between event correlation and message correlation.

Table 12: Comparing Event Correlation and Message Correlation

Event Correlation	Message Correlation
• Default EC policies delivered with HPOM. • Purchase of the event correlation product is required, for example, HP Event Correlation Designer.	• No separate purchase is necessary.
• More difficult to set up and maintain, but supports complex conditions.	• Easy to configure, but supports only simple correlation tasks.
• Handles event streams. Events can change their state while they are processed by the event correlation engine. Identical input events can generate different output events, depending on the current state.	• Static message handling.
• "Annotate node" concept of HP Event Correlation Designer allows you to attach different actions to the output events.	• Only suppression[a] or automatic acknowledgment is possible.
• Data is exchanged between HPOM and the event correlation product.	• All data is processed by HPOM. Performance is not affected.
• Loss of data is possible when event correlation services go down.	• All data is processed by HPOM. If HPOM goes down, data is stored in a database.

Message Correlation

Message correlation describes the mechanism by which HPOM compares similar or identical events and messages.

HPOM reacts to events and messages in one of two ways:

[a] If enabled on the management server, suppressed messages are also counted.

- **Automatic Acknowledgement**

  Automatically acknowledges the message to which a relation was established (see "Automating Standard Scenarios" on page 168).

- **Duplicate Message Suppression**

  Suppresses duplicate messages (see "Suppressing Duplicate Messages " on page 173).

If duplicate message suppression is enabled on the management server, HPOM also keeps a counter of the suppressed messages.

Message Keys

Messages are correlated on the basis of their **message key**. In some instances, other event attributes or message attributes are compared. The message key is a message attribute that summarizes the important characteristics of the event that triggered the message.

Use `MSGKEY` and `MSGKEYRELATION ACK` keywords to set up message key relations.

Guidelines for Effective Message Keys

Effective message keys provide a concise description of the event that triggers a message. Message keys should contain only important information about the event. They should exclude unnecessary data, such as time stamps or fill words. Effective message keys can be used for state-based browsers (see "Automating Standard Scenarios" on page 168) and for suppressing duplicate messages (see "Suppressing Duplicate Messages " on page 173).

To build effective message keys, follow these guidelines:

- **Include the Node Name in the Message Key**

  Incorporate the HPOM variable `$MSG_NODE_NAME`. This variable ensures that messages generated on one computer have a different message key from messages generated on another computer.

  Example:

  ```
  my_appl_down:<$MSG_NODE_NAME>
  ```

- **Include Other Important Message Attributes**

  A message key should take into consideration all message attributes that describe the important aspects of the message. Typical attributes are node, object, application, severity, service name, monitor name (when defining a condition for a threshold monitor policy), or any variables that are defined in the `Condition` section.

  Example:

  ```
  appl_status:<$MSG_APPL>:<$MSG_OBJECT>:<$MSG_NODE_NAME>
  ```

  For a list of HPOM variables that can be used, see the *HPOM Administrator's Reference*.

- **Reflect the Severity of the Message**

Messages with different severities should have different message keys. This does not necessarily mean you should include the severity string itself in the message key. Try to reflect the severity by including the cause of the severity level. The cause is included in the message information itself. For example, for threshold monitor policies, the variable <$THRESHOLD> can be included. Each value of <$THRESHOLD> represents a distinct severity level.

- **Enable HPOM to Generate a Default Key**

  To generate a default message key along with a default message key relation for each condition of the policy, use the AUTOMATIC_MSGKEY keyword, optionally followed by a string value. This keyword also supplies existing conditions with a message key.

  For more information about message key relations, see "To Generate a Default Message Key and Message Key Relation" on page 170.

- **Improve Readability**

  Separate the components of a message key from each other, for example, with colons (:).

  Example:

  ```
  my_appl_down:<$MSG_NODE_NAME>
  ```

Guidelines for Effective Message Key Relations

To build effective message key relations, follow these guidelines:

- **Watch Out for Variable Resolution**

  Message key relations consist primarily of HPOM variables that are resolved on the managed nodes. Relations can also contain HPOM pattern definitions. The patterns are matched on the management server.

- **Improve Readability**

  Separate the components of a message key relation from each other (for example, with colons (:)).

  Example:

  ```
  my_appl_down:<$MSG_NODE_NAME>
  ```

- **Enclose Relations in Anchoring Characters**

  Enclose message key relations in anchoring characters. Use the caret (^) as the first character, and the dollar sign ($) as the last. This improves processing performance.

  Example:

  ```
  ^<$NAME>:<$MSG_NODE_NAME>:<$MSG_OBJECT>:<*>$
  ```

- **Place Pattern Definitions in the Correct Location**

  If you use pattern definitions in message key relations, place them into the right part of the relation string. Correct pattern-definition placement improves processing performance.

  Example of pattern definitions in the right place:

  ```
  ^<$MSG_NODE_NAME>:abcdef:[pattern]$
  ```

Example of pattern definitions in the wrong place:

```
^[pattern]:<$MSG_NODE_NAME>:abcdef$
```

- **Specify Case-sensitive Check and Field Separators**

For a list of HPOM variables that can be used when defining which messages are to be acknowledged, see the *HPOM Administrator's Reference*.

Recommendations for Message Key Correlation Patterns

The following is a list of factors that may impact the time needed to process a message key correlation pattern and recommendations that will help you improve its performance:

- **Number of active messages**

  Recommendation: Keep the number of active messages low.

- **Length of message keys**

  Recommendation: Make sure message keys are of a reasonable length.

- **Anchoring the pattern**

  Recommendation: You should anchor the pattern—especially the beginning of the pattern by using the caret (^). To anchor the end of the pattern, use the dollar sign ($).

- **Multiple variations**

  Recommendation: Combine multiple alternatives by using the asterisk wildcard (*).

  For example, instead of using:

  ```
  UP_xyz|DWN_xyz_cri|DWN_xyz_maj|DWN_xyz_min|DWN_xyz_war
  ```

  You could use:

  ```
  UP_xyz|DWN_xyz_<*>
  ```

  Or, if you have two different conditions for `UP` and `DOWN`, you could have the specific ones, one for the `UP` message and one for the `DOWN` message:

  ```
  ^UP_xyz<*>$
  ^DWN_xyz_<*>$
  ```

- **Combinations of patterns right next to each other**

  Recommendation: Avoid combinations of patterns right next to each other (for example, `some<*><@><*>thing`).

Automating Standard Scenarios

Sometimes, you may want to acknowledge messages automatically.

The following example illustrates this:

Problem Resolution

A first message might report a problem. Then, a second message might report that the problem has been solved (for example, if a user fails to log on because of an incorrect password, but succeeds with a second attempt). Or it might report that the problem has deteriorated. In either case, the first message would no longer be relevant. For this reason, you would want to the second message to automatically acknowledge the first message.

HPOM enables you to automate such scenarios (for example, an application shutting down and starting up again).

State-Based Browsers

When you acknowledge messages automatically, a maximum of one message per managed object exists in the browser. This message reflects the current status of the object. In effect, the message browser has become a state-based browser. (For suggestions on how to implement this concept for threshold monitors, see "To Generate a Default Message Key and Message Key Relation" on the next page.)

Acknowledging Messages with Message Keys

When you work with related message, the relationship between the first and the second (or the second and the third) message is established through message keys. A message key acknowledges a message by matching, and thereby identifying, the message key of that message. The pattern is specified with `MSGKEYRELATIONS ACK` keywords in the policy body.

Annotating Acknowledged and Acknowledging Messages

When a message is acknowledged automatically by another message, both messages are annotated, as follows:

- **Acknowledged Message**

  The *acknowledged* message is annotated automatically with details about the *acknowledging* message. These details about the *acknowledging* message include its message ID, condition ID, and message key relation.
- **Acknowledging Message**

  The *acknowledging* message is annotated with details about the *acknowledged* message. These details about the *acknowledged* messages include its message key relation, the number of messages it has acknowledged, its message IDs, and its condition IDs.

These annotations can be useful when troubleshooting.

Note: The status of the message has no influence on whether it will be acknowledged: owned messages, pending messages, and messages with running actions are also acknowledged.

To Generate a Default Message Key and Message Key Relation

For threshold monitor policies, HPOM can generate a default message key, along with a message key relation, for each condition.

To generate a default message key and a message key relation for each condition, use the `AUTOMATIC_MSGKEY` keyword.

The following default values are generated:

- **Message Key**

  `<$NAME>:<$MSG_NODE_NAME>:<$MSG_OBJECT>:<$THRESHOLD>`
- **Message Key Relation**

  `^<$NAME>:<$MSG_NODE_NAME>:<$MSG_OBJECT>:<*>$`

For example, a resolved message key could look like the following:

`disk_util:managed_node.hp.com:/:90`

This message key indicates that the monitor `disk_util` has detected more than `90`% used disk space in the root directory (`/`) on the node `managed_node.hp.com`.

The message key relation ensures that all those messages are acknowledged that are triggered by the monitor `disk_util` and report that disk utilization of `/` on the node `managed_node.hp.com` has exceeded or fallen below a threshold.

Sending a Reset Message Automatically

HPOM also automatically acknowledges the last message of a monitored object by sending a reset message if a threshold was first exceeded and the monitored value then drops below all possible reset values. In other words, no condition matches any longer.

The reset message is provided with a message-key relation that acknowledges the last message that was sent for a certain monitor. This last message must contain a message key. Otherwise, no reset message is sent.

The reset message cannot be configured, but has the following default text: `<monitor_name> [(<instance>)]:<value> (below reset)`.

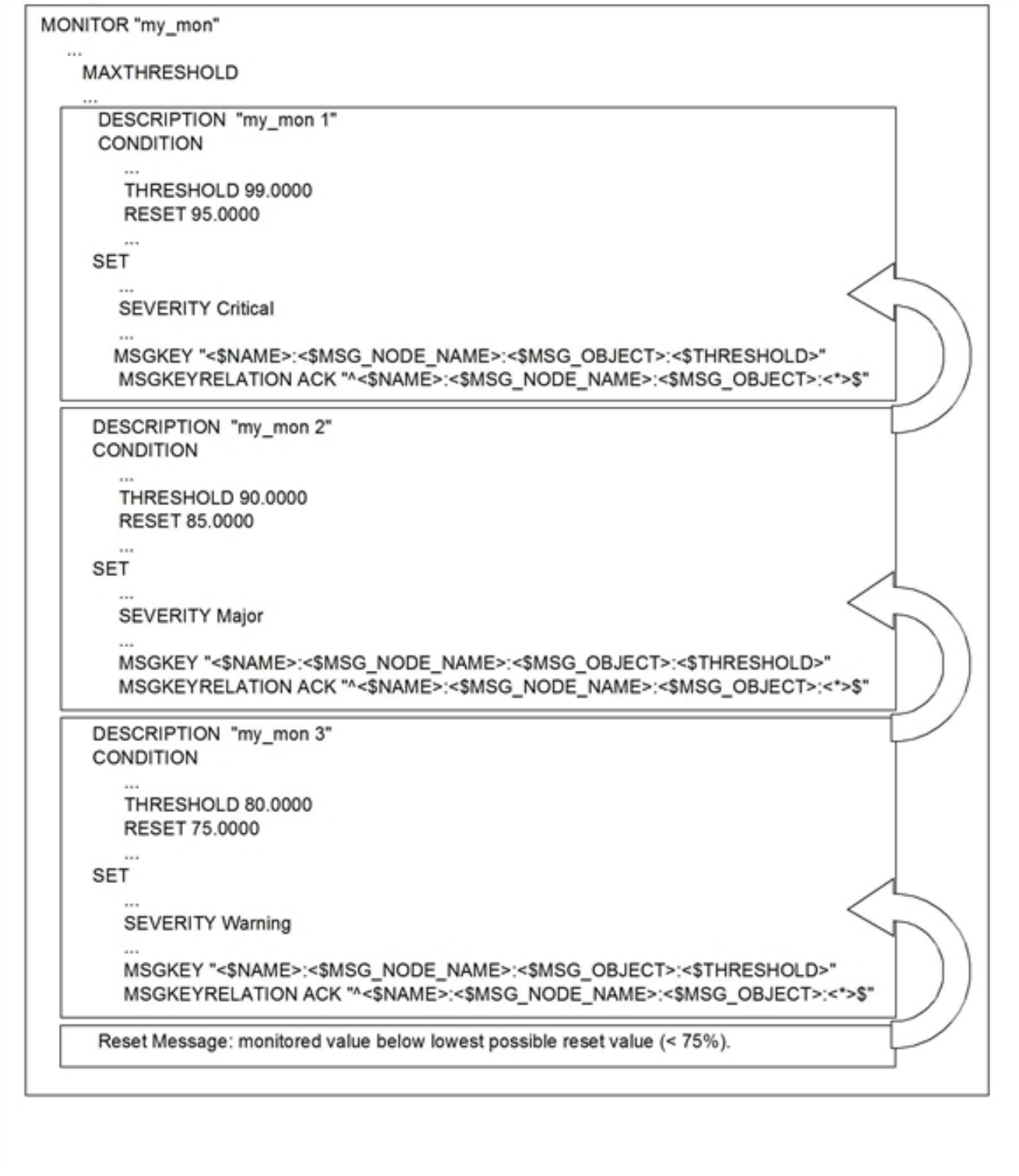
The reset message does not display in the message browser, but is sent directly to the history database. The severity of the reset message is set to normal. Automatic or operator-initiated actions are not defined.

For details on how the monitor agent behaves when multiple conditions exist for one monitored object, see "Threshold Monitoring with Multiple Conditions" on page 199.

Example of an Automatic Reset Message

Figure 26 shows an example of a reset message sent automatically.

Figure 26: Example of an HPOM Reset Message



In this example, the `my_mon` monitor has three conditions:

- `my_mon 1`

  Generates a critical message when the monitored value exceeds 99%. When the value falls below 95%, the counter is reset.

- `my_mon 2`

  Generates a major message when the monitored value exceeds 90%. When the value falls below 85%, the counter is reset.

- `my_mon 3`

  Generates a warning message when the monitored value exceeds 80%. When the value falls below 75%, the counter is reset.

The message key relation of each message ensures that the previous message is automatically acknowledged. The last message (generated by “my_mon 3”) is automatically acknowledged by the reset message when the monitored value falls below the lowest possible reset value (in this example, below 75%).

The resolved message keys of each generated message are different from each other. Each resolved message key contains the threshold value specific to the condition, and thereby indicates the severity of the message.

Suppressing Duplicate Messages

In general terms, duplicate messages are messages that report the same event or a similar event. For example, each time a user switches to another user, HPOM generates a message. If the user always changes to the same user, you could consider this information superfluous, declare it an identical event, and have HPOM suppress further messages relating to it. In addition, HPOM lets you decide how often or for how long further messages are suppressed before a “new” duplicate message is sent.

Note: If an incoming, duplicate message has a different severity or message text than the existing duplicate messages, you can configure HPOM to display the new values instead of the previous data. For more information, see "Implementing Message Policies" on page 127.

You can configure HPOM to suppress duplicate messages on the managed node or on the management server. Filtering out (or suppressing) messages on the managed node reduces network traffic and keeps the management server free for other tasks.

Because suppressing messages on the management node is more beneficial to performance, HPOM offers a variety of suppression types and settings on the managed node, but only a global setting on the management server. Use the configuration variable `OPC_MAX_DUPL_ANNO` to limit the number of duplicate messages for which annotations are added.

Verifying Suppression Types

If the condition event matches the condition, HPOM verifies that one of the following suppression types has been selected:

- **Suppress Messages Matching Condition**

  HPOM suppresses *all* duplicate messages matching the chosen condition. That is, HPOM checks whether a message generated by the same condition already exists.

  If a message already exists, and the second event occurs within a specified time interval or below a configured counter, the second message is suppressed.

- **Suppress Identical Input Events**

  HPOM suppresses only those duplicate messages whose *event attributes* are identical to each other. That is, HPOM checks whether a message already exists that was generated by the same input events. A message's input events are defined in the `Condition` section of the policy condition. Two or more messages are considered to be identical if the following event attributes are the same:

  - Severity
  - Node
  - Application
  - Message group
  - Object
  - Message text (= original message text)

  If an identical message already exists, and the second event occurs within a specified time interval or below a configured counter, the second message is suppressed.

- **Suppress Identical Output Messages**

  HPOM suppresses only those duplicate messages whose *message attributes* are identical to each other. That is, HPOM checks whether a message already exists that has the same message attributes. The attributes of a message are defined in the `Set Attributes` section of the policy condition. Two or more messages are considered to be identical if their message keys are the same.

  If either message does not have a message key, the following message attributes must be identical:

  - Severity
  - Node
  - Application

- Message group
- Object
- Message text
- Service name

If an identical message already exists, and the second event occurs within a specified time interval or below a configured counter, the second message is suppressed.

The log file of the `syslog` daemon, `/var/adm/syslog/syslog.log` on HP-UX, illustrates suppressing duplicate messages with the Suppress Identical Output Messages option.

For example, consider the following lines from the `syslog` log file:

```
Mar 14 14:39:01 server inetd[9900]: telnet/tcp: Connection from node1 at Tue Mar
14 14:39:01 2009
Mar 14 12:46:02 server inetd[9005]: login/tcp: Connection from node2 at Tue Mar
14 12:46:02 2009
```

The pattern-matching text might look like this:

```
"inetd\[<#>\] <@.service>: Connection from <@.from_node>"
```

The important characteristics of the `inetd` connection messages are the local node, the node from which the connection is made, and the `inetd` service. The `syslog` time stamp, the PID, and the connection time are not important.

A message key might look like this:

```
inetd_connect_from:<$MSG_NODE_NAME>:<from_node>:<service>
```

This message key would suppress all messages from the same nodes that connect to the same node and connect for the same service.

Duplicate message suppression processes only those messages that are generated by the same condition. If you want to suppress duplicate messages that are generated by different conditions, enable this functionality on the management server. For details, see "Suppressing Duplicate Messages on the Management Server" on page 178.

Note: Duplicate message suppression on managed nodes is not available for threshold monitor policies. To find out how to suppress duplicate messages from threshold monitor policies, see "Suppressing Duplicate Messages on the Management Server" on page 178.

You can specify the type of duplicate message suppression, as well as the time interval and counter or threshold settings. There are three types of duplicate message suppression. Use the following keywords to achieve the desired results:

SUPP_DUPL_COND	Suppresses all messages that match the same condition.
SUPP_DUPL_IDENT	Suppresses all messages that have same original message text (input).
SUPP_DUPL_IDENT_OUTPUT_MSG	Suppresses all messages that have the same output message text.

The value following these keywords represents the time interval within which messages are suppressed. If instead of a time value, a keyword COUNTER_THRESHOLD (followed by a number) is used, the messages are suppressed until the defined number of messages are received, after which one message is sent and the counter is reset. Using keyword RESET_COUNTER_INTERVAL will set a time interval after which the counter is reset regardless of the number of messages received. Keyword RESEND sets the time limit after which messages are being sent again. For more information, see the policy body grammar in "Policy Body Grammar" on page 253.

Type of Suppression Settings

You can choose between the following suppression settings:

- **Time Interval**

  Specify a time interval during which duplicate events are ignored, and a time period after which you want to start sending messages again. In the example shown in Figure 27, the suppression time interval is set to 30 seconds, but the suppression time is limited to 60 seconds.
- **Counter**

  Specify a threshold for a duplicate message counter. HPOM increments the counter until it equals or crosses the threshold. At that point, HPOM allows the transmission of the duplicate message. In the example shown in Figure 28, the counter threshold is set to two. The counter is reset after 30 seconds.
- **Combination of Time Interval and Counter**

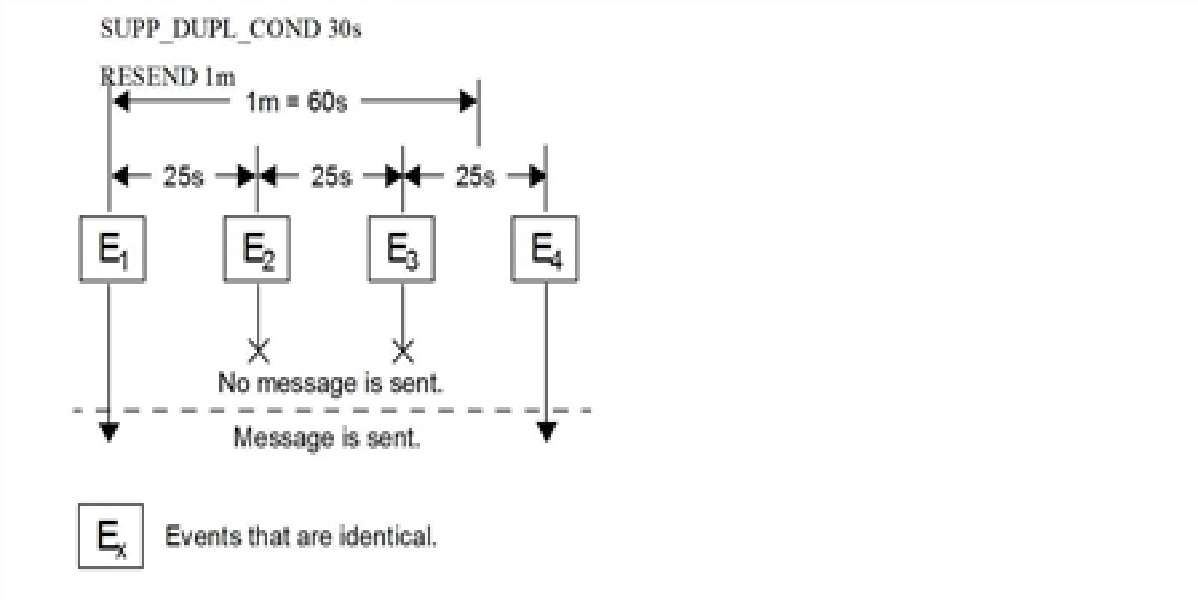
  If you use the time interval and counter together, events are evaluated first by the timer. If an event passes the timer, it is then evaluated by the counter, which either suppresses it, or sends a message to the management server.

Suppression Based on Time

Figure 27 illustrates suppression based on time.

For suppression of identical input events, use SUPP_DUPL_IDENT keyword instead. For suppression of identical output messages, use SUPP_DUPL_IDENT_OUTPUT_MSG keyword.

Figure 27: Suppression Based on Time

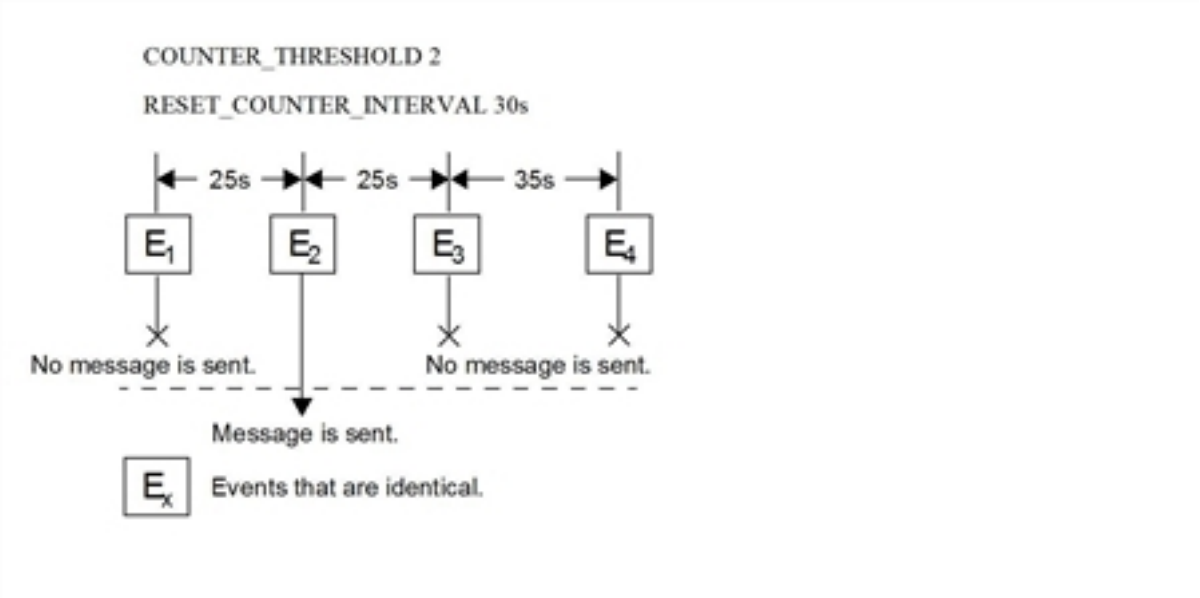


1. The first event (E_1) matches a condition. A message is sent. The timer is started.
2. A second event (E_2) occurs 25 seconds later. This event occurs *less than 30 seconds* after the first event. So it is suppressed.
3. A third matching event (E_3) occurs *less than 30 seconds* after the second event. So it is also suppressed.
4. The next matching event (E_4) occurs *less than 30 seconds* after the third event. But it also occurs *more than 60 seconds* after the first event. So a new message is sent.

Suppression Based on Counter

Figure 28 illustrates suppression based on counter.

Figure 28: Suppression Based on Counter



1. The first event (E_1) matches a condition. The counter increments to one. No message is sent because the threshold of two has not yet been reached.
2. A second matching event (E_2) occurs 25 seconds later. The counter increments to two. A message is sent. The counter resets to zero.
3. A third matching event (E_3) occurs. The counter increments to one. No message is sent.
4. The next matching event (E_4) occurs more than 30 seconds after the third event. At 30 seconds, the counter was reset to zero. So the counter now increments to one. No message is sent.

Suppressing Duplicate Messages on the Management Server

Suppression of duplicate messages can also be configured for the management server. By suppressing duplicate messages on the management server, you can significantly reduce high system loads caused by large numbers of messages. In addition, you can correlate messages from more than one managed node.

The suppression method used by the management server is identical to the `Suppress Identical Output Messages` method used on the managed node. HPOM compares the message attributes of incoming messages with the message attributes of existing messages.

If *either* message does not have a message key, the following message attributes must be identical:

- Severity
- Node
- Application

- Message group
- Object
- Message text
- Service name

If an identical message already exists, HPOM suppresses the duplicate message and all subsequent duplicate messages. HPOM also starts a counter of duplicates on the first message. This counter is displayed in the browser windows of the responsible operator, as well as in the Message Properties window. The counter gives an indication of how often the problem occurred. The Message Properties window also shows when the last duplicate message was received (and suppressed).

If suppression is enabled, HPOM stores information about suppressed duplicate messages in annotations to the first message.

Note: Automatic actions started by duplicate messages on the managed node are still executed. Any action responses, however, are lost.

Enabling Duplicate Message Suppression on the Management Server

Duplicate message suppression on the management server can be enabled with the `opcsrvconfig -dms` command. You can also specify that the duplicated messages are added as annotations. For example:

```
# opcsrvconfig -dms -enable anno
```

Note: Duplicate message suppression is a global setting that affects all messages and operators.

For more information, see the *opcsrvconfig(1m)* manual page.

Suppressing Duplicate Messages in Flexible Management Environments

In flexible MoM environments, each management server is responsible for counting the messages it receives from its direct managed nodes. This means that messages received from other management servers are not aggregated into one message but are displayed as multiple messages, each with the count as received from the originating management server.

If you do not want a management server to send or receive message count events, set the following variables on the HP Operations management server by using the `ovconfchg` command-line tool:

- **Send message count events**

  ```
  ovconfchg -ovrg <OV_resource_group> -ns opc -set OPC_SEND_MSG_COUNT FALSE
  ```

- **Receive message count events**

  ```
  ovconfchg -ovrg <OV_resource_group> -ns opc -set OPC_ACCEPT_MSG_COUNT FALSE
  ```

<OV_resource_group> is the name of the management server resource group. By default, both variables are set to `TRUE`.

Updating Severity and Message Text of Duplicate Messages

If an incoming, duplicate message has a different severity or message text than the already existing messages, you can configure HPOM to display the new values instead of the previous data:

- **Update severity of duplicate messages**

  The following command configures HPOM to display the severity of the last duplicate message received:

  ```
  ovconfchg -ovrg server -ns opc -set OPC_UPDATE_DUPLICATED_SEVERITY LAST_MESSAGE
  ```

- **Update message text of duplicate messages**

  The following command configures HPOM to display the message text of the last duplicate message received:

  ```
  ovconfchg -ovrg server -ns opc -set OPC_UPDATE_DUPLICATED_MSGTEXT LAST_MESSAGE
  ```

When set to `LAST_MESSAGE`, the appropriate value will be changed in the message browser.

If you prefer the default behavior, which is no update of the severity and message text fields, do not set these two variables or set them explicitly to `FIRST_MESSAGE`.

Logging Messages

The message handling facility of HPOM produces the following types of message results:

- **Messages Matching Message Conditions**

  These messages are filtered into HPOM. They are forwarded from the local node to the management server for further processing. You can also specify logging on the source node of these messages. Matched messages are displayed in the browser windows of the responsible operators.

- **Messages Matching Suppress Conditions**

  These messages are filtered out of HPOM without further processing. You can choose to log these messages on their source nodes.

- **Messages Not Matching Any Conditions**

If no filtering occurs, unmatched messages can still be routed into HPOM. Typically, unmatched messages are messages you have not seen before, so you have not yet set up filtering conditions for them. You can choose to log these unmatched messages locally on the managed node, or forward them to the management server for processing. If you forward them to the management server, you can then choose to display them in the Java GUI message browser, or to put them directly into the history database. If they are displayed in the browser, they are identified by an `X` in the U column of the Java GUI message browser.

You can set up logging options after you have defined message source policies and their message and suppress conditions. You specify how you want message types logged by using the following keywords in the policy body:

```
LOGMATCHEDMSGCOND
LOGMATCHEDSUPPRESS
LOGUNMATCHED
```

Note: If you switch on logging for any of the event correlation policies, logging is automatically enabled for all other event correlation policies.

Regrouping Messages

After a message has been integrated and filtered by HPOM, you can change its current default message group on the management server. You can customize message groups specifically suited to an operator's tasks and responsibilities. This means that you are not required to change the conditions for a message source. And you are not required to redistribute the policies to move messages from one message group to another.

Message conditions and attributes are processed as follows:

- **Message and Suppress Conditions**

  Message conditions and suppress conditions are processed on managed nodes, where they are compared with all incoming messages.

- **Regroup Conditions**

  Regroup conditions are processed on the management server only. Regroup conditions are compared to messages that have already passed through the filters on managed nodes.

  Messages that match a regroup condition are forwarded to a different message group, according to your policies. You can group all messages from the spooling applications can be grouped into the message group `Output`.

- **Messages Attributes**

  As with message and suppress conditions, message attributes you define for the regroup condition are used in any check against the actual values of the messages.

Defining a Regroup Condition

If you want to define a regroup condition, use the Administration UI.

Note: If you regroup messages into a message group that does not exist, all messages relating to this message group belong, by default, to the message group `Misc` until that message group is created. To check the original message group of a message currently assigned to `Misc`, use the Java GUI Message Properties window of the Java GUI message browser.

Alternatively, you can use the regroup conditions APIs.

Examples of Regroup Conditions

The message conditions shown in earlier examples form the basis for the regroup condition example shown in this section.

All messages filtered into HPOM have been forwarded to the message group `FINANCE`.

Now you want to split the messages into two groups:

- Payroll
- Accounting

The regroup conditions for these two groups are listed below:

Regroup Condition No. 1

Application:	`idris4|idris5`
Application:	`FINANCE PAYROLL`
Text Pattern:	`^***PAYROLL:[ERROR|WARNING]`
New Message Group:	`payroll`

Regroup Condition No. 2

Application:	`idris4|idris5`
Application:	`FINANCE ACCOUNTING`
Text Pattern:	`^***ACCOUNTING:[ERROR|WARNING]`
New Message Group:	`accounting`

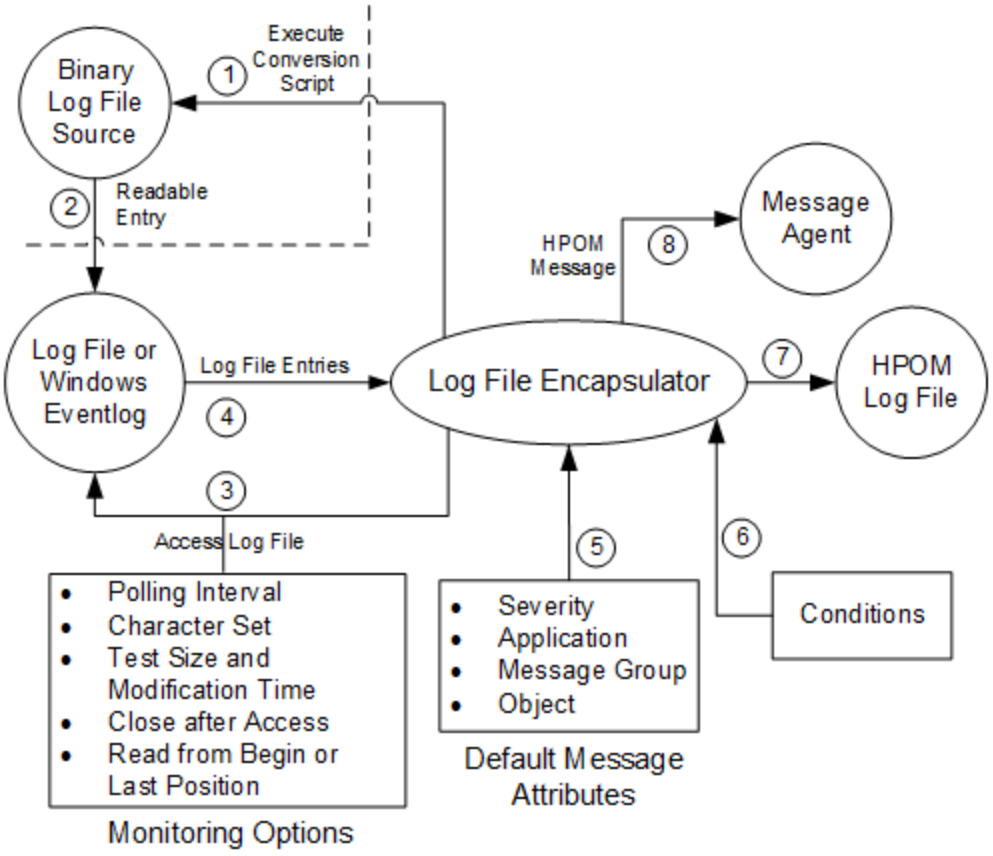
Log File Messages

Application and system log files are intercepted by the HPOM **log file encapsulator**. You can bring messages into HPOM from an application or service that writes a log file, without making any changes to the application or service. HPOM provides default logfile entry policies you can copy and modify to meet your specific needs. With multiple policies, you can set up log file monitoring for a variety of applications and services.

Log File Encapsulator

Figure 29 shows how the log file encapsulator collects, filters, reformats, and displays log file messages in the Java GUI Browser.

Figure 29: Log File Encapsulator



Steps 1 and 2 are required only if the log file is in binary format.

Step 7 is performed only if local logging is configured.

Step 8 is applied if the log file entry has been matched by a message condition, or if the forward unmatched condition is true.

Logfile Entry Policies

The Logfile Entry policy defines the default message attributes that describe all messages from the source. The policy also includes monitoring options that specify how and when a log file is monitored, as shown in Figure 29.

Settings for a Logfile Entry policy include:

- **Policy Name and Description**

  The `opcpolicy -list_pols` command lists the name and description of the policy.
- **Pathname and Name of the Log File**

  The location of the log file in the file system. On UNIX managed nodes, you can use shell variables to set up dynamic paths. You can also enter the name of a command or script that dynamically discovers the name of the log file and writes it to `stdout`.
- **Monitoring Options**

  Monitoring options include a command or program to be executed before scanning the log file, an alternate file name, the polling interval, the log file character set, and details about how monitoring should occur.

  HPOM provides three read position alternatives from which a log file is processed:

  - *Read from the last file position*

    Monitors only for new—appended—entries.
  - *Read from the beginning of the file (first time)*

    Monitors the entire log file the first time the log file encapsulator starts monitoring. With the next polling interval, only new—appended—entries are monitored.
  - *Read from the beginning of the file (always)*

    Reads the entire log file when a modification to the log file is detected. The log file encapsulator does not process the log file when the polling interval ends, at startup, or when the Logfile Entry policy is distributed.

  HPOM treats a log file as new if the inode (on UNIX) or the creation time (on Windows) has changed. If the log file is new, the log file encapsulator processes the entire log file.

  If a log file displays again with the same inode or the same creation time, it will be processed as if it was never removed. This is the case with, for example, system log files.

  Log files are *not* considered new in the following cases:

- File is copied over existing file.

  If a file is copied over an existing file, for example with `cp /tmp/xxx /tmp/logfile`, the inode is still the same and the log file encapsulator reads the file from the last position.

- Echo arguments into a file.

  If you echo text into a file with, for example, `echo "xxx" > /tmp/logfile`, the inode is still the same and the log file encapsulator reads the file from the last position.

- **Message Defaults**

  Message defaults enable you to enter default attributes for messages filtered into HPOM by the Logfile Entry policy.

- **Other Options**

  Other options include instructions, message correlation options, and pattern-matching and message stream interface options.

To define the Logfile Entry policies, you need to make updates in the policy body. For details, see "Policy Body Grammar" on page 253.

Monitoring Log Files on Nodes

Note: If you do not use the `NODE` keyword in the Logfile Entry policy, HPOM uses the node on which the log file encapsulator is running. In a clustered environment with HP Operations Agents running on cluster nodes, you can specify a different node on which to monitor the log file.

Specify a different node when you modify log files located on NFS-mounted file systems, or if you have copied the log files from other remote nodes to the system where the HPOM log file encapsulator is running.

Monitoring Log Files on External Nodes

When an event occurs on an external node, the log file encapsulator is not automatically informed. To monitor log files from external nodes, choose between the following alternatives:

- You can use the Network File System (NFS) to mount the file system to a specified directory on the node running HPOM. You must then configure the log file encapsulator to monitor the log file on the mounted file system, as it would any other local file.
- You can configure the external node to copy a defined log file, or extracts from a log file, to a directory on the host system. This is done automatically, either after a specified time interval, or whenever the log file changes. As HPOM does not support this feature, the copy operation must be performed by an external facility. The log file encapsulator tracks the local copy of the file, and processes new entries after they are copied.

The corresponding Logfile Entry policies must be configured so that the HPOM operator knows which system originated the log file, or which event triggered the message. For information on policy body grammar, see "Policy Body Grammar" on page 253.

Defining Advanced Options for Message Policies

You can also define options for pattern matching, suppressing duplicate messages, and outputting messages to the Message Stream Interface (MSI). These options will be used as default for any new policies you add. They do not change the behavior of existing policies.

Specifying Conditions for Messages

You can specify conditions for messages by editing the policy body of the corresponding policy. For details on policy body grammar, see "Policy Body Grammar" on page 253.

LogFile Entry Policy Body Example

The following example monitors an application log file. The file is checked every 60 seconds and the log file encapsulator checks the contents of the file from the position where it last finished checking (keyword `FROM_LAST_POS`). File is opened just before and closed immediately after each reading (keyword `CLOSE_AFTER_READ`). Messages that have the word "`missing`" will be suppressed, messages with word failure will have their severity set to `Critical`, and all other messages will be forwarded to the server (this is achieved by using `FORWARDUNMATCHED` keyword). All messages will have their application attribute set to "`App`" and message group set to "`AppLog`", and all unmatched messages will have severity `Unknown`.

```
LOGFILE “Application log”
                DESCRIPTION “Logfile for Application”
                LOGPATH “/opt/App/log/logfile.txt”
                INTERVAL “60s”
                FROM_LAST_POS
                CLOSE_AFTER_READ
                SEVERITY Unknown
                APPLICATION “App

MSGGRP “AppLog”
FORWARDUNMATCHED

SUPPRESSCONDITIONS
                DESCRIPTION “App messages to be ignored”

                CONDITION
                                TEXT “<*> missing<*>”

MSGCONDITIONS
```

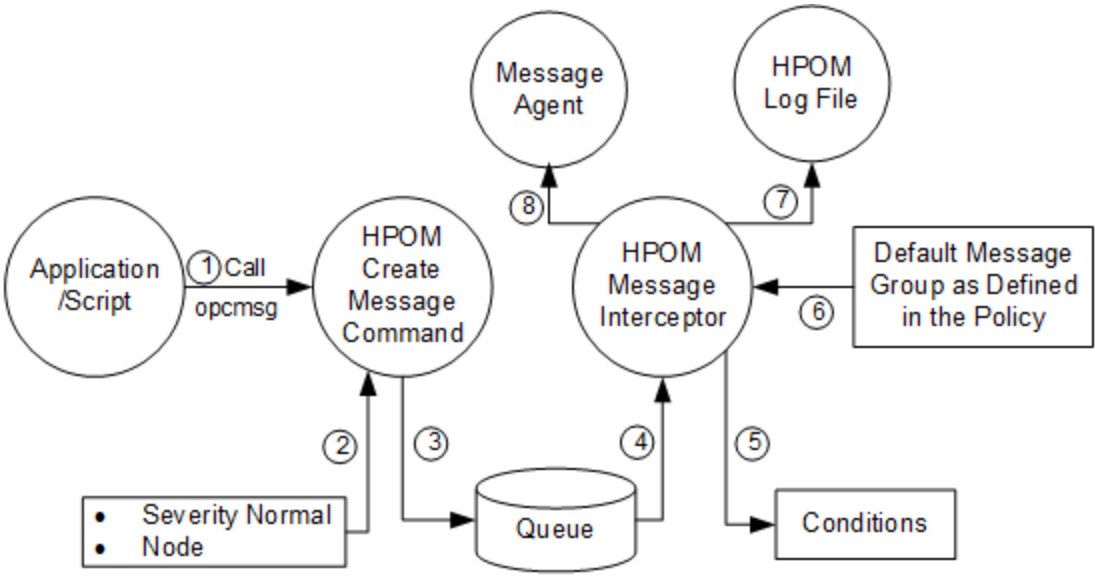
```
DESCRIPTION “App messages to be ignored”

CONDITION
                TEXT “<*> failure<*>”

SET
                 SEVERITY Critical
                 TEXT “<$MSG_TEXT>”
```

HPOM Message Interface

With the HPOM message interface command `opcmsg(1)` and application programming interface (API) `opcmsg(3)`, you can enable existing applications to send messages directly to HPOM.

Figure 30 shows how the HPOM message interface intercepts, filters, reformats and displays HPOM messages in the browsers.

Figure 30: HPOM Message Interface



Step 7 is performed only if local logging is active.

Step 8 is performed only if it is matched by a message condition.

For more information about the `opcmsg(1|3)` command API, see the manpages.

You can also define options for pattern matching, suppressing duplicate messages, and outputting messages to the Message Stream Interface (MSI). These options are used as defaults for any new policies you add. They do not change the behavior of existing policies.

Messages from Threshold Monitors

In HPOM, a message can be generated whenever specified threshold values are met or exceeded. Because you may not want to create a message for a single short-term peak, HPOM enables you to define a time period over which the monitored value must exceed the threshold before generating a message.

Note: A set duration (for example, three minutes) does not necessarily mean that the monitored value exceeds the threshold throughout the whole period. A message is generated when all samples collected during the polling interval have exceeded the threshold.

Starting Corrective Actions in Response to Messages

You can start corrective actions immediately by configuring automatic or operator-initiated actions as responses to the message. You can define monitors that respond to existing problems, as well as monitors that respond to developing problems. As a result, you can use monitoring as both a proactive and a reactive tool.

Integrating Monitoring Programs or Utilities

You can integrate new or existing monitoring programs or utilities, then specify minimum or maximum thresholds. You specify a polling interval that directs HPOM to start the monitor. The results of the monitor program are read by HPOM and compared with the threshold limits you have defined.

For example, you can integrate the UNIX utility `who(1)` to check how many users are logged on, or `df (1M)` to check the number of free disk blocks. The result of the script is compared to a threshold limit you define, and a message is generated if the threshold is exceeded.

By setting a threshold beneath the maximum acceptable limit, you warn the operator before performance exceeds the absolute limit. In this way, you can manage thresholds proactively by starting corrective actions before problems affect users.

To find out how to configure a threshold monitor policy, see "Integrating a Threshold Monitor " on page 196.

How the Monitor Agent Works

The HPOM monitor agent supports the following types of monitors:

- **Program Monitors**

  The monitoring scripts and programs that you provide are invoked by the monitor agent in the configured polling interval. The monitor agent checks the success of the scripts and programs by reading the exit value. If the exit value is not equal to zero, the monitor agent sends a message to the message agent.

  The monitoring scripts or programs collect the current value of the monitored object. The value is sent to the monitor agent through the `opcmon` application program interface (API) or through the command interface provided by HPOM. The monitor agent checks the value against the configured threshold. If the threshold is exceeded, the monitor agent sends a message.

  The program monitor is also used to monitor Windows objects. For details, see the *HPOM Administrator's Reference*.

  In addition, the program monitor is used to integrate metrics collected by the embedded performance component. For details, see "Monitoring Performance Metrics " on page 191.

- **MIB Object Monitors**

  You can monitor MIB objects by using the `SNMP Get` request functionality. The monitor agent checks the returned value against the configured threshold.

  > **Note:** By default, the community `public` is used for SNMP queries. If the MIB object resides in another community, the community name must be defined by using the `ovconfchg` command-line tool on the HTTPS-based managed node where the MIB monitoring takes place.

  The syntax for defining the community name is as follows:

  ```
  ovconfchg -ns eaagt -set SNMP_COMMUNITY <community>
  ```

  In this instance, <*community*> is the community for which the snmpd is configured.

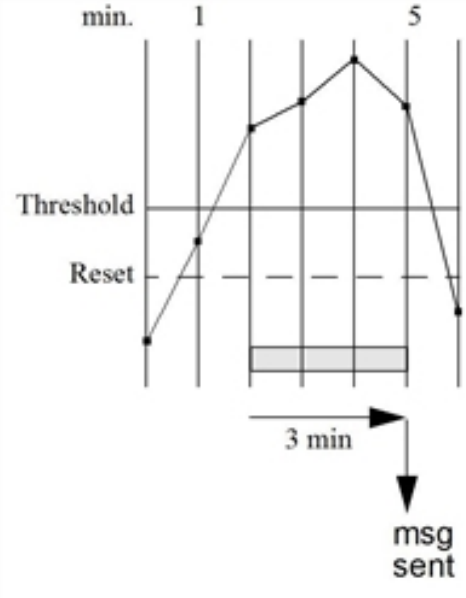
- **External Monitors**

  External monitors are the same as program monitors, except that external monitors are not invoked by HPOM. External monitoring is triggered by calls to `opcmon`. The first time the monitored value exceeds its threshold, the timer is started. Or, if the duration is not specified, a message is generated. If all subsequent values reported by `opcmon` within the specified interval exceed the specified threshold, a message is sent. The monitored value does not necessarily exceed the threshold throughout the whole period, but specifically when each sample is collected.

Monitoring Program or MIB Objects with Polling Intervals

Figure 31 shows when messages are generated after polling intervals for program or MIB object monitors have been defined. The first time the monitored value exceeds the threshold, the timer starts counting. Each time the value is rechecked and still exceeds the threshold, the counter is incremented and compared with the specified duration. When the duration is reached, a message is generated.

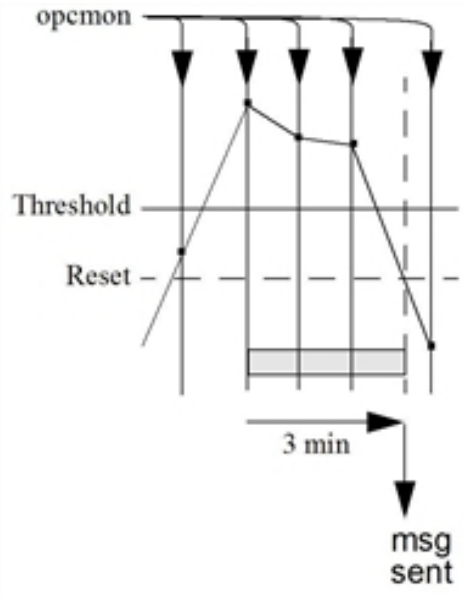
Figure 31: Program/MIB Monitoring with Polling Intervals



Monitoring External Objects with opcmon

Figure 32 shows that a message is generated after the threshold was reached or exceeded by an external application within a time frame of three minutes.

Figure 32: External Monitoring Using opcmon



Monitoring Performance Metrics

Performance metrics are collected by the embedded performance component that is part of the HP Operations Agents. The embedded performance component collects performance counter and instance data from the operating system.

The collected values are stored in a proprietary persistent data store from which they are retrieved and transformed into presentation values. The presentation values can be used by extraction, visualization, and analysis tools such as HP Reporter and HP Performance Manager />. You cannot extract and export, view, or aggregate the data directly on the managed node.

With HPOM, you can set up threshold monitors for the performance metrics collected by the embedded performance component.

For a complete list of supported platforms for the HP Operations Agent and its embedded performance component, see the *HPOM Installation Guide for the Management Server*.

Performance Metrics

The embedded performance component provides the following sets of metrics:

- **Platform-generic Metrics**

  Metrics available on all supported platforms. These metrics are basic metrics that can be used to answer most questions about a system's global configuration, CPU, disk, swap, and memory usage.
- **Typical Metrics**

  Additional metrics on each of the supported platforms. Although these metrics vary by platform, they are available on most platforms and are generally useful for drill down and diagnosis on a particular system.

The metrics that are currently available with the embedded performance component are described in more detail on the following web page:

- Standard connection:

  `http://<management_server>:8081/ITO_DOC/<lang>/manuals/EmbedPerfAgent_Metrics.htm`
- Secure connection:

  `https://<management_server>:8444/ITO_DOC/<lang>/manuals/EmbedPerfAgent_Metrics.htm`

In this instance, *<management_server>* is the fully qualified hostname of the management server, and *<lang>* stands for your system language, for example, `C` for the English environment.

Setting Up Performance Thresholds

Use threshold monitor policies to access data collected by the embedded performance component. The monitor type must be set to `Program` and the following syntax must be used in the `Monitor Program or MIB ID` field:

`OVPERF\\<data source>\\<object>\\<metric>`

This syntax includes the following parameters:

<data source>	Identifies the data source. When collecting metrics from the embedded performance component, *<data source>* must be set to `CODA`.
<object>	Identifies the name of the object class to be monitored. The performance component collects the following object classes: • Global (object name: `GLOBAL`) • CPU (object name: `CPU`) • Network interface (object name: `NETIF`) • File system (object name: `FS`) • Disk (object name: `DISK`)

<metric>	Identifies the metric to be collected. For a list of metrics available for each object class, see the following web page:
	• Standard connection:
	`https://<management_server>:8081/ITO_DOC/<lang>/manuals/EmbedPerfAgent_Metrics.htm`
	• Secure connection:
	`https://<management_server>:8444/ITO_DOC/<lang>/manuals/EmbedPerfAgent_Metrics.htm`

```
ADVMONITOR "Outpacketrate"
DESCRIPTION "Get the global metric GBL_NET_OUT_PACKET_RATE"
INTERVAL "5m"
INSTANCEMODE SAME
MAXTHRESHOLD
SEVERITY Warning
PROGRAM "Source"
DESCRIPTION ""
MONPROG "OVPERF\\CODA\\GLOBAL\\GBL_NET_OUT_PACKET_RATE"
```

The performance component constantly collects all platform-generic and typical metrics. The collection interval is by default five minutes and cannot be changed. The data is kept in the data store for up to five weeks. When the database is full after five weeks of data have been collected, the oldest data is rolled out one week at a time and deleted.

For more information about troubleshooting the embedded performance component, see the *HPOM Administrator's Reference*.

Selecting Variables to Monitor

Which variables you select to monitor depends on your environment, the monitors that you currently use, and the parameters that should be controlled. You can integrate existing and custom monitoring programs. You can also determine the variables to monitor by reviewing existing monitors and examining important environmental parameters.

For example, check the following:

- Which monitors are currently used?
- Which monitors are used daily, weekly, or monthly?
- Which custom monitors have you generated?
- Which monitors are used by the operating system or applications within the environment?

You should also check which parameters should be monitored.

For example, you can review the following:

- Which parameters can be monitored?
- Which parameters are critical?
- Which parameters can have a threshold applied?
- Which parameters should issue warning before limits are exceeded?

Selecting a Threshold Type

You can set either a minimum or maximum threshold for a monitor:

- **Minimum Threshold**

  A message is generated if the monitored value equals or drops below the minimum acceptable limit. For example, you can use a minimum threshold for the `df` monitor (free disk blocks). HPOM generates a message when the number of free disk blocks drops below the threshold you define.
- **Maximum Threshold**

A message is generated if the monitored value equals or exceeds the maximum limit. For example, you can use a maximum threshold for the `who` monitor for the number of users. HPOM generates a message when the number of users exceeds the threshold you define.

Selecting a Message Generation Policy

The following three message generation policies are available for use with threshold monitors:

- Message Generation with Reset
- Message Generation without Reset
- Continuous Message Generation

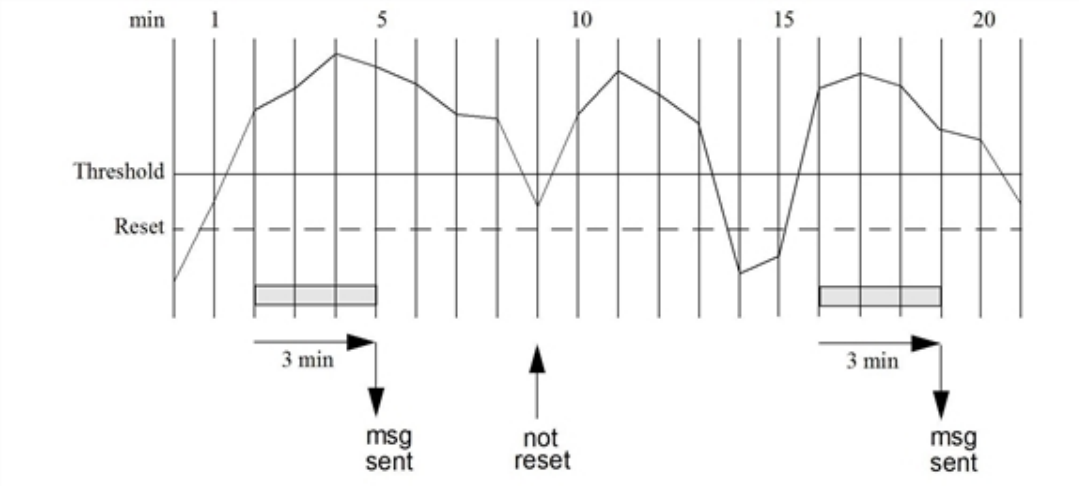
Note: In all three cases, HPOM recognizes threshold crossing if the value of the monitored objects meets or crosses the threshold at polling time.

The following examples demonstrate the differences between each of these settings. For each example, a polling interval of one minute and a duration of three minutes is assumed.

Message Generation with Reset

Message generation with reset is shown in Figure 33. In the second polling (2 min), the value exceeds the threshold, and the timer is started. Three minutes later, the threshold is still exceeded and a message is sent. When the value dips below the reset level, the next occurrence of threshold violation resets the timer, and the cycle is resumed.

Figure 33: Message Generation with Reset

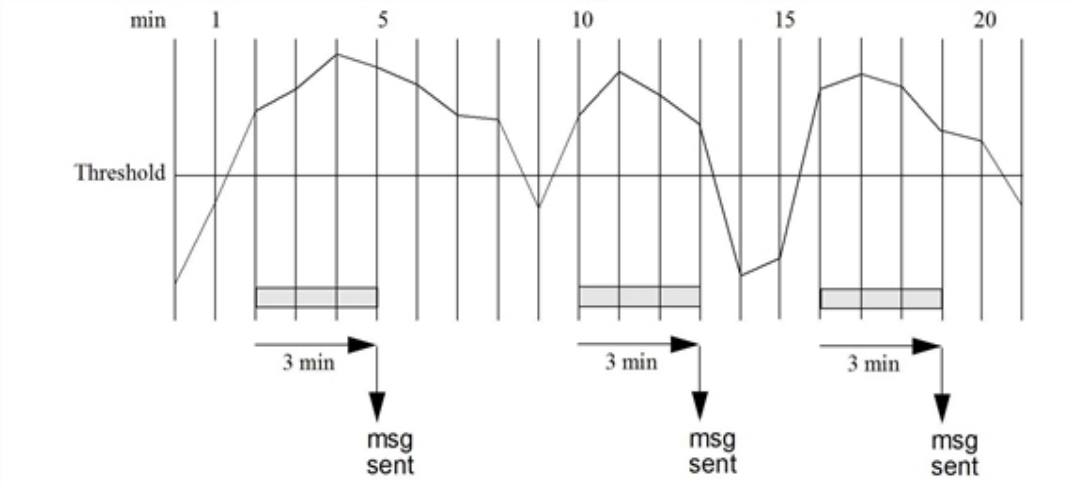


Message Generation Without Reset

Message generation without reset means that there is no separate reset value. The reset value is the same as the threshold value.

Message generation without reset is shown in Figure 34. At the second polling (two minutes), the value exceeds the threshold, and the timer is started. After three minutes, the threshold is still violated, and a message is sent. When the value dips below the threshold, the next occurrence of threshold violation resets the timer, and the cycle is resumed.

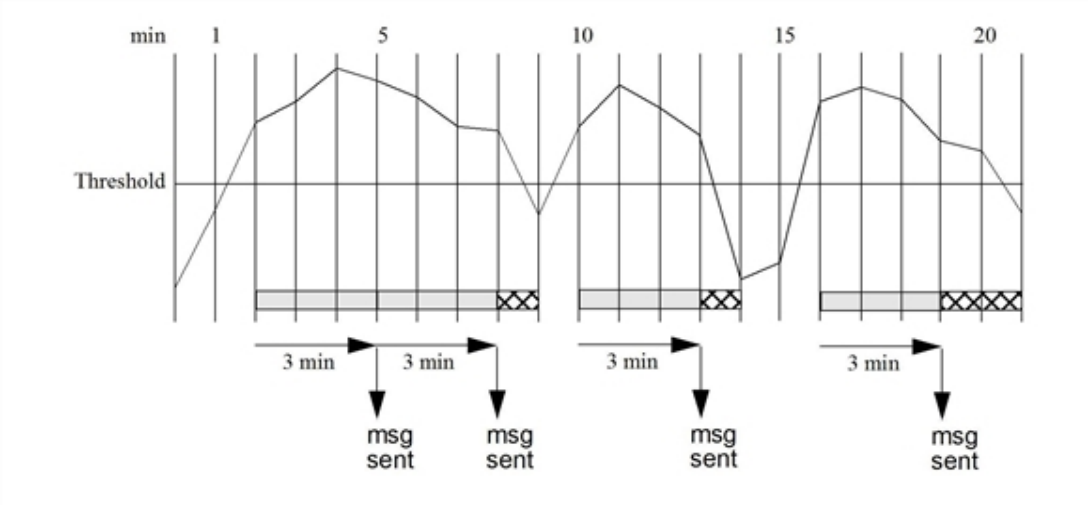
Figure 34: Message Generation Without Reset



Continuous Message Generation

An example of message continuous message generation is shown in Figure 35. At the second polling (two minutes), the value exceeds the threshold, and the timer is started. After three minutes, a message is sent, and the timer is immediately reset. This continues until the value meets or drops below the threshold value. When the monitored value again exceeds the threshold, the timer is started, and the cycle is resumed.

Figure 35: Continuous Message Generation



Short-Term Peaks

Because it may not be reasonable to create a message when a threshold is exceeded only for a short time, HP Operations allows you to define a minimum time period over which the monitored value must exceed the threshold before generating a message. For as message to be sent, the value must be greater than the threshold each time the value is measured during a duration that you select. Select a value that is a multiple of the policy polling interval.

For example, if the polling interval is 2m (two minutes), set the short-term peak duration to 4m, 6m, 8m, or 10 min, and so on. If the duration is set to 0 or the value is not specified, an alarm is generated as soon as HP Operations detects that the threshold is equaled or crossed.

Integrating a Threshold Monitor

You can define a threshold monitor by using a policy in a way that is similar to the way you define a log file. You can generate new monitors, and modify or copy existing ones.

Integrating a New Threshold Monitor

To integrate a new threshold monitor, follow these steps:

1. **Prepare the threshold monitor data for deployment.**

   Instrumentation data planned for deployment is placed in the `instrumentation` directory on the HP Operations management server, at the following location:

   `/var/opt/OV/share/databases/OpC/mgd_node/`

Note: If no categories are created, the data from the monitor directory is anyway deployed. Although the category-based distribution method is recommended, you can chose to distribute your monitor from this directory. If you do so, you must place the monitor on the management server in a directory specific for each managed node platform to which it will be distributed. For example, monitor programs or scripts for HP-UX 11i managed nodes are located on the management server at:

`/var/opt/OV/share/databases/OpC/mgd_node/customer/hp/ipf32/hpux1100/monitor`

All distribution methods, the administration tasks related to them (including category management), and the instrumentation data directory structure, are described in the *HPOM Administrator's Reference*.

a. In the `instrumentation` directory, place the monitor program or script properly (for each managed node platform to which you want to distribute the data) within the category related to your data. If there is no such category, you can create and assign it to your policy, and/or managed node.

2. **Distribute the threshold monitor to the managed nodes.**

   To do this, use `opcragt` command-line utility (see *opcragt.1M* manual page for usage information).

   On HPOM managed nodes, all deployed instrumentation data (category-based instrumentation, as well as the `monitor|actions|cmds` files) is located in the following directory:

   `/var/opt/OV/bin/instrumentation`

3. **Configure the threshold monitor policy.**

   Use the `opcpolicy` command-line tool to upload a threshold monitor policy. For the correct syntax of threshold policy body, see "Policy Body Grammar" on page 253. Each policy defines a monitor, including automatic actions or operator-initiated actions to be started if the threshold is exceeded.

   Note: If you have chosen a category-based method for distributing your threshold monitor, make sure that the appropriate categories are assigned to the policy.

4. **Configure conditions for the threshold monitor policy.**

   The `MSGCONDITIONS` section of the policy body determines whether the matched condition produces a message that is sent to the Java GUI message browser. You can further filter the messages by using the `SUPPRESSCONDITIONS` sections. For policy body grammar, see "Policy Body Grammar" on page 253.

   Note: If you have more than one condition for a monitor, the order of the conditions is important.

   Order the conditions according to size of the threshold value:

- **Maximum Threshold Type**

  List the condition with the highest threshold first, and the condition with the lowest value last.

- **Minimum Threshold Type**

  List the condition with the lowest threshold first, and the condition with the highest value last.

Ordering conditions enables the state-based browser configuration to automatically acknowledge a previous message from the same monitor. For information about state-based browsers, see "State-Based Browsers " on page 169.

Configuring a Threshold Monitor

You can configure a threshold monitor policy by editing the policy body of the `ADVMONITOR` policy. For details on policy body grammar, see "Policy Body Grammar" on page 253.

Monitor Threshold Policy Body Example

In this example, a user is running a custom filesystem utilization calculation script `fs_util_mon.sh`, which calls the `opcmon` command-line tool to pass the calculated value back to the monitor agent, naming it `extra_util` (passed as a parameter to the script).

The monitor agent will produce a message when the filesystem utilization is higher (`MAXTHRESHOLD`) than the configured one (`THRESHOLD`). However, messages will not be sent again until the utilization falls below the value configured with the `RESET` keyword. All messages will have severity set to `Warning`, application field set to "`Filesystem`", object to "`/extra`" and message group to "`Disks`". No other messages will be sent to the management server apart from ones matching the configured condition.

```
ADVMONITOR “extra_util”

       DESCRIPTION “Monitor /extra filesystem utilization”
       INTERVAL “5m”
       INSTANCEMODE SAME
       MAXTHRESHOLD
       SEVERITY Warning
       PROGRAM “Source”
                      DESCRIPTION “Universal FS usage monitoring script”
                      MONPROG “fs_util_mon.sh /extra extra_util”
       MSGCONDITIONS
                      DESCRIPTION “Monitor /extra FS util”
                      CONDITION

                      THRESHOLD 85.00
                                     RESET 80.00
                                     SETSTART
```

```
                    SEVERITY Warning
                                    APPLICATION “Filesystem”
                                    MSGGRP “Disks”
                                    OBJECT “/extra”
                                    TEXT “Filesystem /extra utilization <$VALUE>
                                    exceeds configured threshold <$THRESHOLD>”
                                    AUTOACTION “du -k /extra” ANNOTATE
```

Default Threshold Monitors

HPOM provides a set of default threshold monitors. For details, see the HP Operations Agent documentation.

To Set Conditions for Advanced Monitoring

You can set conditions for threshold monitor policies to monitor multiple instances of a single monitored object.

To do set conditions for threshold monitors, follow these steps:

1. Use the `opcmon(1)` command with the option `-object` to submit the name of the monitored object to the monitor agent.

   The option `-option` gives passes additional information to the monitor agent. This information can be used in the message text or referenced in corrective actions.

   HPOM compares the name against the pattern set with `OBJECT` keyword in the advanced monitor policy body.
2. Use the HPOM pattern-matching language to match the incoming object pattern.

   For more information, see the *opcmon(1)* manpage.

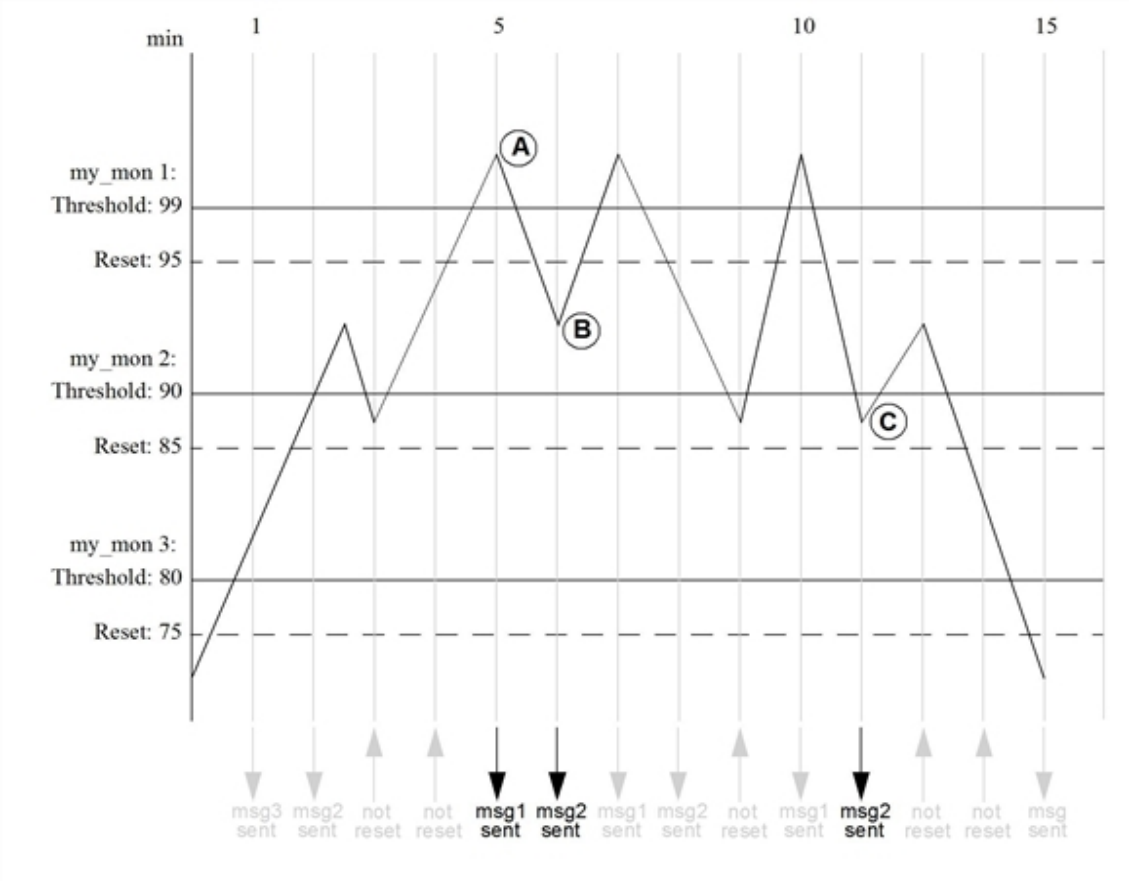
For an example of how you can monitor disk utilization in different file systems, see "Examples of Threshold Monitor Conditions" on page 201.

Threshold Monitoring with Multiple Conditions

When setting up multiple conditions with different threshold and reset values for a monitored object in one policy, you receive messages whenever the monitoring range of another condition is reached.

Consider the example in Figure 36. The figure shows three conditions, each with a maximum threshold and a reset.

Figure 36: Message Generation with Multiple Conditions Using Reset



At the fifth polling (five minutes), the value exceeds the threshold of condition `my_mon 1` (threshold value = 99) and a message is sent **(A)**. One minute later, the value drops below the reset value of condition `my_mon 1` (reset value = 95). Since it exceeds the threshold value of condition `my_mon 2` (threshold value = 90), another message is sent **(B)**.

This means that reaching a condition's monitoring range also generates a message, although the value does not drop below the reset value of that condition.

After 11 minutes, the value drops below the threshold value of condition `my_mon 2` (threshold value = 90) but still exceeds the reset value of 85. Another message is generated **(C)**. The original message text of this message reports `Reset value still exceeded` because only the threshold value was crossed, not the reset value. This message is generated only when the monitored value drops below the threshold value. When the monitored value exceeds the threshold value, the reset value is also exceeded and the message is not generated.

In this example, you receive many messages for the same monitored object. To reduce the number of messages in the browser, configure your conditions so that messages are acknowledged automatically. For more information, see "State-Based Browsers " on page 169.

Examples of Threshold Monitor Conditions

The following examples show how you can use threshold monitor conditions to monitor the disk space in the `/var` and `/file` systems with the `disk_util` threshold monitor policy. These examples assume that you have written a shell script that determines and reports the disk utilization in each file system.

Message Condition No. 1

Object Pattern:	`/var`	
Threshold:	`90`	
Reset:	`85`	
Duration:	`3`	
Set Attributes:	Severity:	`warning`
	Message Group:	`OS`
	Text:	`Utilization of /var file system (<$VALUE>) is greater than (<$THRESHOLD>)%.`

Message Condition No. 2

Object Pattern:	`^/`	
Threshold:	`95`	
Reset:	`90`	
Duration:	`3`	
Set Attributes:	Severity:	`critical`
	Message Group:	`OS`

	Text:	`Utilization of root file system (<$VALUE>) is greater than (<$THRESHOLD>)%.`

SNMP Traps and Events

The HPOM event interceptor (`opctrapi`) is the message interface for feeding SNMP traps into HPOM.

For detailed information about intercepting SNMP traps and events, see the *HP Network Node Manager i Software—HP Operations Manager Integration Guide*.

Defaults for Intercepting Traps and Events

By default, HPOM intercepts SNMP traps and events as follows:

- **From Application**

  From any application sending traps to the `opctrapi` daemon running on the HP Operations management server.
- **On Managed Nodes**

  On all managed nodes where the trap daemon (`ovtrapd`) is running.
- **On Managed Node Platforms**

  In direct port access mode on selected managed node platforms.

  Intercepting events directly on the managed node enables you to process messages locally, which enhances performance. Automatic actions, for example, can be triggered and executed directly on the node or in the subnet, instead of being first forwarded to the management server.

Adding SNMP Trap Policies

When setting up an SNMP trap policy, you can assign any number of trap policies to the HP Operations management server or the managed nodes where the HPOM event interceptor is supported.

You can configure a trap policy by editing the policy body of the `SNMP` policy. For details on policy body grammar, see "Policy Body Grammar" on page 253.

Trap Interceptor Policy Body Example

This sample catches Cisco linkDown trap (`.1.3.6.1.4.1.9.2.0`) produced by Cisco routers. When the trap is caught, message is produced with its severity set to `Warning`. Notice that the

enterprise is separate from the generic trap. Variables <$1> and <$2> are a part of the trap (link index and description, respectively).

```
SNMP “Sample trap interceptor template”
               DESCRIPTION “This is catches Cisco linkDown trap”
               CONDITION
                               $G 2
                               $e “.1.3.6.1.4.1.9”
               SET
                               MSGTYPE “Cisco_Link_Down”
                               SEVERITY “Warning”
                               OBJECT “<$2>”
                               TEXT “Interface <$1> down”
```

Example of an SNMP Trap Condition

HP Data Protector issues the following SNMP trap when a backup starts and a syntax error is detected in the worklist file:

```
snmptrap idriss1 1.3.6.1.4.11.2.3.2 15.232.
117.22 58916871  6
1.3.6.1.4.11.2.15.2.0 Integer 1
1.3.2.1.4.11.2.15.3.0 OctetString doghouse.bbn.hp.com
1.3.2.1.4.11.2.15.4.0 OctetString
”HP Data Protector:[Error](Worklist Syntax)Can’t open
worklist ‘/etc/omni/work’ Status:Critical”
1.3.2.1.4.11.2.15.5.0 OctetString ”Critical”
1.3.2.1.4.11.2.15.6.0 OctetString ”dp”
```

The SNMP trap policy needs a condition with the following definition:

Node	doghouse	
Enterprise ID	1.3.6.1.4.11.2.3.2	
Generic Trap ID	6	
Specific Trap ID	58916871 (SNMP status event)	
Variable Bindings	Application Type:	1(agent)
	Object ID:	mailhouse.bbn.hp.com.omniback

	Event Description:	`HP Data Protector: [Error](Worklist Syntax)Can’t open worklist ‘/etc/omniback/work’ Status:Critical`
	Trap-specific Data:	`critical`
Set Attribute	Severity:	`critical`
	Message Group:	`print services`
	Text:	`Error in HP Data Protector: <text>`

Filtering Internal HPOM Error Messages

Internal HPOM error messages can be extracted from or filtered out of the internal Message Stream Interface (MSI) so that automatic and operator-initiated actions may be attached, and the message treated as if it were a normal, visible HPOM message. You can enable this functionality on the managed node and on the management server. Depending on where the functionality is enabled, all internal HPOM messages are sent back to the local message interceptor, either on the HP Operations management server or on the managed node. There they are read and handled in the same way as any other HPOM message.

Management Server

On the management server, use the `ovconfchg` command-line tool. Enter the following:

```
ovconfchg -ovrg <OV_resource_group> -ns opc -set OPC_INT_MSG_FLT TRUE
```

In this command, *<OV_resource_group>* is the name of the management server resource group.

Managed Nodes

On HTTPS-based managed nodes, use the `ovconfchg` command-line tool. Enter the following:

```
ovconfchg -ns eaagt -set OPC_INT_MSG_FLT TRUE
```

Set up at least one condition for internal HPOM error messages in the `opcmsg (1/3)` policy (using message group `OpC`). Then set the `SUPP_DUPL_IDENT_OUTPUT_MSG` keyword in the policy body.

Event Correlation in HPOM

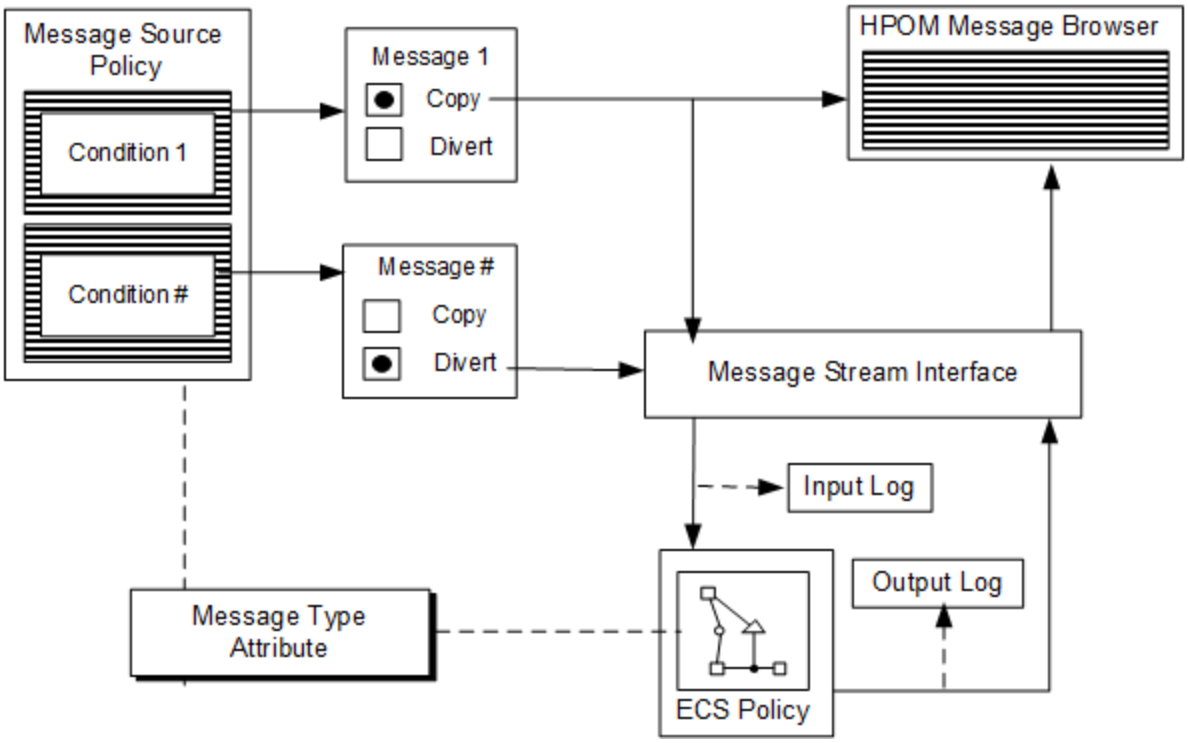
In general terms, messages that are generated by the conditions defined in typical HPOM message source policies are used as an input by the HPOM event correlation (EC) policies. The event correlation policies then process the HPOM messages and, where appropriate, generate new messages, which

are displayed in the Java GUI message browser. In the context of HPOM, a correlation circuit is viewed and treated as a policy.

How Event Correlation Works

Figure 37 shows how the event correlation policy works in the context of HPOM. The HPOM message source policy enables you to specify which condition generates a message. The policy also enables you to determine whether the generated message is copied or diverted to the Message Stream Interface (MSI). From the MSI, the generated message can be passed to and processed by the event correlation policy. HPOM allows you to copy, rather than divert, messages to the correlation engine. As a result, critical messages are not delayed or lost in the correlation process. This feature is particularly useful for troubleshooting.

Figure 37: Logical Event Correlation Flow in HPOM



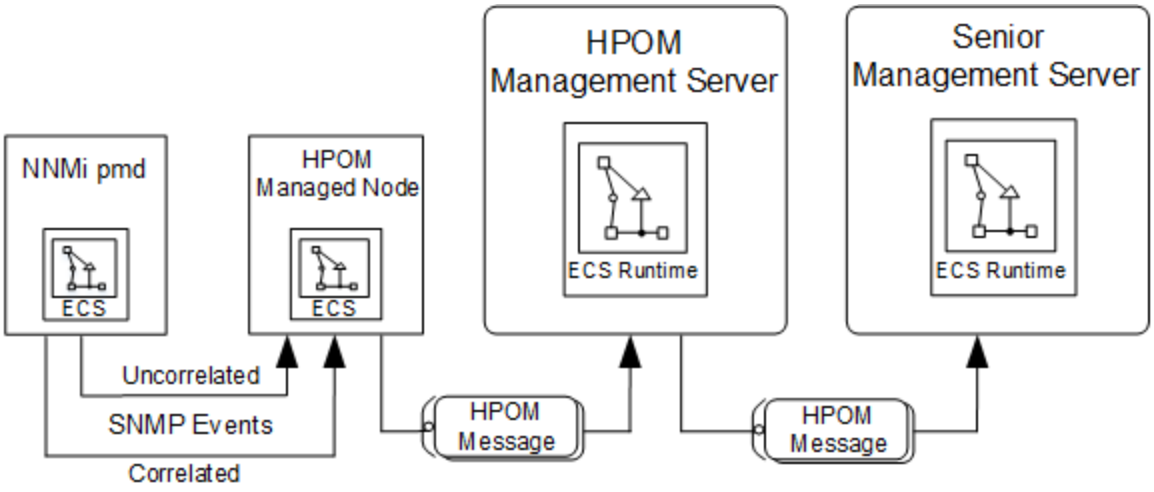
The event correlation policy determines which event correlation circuit the messages pass through. It does so by matching the `Message-type` attribute specified in the message condition with the message attribute specified in the `Event-type` field of the `Input` node (port) of the event correlation circuit.

For more information about how to set up event correlation in HPOM, see the *HPOM Administrator's Reference*.

Where to Correlate Messages

It is important to consider the advantages and disadvantages of where the correlation should take place in an HPOM environment. In a standard environment, you can decide to correlate messages on the managed node, on the management server, or on both. However, your choice widens in larger environments that use HPOM flexible-management configuration. In such environments, several layers of management servers are configured hierarchically and you should consider the relationship between the various levels of management servers. The correlation can be also set up in the NNMi domain to significantly reduce the amount of SNMP-related messages intercepted by HPOM event interceptor. For the possible correlation approaches, see Figure 39.

Figure 38: Configuring Correlation in HPOM



Generally, the earlier the correlation is performed the better, because the load down-stream is reduced and fewer messages arrive in the message browser. Before you assign and distribute event correlation policies to the management server or to the managed nodes, you need to consider where the correlation should take place:

- **Before arriving on the managed node**

  You can greatly reduce the number of events intercepted by HPOM by using the correlation circuits of NNMi to correlate the events before they reach HPOM.
- **On the managed node**

  You can reduce the number of messages passed to the management server by correlating messages on the managed node. As a result, both the load on the management server and the general amount of network traffic are reduced.
- **On the management server**

  You can filter out similar or related messages coming from different managed nodes by correlating messages on the HP Operations management server.

- **In the flexible management environment**

  The relationship between managed nodes and the management server in the HPOM environment can be extended to the relationship between the management servers in the management hierarchy.

Correlating Messages from Different Sources

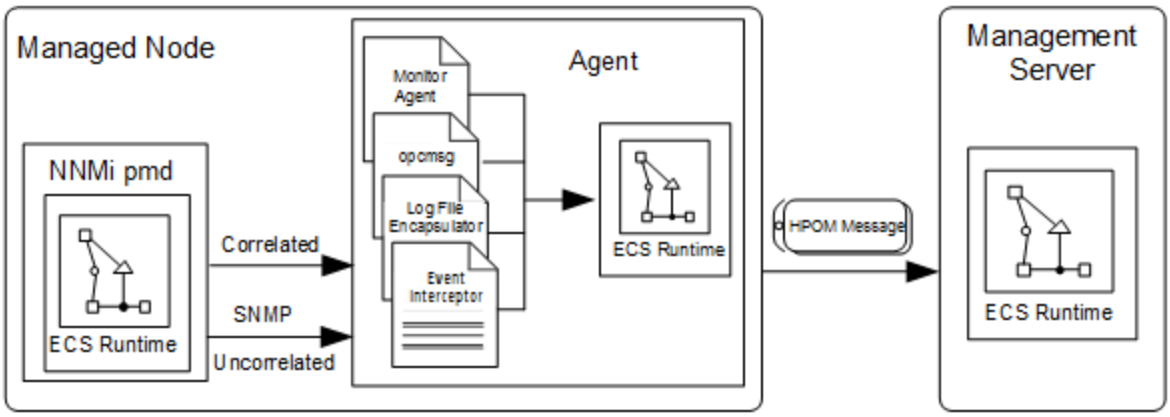
You do not need to restrict correlation to individual sources. Correlating messages from different sources within HPOM has a number of advantages.

You can correlate messages from the following sources:

- SNMP traps
- `opcmsg`
- Log files
- Monitor agents

For example, messages generated by SNMP and related to a node being down can be correlated with the messages generated by the log file entries related to the servers that are unreachable.

Figure 39: Correlating Events from Different Sources



HPOM Event Interceptor

The HPOM event interceptor is a link between NNMi and HPOM. The HPOM event interceptor taps both the correlated and uncorrelated stream of SNMP events produced by the NNMi postmaster daemon (`pmd`). Where appropriate, the event interceptor generates HPOM messages. The resulting messages may then be passed through the event correlation policy of the HP Operations Agent along with the messages generated by other HPOM sources, such as log files.

Correlating Events in NNMi Before They Reach HPOM

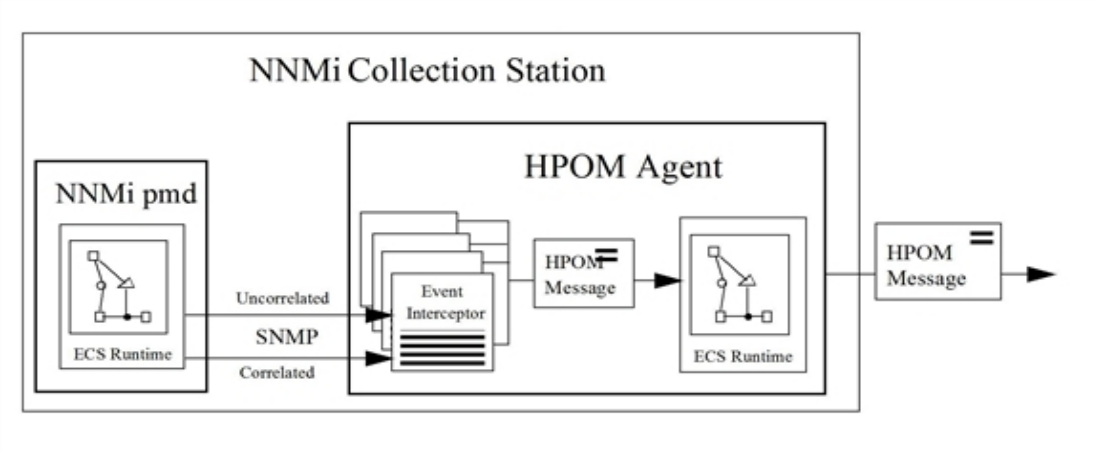
To reduce the overall load on the HP Operations Agent and make it easier for HPOM to perform the correlation, use the correlation circuits of NNMi to correlate the events before they reach HPOM.

For example, you can configure a circuit in NNMi to correlate events that are generated if a router does not respond. When you ensure that the HPOM event interceptor on the NNMi collection station receives only correlated events, you can significantly reduce the number of the generated HPOM messages. These messages could be further correlated by the event correlation policies on the HPOM managed node.

Synchronizing HPOM and NNMi Event Correlation

The event correlation processes of NNMi and HPOM are synchronized, which means that the events discarded by NNMi are also suppressed by HPOM. Similarly, the events acknowledged or deleted by the NNMi circuits are also acknowledged by HPOM automatically. In addition, automatic annotations are added to each message associated with the correlated (and therefore suppressed) events. This functionality is included in the conditions of the SNMP trap policy `SNMP ECS Traps`.

Figure 40: Correlation in NNMi

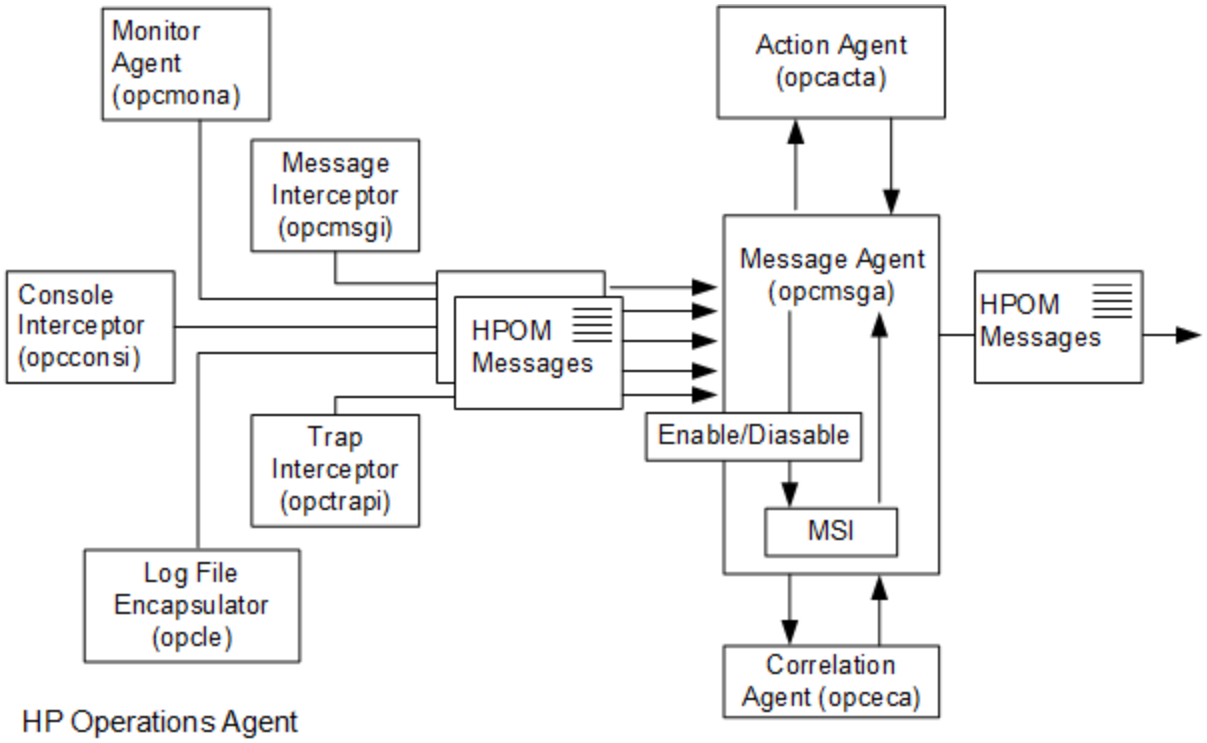


Correlating Messages on Managed Nodes

Using an HP Operations Agent to correlate events on managed nodes significantly reduces network traffic between an agent and the server. The network traffic reduction takes place on all managed nodes on which the event correlation agent is running. Because the CPU load on the management server is reduced, the management server can attend to other problems more efficiently.

Figure 41 shows how messages are generated on a managed node and then processed, and how you can control the flow of messages to the event correlation agent (opceca) by enabling and disabling message output to the agent MSI.

Figure 41: Message Flow on the HPOM Managed Node



The opceca process connects to the agent MSI to enable access to messages from the HPOM agent message stream. The messages are then correlated and sent back to the HPOM message agent. Messages created or modified by opceca appear in the message browser with the message source MSI:opceca. If a message does not match any of the rules and conditions specified in the correlation circuits, it retains its original message source.

If the ECS configuration is available on a managed node, you can check opceca status by using the command opc(r)agt -status. The opceca process is started if the event correlation policy is distributed to the managed node. The process opceca is stopped if no event correlation policy configuration is available on the managed node.

Event correlation policies are assigned to a managed node in the same way as any other HPOM policy.

Note: If you try to load a correlation circuit into a running correlation engine (for example when distributing new or modified event correlation policies) the engine forwards all open messages to the message agent (opcmsga) stops the engine, loads the circuit, and then restarts. The message agent does not send the forwarded messages back to the correlation engine.

Note: Automatic actions are not carried out if they are associated with a message that is discarded in the correlation process and if the `MPI_IMMEDIATE_LOCAL_ACTIONS` keyword is not specified in the policy body. This option is enabled by default and available only if you have both enabled output to the MSI and specified the `MPI_SV_DIVERT_MSG` keyword in the policy body.

If any new action is required, it should be specified and associated with a new message or with modified existing messages during the correlation process.

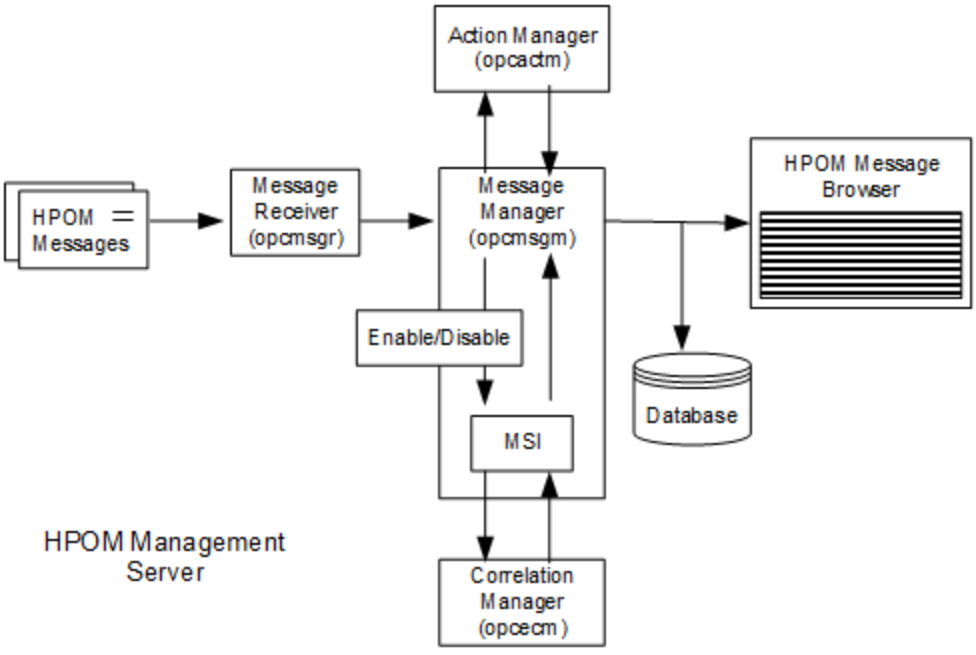
Correlating Messages on the Management Server

Correlating messages on the HP Operations management server enables you to reduce the number of identical or similar messages coming from different nodes regarding the same problem. These messages display in the Java GUI message browser.

Reducing redundant messages is particularly useful in environments with distributed client-server applications and shared network devices, such as printers, backup devices, or NFS file servers. For example, if a database server is temporarily unavailable, you can filter out the similar messages arriving from the managed nodes that lost contact with the database server.

Figure 42 shows the message flow on the management server, and how you can control the flow of messages to the correlation manager (`opcecm`) by enabling and disabling message output to the server MSI.

Figure 42: Message Flow on the HP Operations Management Server



The `opcecm` process connects to the server MSI to enable access to messages from the message stream. The messages are then correlated and written back to the HPOM message manager. Messages created or modified by the correlation process appear in the message browser with the message source `MSI:opcecm`. If a message does not match any of the rules and conditions specified in the correlation circuits, it retains its original message source.

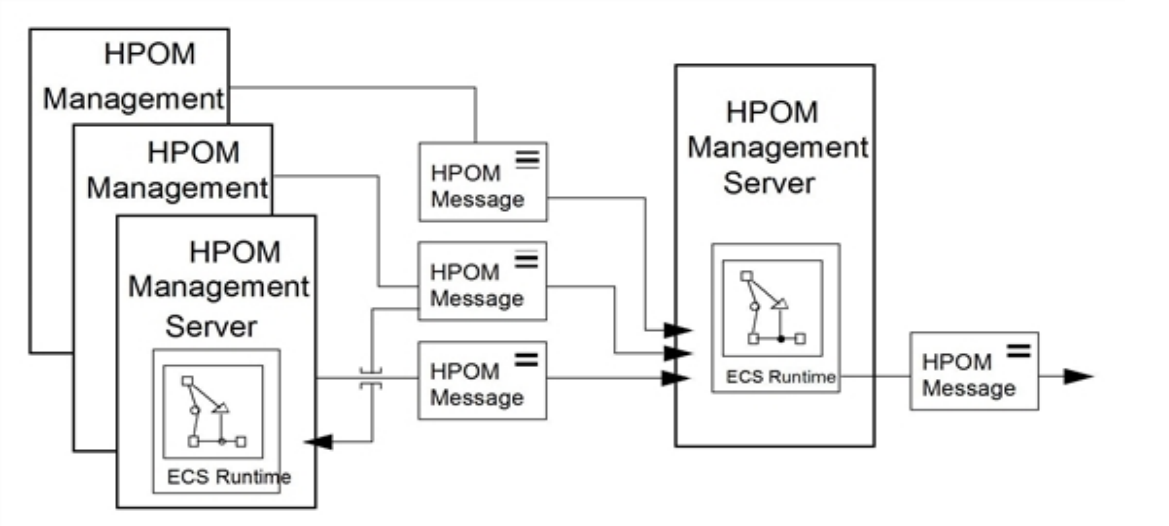
If the ECS configuration is available on the management server, you can check the `opcecm` status with the `opcsv -status` command. The `opcecm` process is started if the event correlation policy is distributed to the management server and stopped if no event correlation policy configuration is available on the management server.

Correlating Messages in Flexible Management Environments

In larger corporate environments where the flexible management configuration features of HPOM are used, message correlation can be more complex, because the relationships between the different levels in the management hierarchy are considered. The relationship between managed nodes and the management server in the HPOM environment can be extended to the relationship between the management servers in the management hierarchy. Management servers can then correlate the messages they receive from the managed nodes, and send a newly correlated stream to the management servers to which they report. Or, the management servers can send an uncorrelated stream of messages to the next management server up the chain, where they get correlated.

To enable this kind of message correlation in a flexible management hierarchy, assign and distribute management server correlation policies to the HP Operations management servers. For information on how to assign and distribute policies, see "Implementing Message Policies" on page 127.

Figure 43: Correlation with HPOM Flexible Management



Accessing External Data

It is often necessary for a correlation circuit to access information from an external source. This external information can include parameters that change the behavior of the correlation circuit. This enables you to modify the circuit operation without recompiling it. For example, you can have a correlation circuit for transient faults, which is applicable for different message types. You can keep the list of message types external to the correlation circuit, so that you can add additional message types without the need to recompile the circuit.

External information may also be required to assist in making correlation decisions or to enhance the information output from the correlation circuit. For example, when an error message is detected, you can query an external database to obtain service level agreement (SLA) information. The SLA details could then be added to the message before it is displayed in the message browser.

There are two methods of accessing external data from within a correlation circuit:

- "Data and Fact Stores " below
- "Annotate Mechanism " on page 216

Data and Fact Stores

The ECS data store and the fact store enable you to separate the environmental aspects of correlation from the rules and the basic logic which are hard-coded into the event correlation policy. For example, you can configure a generic correlation circuit for one HPOM managed node and, with the help of the data store, use the same policy on other managed nodes.

The data store can be used to hold key-value pairs of information. This may include system-specific information related to the event correlation policy and external network details (such as intervals, threshold values, and other constants), if you know that these details may change. For example, you can configure one generic event correlation policy to monitor user switches (by using the `su` command), and then run the policy on different HPOM managed nodes by storing the names of trusted users (which can differ from system to system) in the data store.

The fact store is a data structure that defines relationships. These relationships can, for example, describe the hierarchy of an organization or the relationships between one organization and another, or between services and service providers. For example, assume that three application servers, A, B, and C, are linked to the same database server `DBserver01` and a fourth application is linked to a different database server `DBserver02`. To define these relationships you can use the fact store. If `DBserver01` does not respond for any reason, the information in the fact store can be used to correlate the messages from this database server with the messages from application servers A, B, and C related to the lost contact with `DBserver01`.

Fact and data stores are text files that are loaded into the ECS engine when an EC policy is distributed to a managed node or a management server. They have a `ds` file extension for the data store and an `fs` file extension for the fact store. Note that fact and data store files are not distributed with the EC policy. They must be created manually and copied to the following location before distributing the EC policy:

- On the management server `/var/opt/OV/shared/server/datafiles/policies/ec`
- On the managed node (UNIX)
  `/var/opt/OV/conf/eaagt`
- On the managed node (Windows)
  `C:\Program Files\HP BTO Software\data\conf\eaagt`

Note: When using Composer, the `ovocomposer` script can be used for creating, distributing, and installing fact store files.

Each correlation circuit can access only one fact store and one data store file.

There are two types of fact and data store in HPOM, specific and global. Global fact and data stores can be shared across multiple correlation circuits. Specific fact and data stores can be accessed by a single correlation circuit only.

- *Specific Data Stores and Fact Stores*

  Each correlation circuit has a compiled ECS circuit file (`.eco`) associated with it. The name of the compiled circuit file is automatically generated when the correlation circuit is created or modified, and has the following format: `<id>.eco` (for example, `EAAAa03015.eco`).

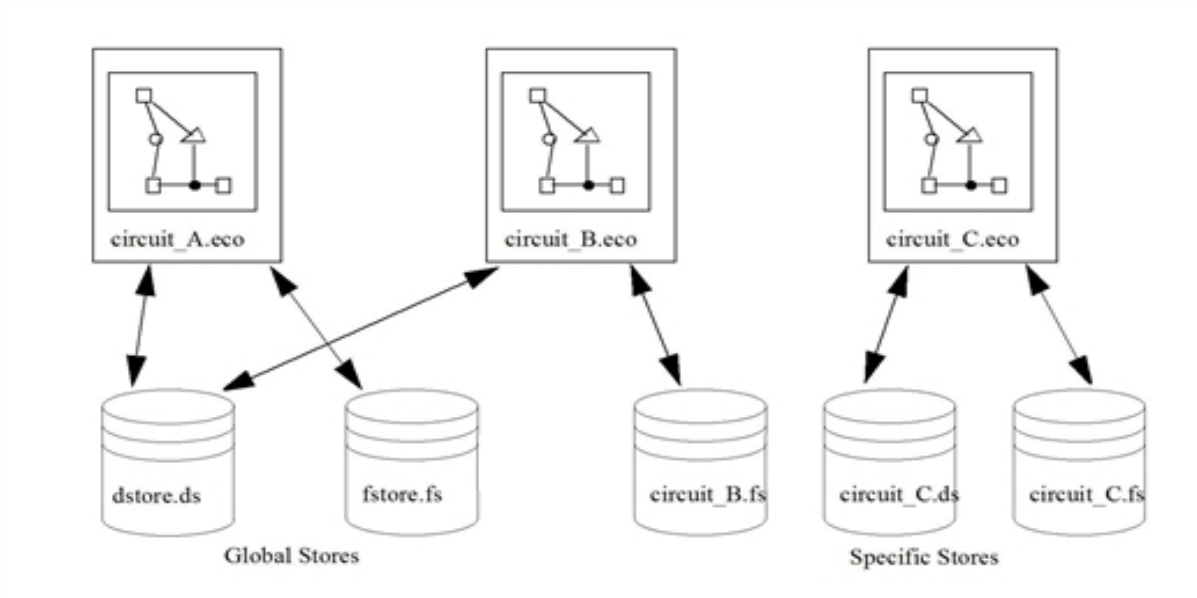
  When a correlation circuit is distributed to a managed node or to the management server, it checks whether a fact or data store with the same name as the compiled circuit file and with the appropriate extension (`fs` for the fact store and `ds` for the data store) exists in the specified directory. If none of them exists, the global fact or data store is used.

- *Global Data Store and Fact Store*

  The global data store file (`dstore.ds`) and the global fact store file (`fstore.fs`) can be shared across multiple correlation circuits. If a specific fact store does not exist for an event correlation policy, the global fact store is loaded. Similarly, if a specific data store does not exist for an event correlation policy, the global data store is loaded.

Figure 44 shows correlation circuits accessing global and specific fact and data stores.

Figure 44: Data Stores and Fact Stores in HPOM



Updating Fact Stores and Data Stores

When HPOM is restarted, or when a new or modified event correlation policy is distributed to the managed node or management server, the fact store and data store files are reloaded. It is also possible to manually force individual fact and data store files to be reloaded using the `ecsmgr` utility from the following location:

- On the management server

  `/opt/OV/bin/OpC/`
- On the managed node (UNIX)

  `/opt/OV/bin/OpC/utils/`
- On the managed node (Windows)

  `\usr\OV\bin\OpC\install\`

Note: You can use the `ecsmgr` utility only as a root user.

When using `ecsmgr`, specify whether you are communicating with the managed node ECS engine or the management server ECS engine. The management server instance is `11`, and the managed node instance is `12`.

You can use the following two commands to update an event correlation policy with a new fact store or data store:

- Data Store

  ```
  # ecsmgr -i <instance> -data_update <datastore_name> <filename>
  ```

For example:

```
# ecsmgr -i 12 -data_update DatabaseDown /var/opt/OV/conf/eeagt/DatabaseDown.ds
```

- Fact Store

  ```
  # ecsmgr -i <instance> -fact_update <factstore_name> <filename>
  ```

  For example:

  ```
  # ecsmgr -i 12 -fact_update fstore /var/opt/OV/conf/eeagt/fstore.fs
  ```

Note: The `ecsmgr` command updates only dynamic circuit parameters and statically evaluated parameters are not affected.

Tip: If the fact or data store files are exceptionally large, perform incremental updates by using smaller files to prevent the correlation process from blocking for a long period of time.

For more information on the `ecsmgr` utility, see the ecsmgr(1M) manual page.

Automating Distribution to the UNIX Nodes

Although fact stores and data stores are not distributed with the event correlation policy, it is possible to partially automate their distribution on the UNIX nodes.

To automate distribution of the fact stores and data stores to the UNIX nodes, perform the following procedure on the management server:

1. Create fact and data store files in the following directory:

   /var/opt/OV/share/databases/OpC/mgd_node/customer/*<arch>*/cmds

   Note: The HPOM commands directory includes *<arch>*, which stands for architecture of the commands (for example, `hp/alpha/tru64` or `sun/sparc/solaris10`).

   After you distribute the commands to the managed node, the fact and data store files are installed in the `/var/opt/OV/bin/OpC/cmds` directory.

2. Write a script that copies the fact and data store files from `/var/opt/OV/bin/OpC/cmds` to `/var/opt/OV/conf/OpC/` in the following directory:

   /var/opt/OV/share/databases/OpC/mgd_node/customer/*<arch>*/cmds

When you want to distribute the fact and data stores, you can distribute the commands to the managed node and run the script by using the HPOM broadcast command mechanism.

In some circumstances, the ECS annotate node feature is more suitable than fact stores or data stores for accessing data that changes regularly. For more information on the annotate mechanism, see "Annotate Mechanism " on the next page.

Annotate Mechanism

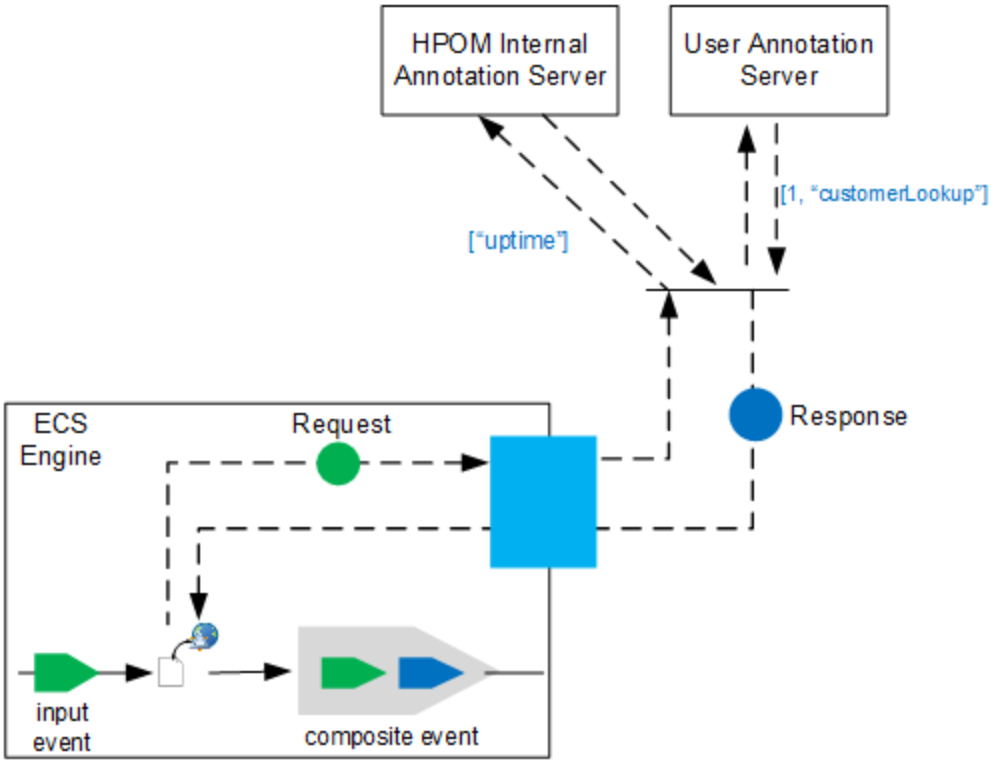
The ECS annotate node mechanism enables you to access external sources of information from a correlation circuit. The annotate node is generally used in preference to the fact store or data store when accessing information that can change on a regular basis.

The annotate node makes a call outside the ECS engine to an external process. The external process is known as an annotation server. The annotation server performs the appropriate processing and returns the information to the annotate node. This information can be used within the correlation circuit to assist in decision making or to enhance the information output from the circuit.

There are two types of annotation server available in HPOM:

- "Internal Annotation Server " on the next page
- "User-built Annotation Server " on page 219

Figure 45: Annotate mechanism in HPOM



The `Annotate_Spec` parameter of the annotate node is used to send requests to the annotation server. By default, if the first element of the list is a string, the HPOM internal annotation server intercepts the request and execute the string as a command. If the first element of the list is any data type other than a string (such as an integer), the HPOM internal annotation server does *not* intercept the request. Instead, the request is available for any user-built annotation servers.

You can override the default behavior and enable each annotation server to register for receiving requests only from the annotate nodes with specific names. To do so, set the following variables by using the `ovconfchg` command-line tool:

- on the management server: `ECM_ANNO_NODE`
- on the managed node: `ECA_ANNO_NODE`

If the `ECM_ANNO_NODE`, `ECA_ANNO_NODE`, or both variables are set, only annotation requests from nodes with the specified names are processed by the HPOM internal annotation server. The format of the variable is as follows:

```
ECM_ANNO_NODE <name1>[,<name2>...]
```

For example, to configure the Internal Annotation Server to process annotate requests from annotate node named `OVOEXE`, set the variable as follows:

```
ECM_ANNO_NODE OVOEXE
```

Note: To have multiple annotate nodes with the same name in your correlation circuit, place each of them in their own compound nodes to avoid naming conflicts.

Internal Annotation Server

The internal annotation server is used to run commands or scripts that return the results to the correlation circuit. The `Annotate_Spec` parameter of the annotate node must contain the full path to a command or script that is to be run. All parameters must also be included. The HPOM internal annotation server returns both the exit code and the standard output of the command or script.

Data returned from all annotation servers is placed in the second sub-event of the composite event that is an output from the annotate node.

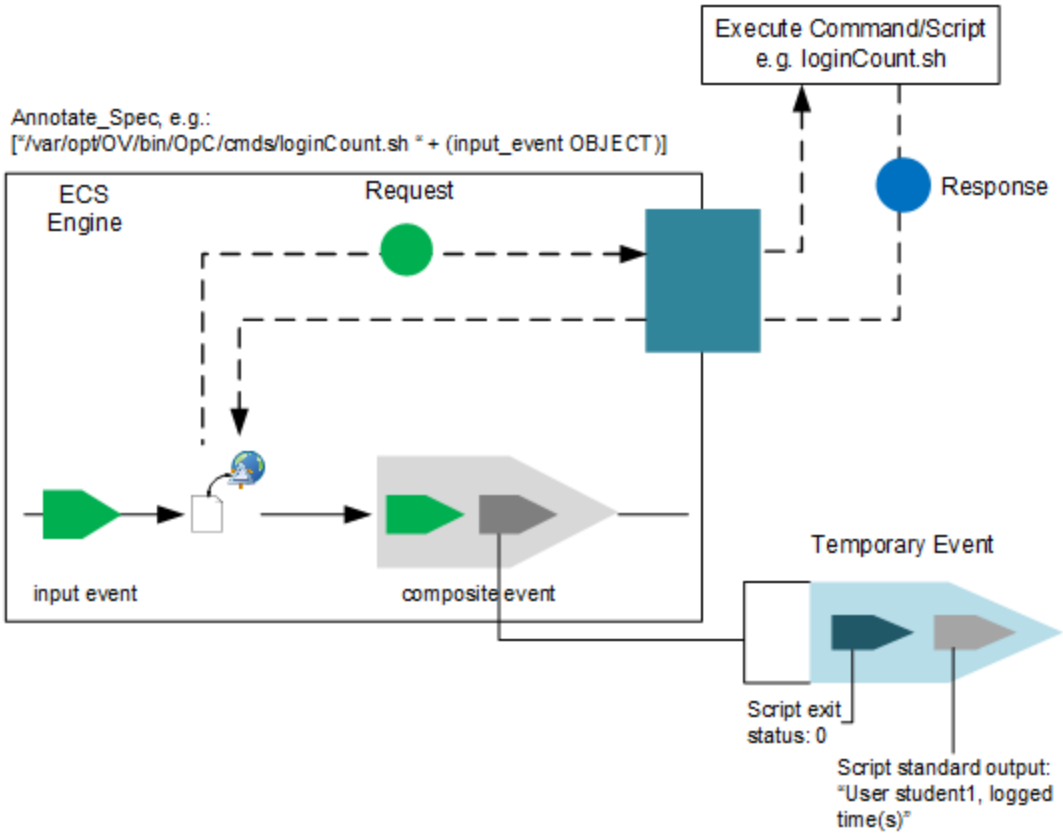
Figure 46 shows that the data returned from the HPOM internal annotation server contains two pieces of information: the exit code and the standard output of the command or script. To obtain the exit code in a filter node, use the following statement:

```
input_event 2 1
```

To obtain the standard output of the command or script, use the following statement:

```
input_event 2 2
```

Figure 46: HPOM Internal Annotation Server



Annotation Script

An annotation request sent to the HPOM internal annotation server must contain `Annotation_Spec` as a list with a single string. The string value is the fully qualified filename of the script or command that is to be run, followed by other parameters. For example:

```
["/var/opt/OV/bin/OpC/cmds/loginCount.sh student1"]
```

In this example, the `/var/opt/OV/bin/OpC/cmds/loginCount.sh` script is run with a single parameter `student1`. Additional parameters can be the subsequent positional parameters in the script (`$2`, `$3`, ...)

Usually the parameters passed to the script or command are obtained from the message attributes of the input message. For example:

```
["/var/opt/OV/bin/OpC/cmds/loginCount.sh " + (input_event OBJECT)]
```

The script or command is not distributed automatically as a part of the EC policy distribution. Because there is no predefined location for the script, you should provide the absolute path in `Annotation_Spec`.

Note: If you place the script files in the HPOM commands directory,

`/var/opt/OV/share/databases/OpC/mgd_node/customer/<arch>/cmds`, the script files are distributed to the `/var/opt/OV/bin/OpC/cmds` directory on the managed nodes.

This path is used in `Annotation_Spec` for the command that is to be run. For example:

`["/var/opt/OV/bin/OpC/cmds/loginCount.sh <other_cmd_parameters>"]`

The script uses the `exit` and `echo` commands (or equivalent) to produce the desired response.

The following script shows how to produce a text string that indicates a number of times a user (the username is the first parameter) logged on.

```
#!/bin/sh
# loginCount.sh script
COUNT=`who | grep $1 | wc -l`
echo "User $1, logged in $COUNT time(s)"
exit 0
```

User-built Annotation Server

For users who want to develop their own annotation server process, an annotation API is provided. User-built (custom) annotation servers are separate processes that register with the ECS engine and listen for annotation requests. Once a request is received, the annotation server processes the request, and then sends a response back to the correlation circuit by using the annotation API. User-built annotation servers can be used for many purposes, such as accessing an SLA database, querying MIBs on a network device, or querying topology information from another application.

Importing ECS Circuits

This section describes how to import an ECS circuit on an HPOM system. To create an ECS policy and upload the `.eco` (EC compiled circuit) and `.ecs` (EC source circuit)[a] files to it by using the command line tools[b], follow these steps:

1. On the HP Operations management server, list all ECS policies by running the following command:

   opcpolicy -list_pols pol_type=Event_Correlation
2. Choose an ECS policy and download it by running the following command:

   opcpolicy -download pol_name=*<policy_name>* version=*<version>* pol_type=Event_Correlation dir=*<download_dir>*

   For example:

[a]The .ecs file is optional.

[b]If you want to use the Administration UI, create a new policy of the Event Correlation policy type, and then, in the Source tab of the created ECS policy, upload the `.ecs` and `.eco` files to it.

opcpolicy -download pol_name=bad_su version=0009.0000 pol_type=Event_Correlation dir=/tmp/ecs_policy

The *<download_dir>* directory contains the *<id>*_circuit, *<id>*_source, *<id>*_data, and *<id>*_header.xml files that you should use to create a new ECS policy according to your needs.

3. List all the files in the download directory by running the following command:

   ls *<download_dir>*

   An output similar to the following one should appear:

   ```
   50287ad6-e455-11dc-94db-00306ef38b73_circuit
   50287ad6-e455-11dc-94db-00306ef38b73_data
   50287ad6-e455-11dc-94db-00306ef38b73_header.xml
   50287ad6-e455-11dc-94db-00306ef38b73_source
   ```

4. Edit the *<id>*_header.xml file by modifying the <name> and <description> lines as needed.

   For example (assuming that the name of the ECS policy you downloaded is bad_su):

   ```
   <name>bad_su</name>
   ```

   ```
   <description>suppress bad_su if followed by succeeded_su</description>
   ```

5. Edit the *<id>_data* file by modifying the ECS, DESCRIPTION, and CIRCUIT_FILE lines.

   For example (assuming that the name of the ECS policy you downloaded is bad_su):

   ```
   ECS "bad_su"
   ```

   ```
   DESCRIPTION "suppress bad_su if followed by succeeded_su"
   ```

   ```
   CIRCUIT_FILE "ECbad_su"
   ```

   Caution: Make sure that the values of ECS and DESCRIPTION correspond to the values of <name> and <description> as used in the *<id>*_header.xml file. For the circuit file name, you can use an optional name.

6. Transfer the .ecs and .eco files from the system with the ECS Designer to the HP Operations management server.
7. Copy the .ecs file to the *<id>*_source file and the .eco file to the *<id>*_circuit file.

   For example:

   cp newcircuit.ecs 50287ad6-e455-11dc-94db-00306ef38b73_source
   cp newcircuit.eco 50287ad6-e455-11dc-94db-00306ef38b73_circuit

8. Upload the policy by running the following command:

   opcpolicy -upload file=*<id>*_header.xml mode=add

   For example:

```
opcpolicy -upload file=/tmp/ecs_policy/50287ad6-e455-11dc-94db-00306ef38b73_
header.xml mode=add
```

9. Verify that the new ECS policy was created by running the following command:

   ```
   opcpolicy -list_pols pol_type=Event_Correlation
   ```

Caution: *Only if you use the Administration UI:* Make sure that you access the Administration UI from a browser running on the same system as the ECS Designer so that the local `.ecs` and `.eco` files can be referenced in the Administration UI.

When the `.ecs` and `.eco` files are referenced in the Administration UI, the policy is saved into the database together with the rest of the policy data.

Modifying the ECS Circuit Names

To modify the ECS circuit name, follow these steps:

1. Obtain a list of EC policy names by running the following command:

   ```
   # opcpolicy -list_pols pol_type=Event_Correlation
   ```

2. Obtain the corresponding compiled circuit name for a particular EC policy by downloading this policy. To do this, run the following command:

   ```
   # opcpolicy -download pol_name=<policy_name> version=<version> pol_type=Event_
   Correlation dir=<download_dir>
   ```

3. In the *<download dir>* directory where you downloaded the policy, open the *<id>*`_data` file and check the `CIRCUIT_FILE` attribute. The value of this attribute corresponds to the ECS circuit name.

   For example:

   ```
   CIRCUIT_FILE "85mtaliep0"
   ```

   To change the ECS circuit name, modify the value of the `CIRCUIT_FILE` attribute.

4. To use the ECS circuit with the new name, run the following command:

   ```
   # opcpolicy -upload dir=<download_dir>
   ```

   For more information about the `opcpolicy` command, see the *opcpolicy(1M)* manual page.

Handling Event Correlation Policy

Assume that you have an event correlation policy named `DatabaseDown` that will be distributed to a UNIX managed node and the circuit name is `EAAAa03016`. The event correlation policy accesses the configuration data from a specific data store and database relationships from the global fact store. Follow these steps:

1. Obtain the name and the versions of the EC policies by running the following command:

   ```
   # opcpolicy -list_pols pol_type=Event_Correlation
   ```

2. Obtain the compiled circuit name for the `DatabaseDown` policy by downloading this policy. To do this, run the following command:

   ```
   # opcpolicy -download pol_name=DatabaseDown version=1.0 pol_type=Event_
   Correlation dir=/tmp/circuit
   ```

3. In the `/tmp/circuit` directory, open the `*_data` file, and then check the `CIRCUIT_FILE` attribute:

   ```
   CIRCUIT_FILE "EAAAa03016"
   ```

   In this instance, `EAAAa03016` is the ECS circuit name.

4. Change the ECS circuit name to `DatabaseDown`, as follows:

   ```
   CIRCUIT_FILE "DatabaseDown"
   ```

5. To use the ECS circuit with the new name, run the following command:

   ```
   # opcpolicy -upload dir=/tmp/circuit
   ```

   For more information about the `opcpolicy` command, see the *opcpolicy(1M)* manual page.

6. Copy the specific data store file (`DatabaseDown.ds`) to the following location on the managed node:

   ```
   /var/opt/OV/conf/eaagt/
   ```

7. Copy the global fact store file (`fstore.fs`) to the following location on the managed node:

   ```
   /var/opt/OV/conf/eaagt/
   ```

When the `DatabaseDown` policy is distributed, it loads the `DatabaseDown.ds` data store file, as it has the same name as the compiled circuit file. Because there is no `DatabaseDown.fs` file, it loads the global `fstore.fs` fact store file.

Service Hours

Service hours are defined and agreed on time slots in which a service provider works on problems that are reported by HPOM and that are related to a given service. Each service may have its own period of service hours.

Buffering Messages

Messages received during the defined service hours are passed to the Java GUI message browser in the normal way. All messages that are received outside of the defined service hours are buffered. The

buffered messages can be viewed in the Java GUI pending messages browser.

Unbuffering Messages Automatically

Buffered messages are automatically unbuffered (that is, moved to the Java GUI message browser) At the start of the next period of defined service hours. After the buffered messages are moved to the Java GUI message browser, they may be worked on in the usual way.

Unbuffering Messages Manually

Buffered messages can be unbuffered manually, either from the Message Properties window or from the Java GUI Pending Messages Browser itself. Messages that are unbuffered manually are owned or marked by the user who carried out the unbuffer operation.

The following restrictions apply to the manual unbuffer operation:

- **Messages Sent Only When Buffer Time Has Expired**

  Buffered messages are sent to the trouble ticket and notification interface *either* when the buffer time of the message has expired (that is, at the start of the next service hours of the service that generated the message) *or* when an operator opens the message.
- **Messages Forwarded On Arrival on Management Server**

  If manager-to-manager forwarding is configured, buffered messages are forwarded to any other defined management servers as soon as the message arrives on the HP Operations management server.

Note: Other managers may have their own configuration for outages and service hours.

Defining Service Hours

To find out how to set up service hours and for more information about the policies and syntax rules used to define service hours, see the *HPOM Administrator's Reference*.

Scheduled Outages

In the context of HPOM, a **scheduled outage** allows you to **log** or **suppress** (delete) incoming messages during a defined period of time if the messages relate to a service or system that is known or planned to be unavailable. For example, you can use a scheduled outage to suppress all messages from a component that is temporarily unavailable for maintenance reasons.

Scheduling an Outage

A scheduled outage defines a planned and potentially recurring period of time in which one or more components or services in a distributed working environment are not available, and in which messages relating to these components or services need to be logged or perhaps even deleted. An outage may be set dynamically if a major problem occurs, and if further messages from a particular component need to be suppressed. For example, if a database server goes down, you could suppress, for a defined period of time, all messages that are related to the database server. The status of an outage can also be set by an external application.

Defining a Scheduled Outage

To find out how to set up a scheduled outage and for more information about the policies and syntax rules used to define a scheduled outage, see the *HPOM Administrator's Reference*.

Configuring Service Hours and Scheduled Outages

As HPOM administrator, you can configure service hours and scheduled outages on the management server with a policy similar to the one used to configure flexible management. The same syntax is used and, as a consequence, may be checked with the `opcmomchk` tool. This policy is located in `/etc/opt/OV/share/conf/OpC/mgmt_sv/respmgrs`. For more information on the policy and the syntax used, see the *HPOM Administrator's Reference*.

Note: Scheduled outages and service hours may be configured by an external application. However, the designated external application must create the policy for outages and service hours and use the `opccfgout(1M)` command to control outages.

Setting Custom Message Attributes Based on Message Selection Criteria

You can set custom message attributes based on different message conditions (for example, a node name, an IP address, a message text, severity, and so on). To do this, use the `msgmodify` configuration file that allows message matching based on different message conditions.

Setting a custom message attribute based on a message condition is an example of message modification. To enable message modification, follow these steps:

1. Copy the `msgmodify` template from the `tmpl_respmgrs` directory to the `work_respmgrs` directory.

   Both directories can be found at the following location:

   `/etc/opt/OV/share/conf/OpC/mgmt_sv`
2. Modify the template to suit your environment.
3. Check the syntax by using the `opcmomchk(1)` tool.
4. Copy the modified file to the following directory:

   `/etc/opt/OV/share/conf/OpC/mgmt_sv/respmgrs`
5. Run the `opccfgout -update` command.

For detailed information about the syntax of the `msgmodify` configuration file, see the *msgmodify(4)* manual page.

Chapter 5: Scalable Architecture for Multiple Management Servers

In this Chapter

This chapter describes how to configure and manage HPOM in widely distributed environments. It also discusses the underlying concepts of flexible management and manager-of-manager (MoM) communications.

Note: In this chapter, the term “manager” refers to the **management server**, and the term “agents” refers to **managed nodes**.

Who Should Read this Chapter

This chapter is designed for HPOM administrators.

What this Chapter Does

This chapter explains the following:

- **Communication Between Servers**

  Basic concepts and sample applications behind server-to-server communication.
- **Transferring Messages to Servers**

  Configurable message transferring from managed nodes to different servers, based on time or message attributes.

  Changes of HPOM message attributes, for example message severity, message text, and custom message attributes can be synchronized with other HPOM management servers.
- **Configuring Management Server Responsibilities**

  Setting up management server responsibilities for the HPOM managed nodes. These responsibilities permit the full use of knowledge centers, and eliminates single-point-of-failure bottlenecks.
- **Distributing Server Configurations**

  Distributing server configurations to other management servers (for example, moving configurations from a test environment to a production environment).

- **Forwarding Messages Between Servers**

  Forwarding messages between management servers.

Flexible Management

HPOM enables you to configure your environment hierarchically. You can then spread management responsibility across multiple management levels, according to criteria as diverse as operator expertise, geographical location, and the time of day. This flexible management enables operators to focus on doing what they do well, safe in the knowledge that they have round-the-clock technical support available automatically and on demand.

Note: For information about flexible management in a Japanese environment, see the *HPOM Administrator's Reference*.

Default Setup

The default setup for HPOM consists of one management server that interacts with agents installed on the managed nodes. In this default setup, agents can send messages only to that management server. You can, however, easily reconfigure HPOM so the agents on the various managed nodes can send messages to other management servers as well.

Primary Manager

The initial management server is called the **primary manager** because it is the HP Operations server that is primarily responsible for the HPOM managed nodes. However, you can switch the primary manager function to another server. If you do so, messages from the managed nodes are redirected to the new (usually temporary) primary manager, which can also perform automatic actions on those managed nodes.

Advantages of Flexible Management

The flexible management architecture of HPOM enables you to do the following:

- **Manage Worldwide Network**

  Manage your worldwide network more effectively (for example, by using the **follow-the-sun** functionality).
- **Increase Efficiency**

  Increase efficiency by implementing competence center policies.

- **Forward Messages**

  Forward messages between management servers.

- **Manage Expansion**

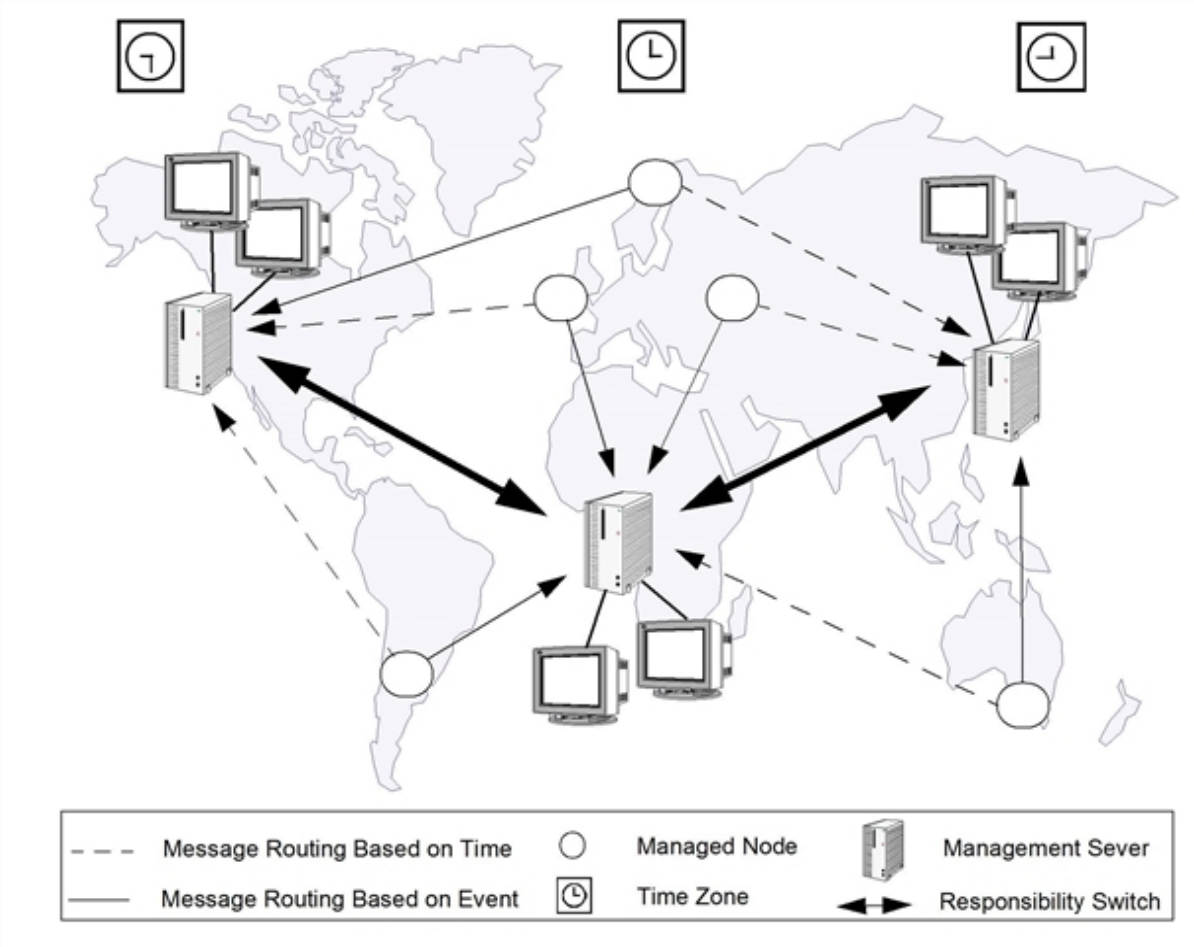
  Manage an expanding network environment, and reduce primary server overload.

Having all managed nodes report to a single management server can cause a bottleneck. This bottleneck can adversely affect database performance. This problem is eliminated if managed nodes report to multiple management servers. For more information about distributed management responsibility, see "Management Responsibilities in Domain Hierarchies" on page 233.

Follow-the-Sun Control

If your distributed operations take place over several time zones (see Figure 47), you can use HPOM to rotate management responsibilities by implementing **follow-the-sun** control. Depending on the time of day, managed nodes report to different management servers. The same capability enables you to set up specific management servers for weekend or holiday operations.

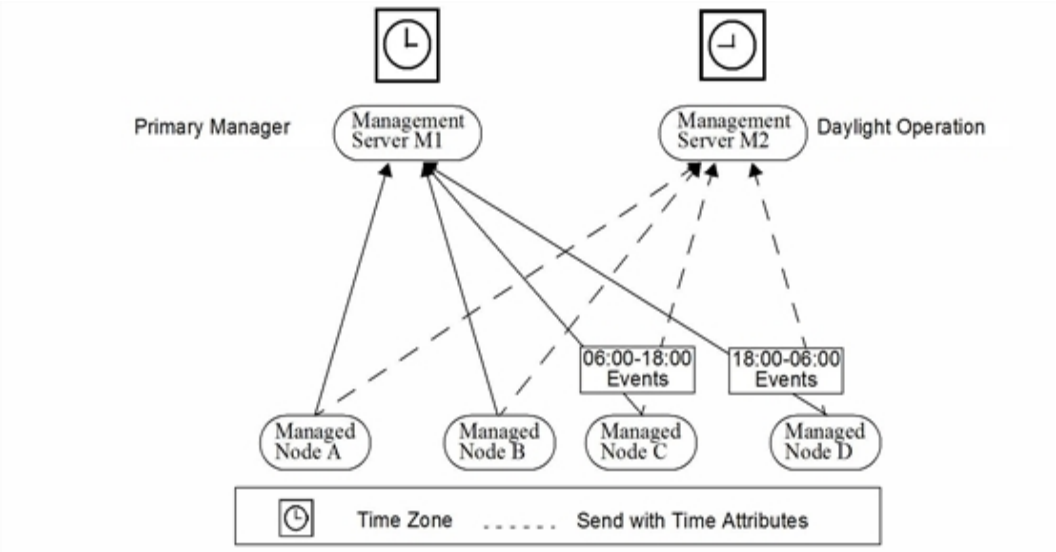
Figure 47: Worldwide Management Domain



The follow-the-sun concept is based on the idea of sending messages to different management servers, according to predefined time attributes. HPOM enables you to configure managed nodes to send messages to different management servers according to rules defined in a **time policy**.

For example, Figure 48 shows how you can configure HPOM so that all messages generated between 06:00 and 18:00 are sent to HP Operations management server M1 from managed nodes C and D. Messages generated between 18:00 and 06:00 are sent to HP Operations management server M2. With follow-the-sun functionality, you can control your entire environment throughout the day by assigning daylight operating shifts to the corresponding regional areas.

Figure 48: Using Time or Message Attributes to Forward Messages



For example, if your enterprise has implemented a 24-hour support desk at a central location, you can use HPOM to send messages from the regional nodes directly to the central management server during regional department off-hours. Implementing follow-the-sun policies requires the addition of two entries in the `allnodes` configuration file.

These two entries might take the following form:

```
CONDITION TIME 6am-6pm SEND TO $OPC_PRIMARY_MGR
CONDITION TIME 6pm-6am SEND TO MC
```

Note: Use the variable `$OPC_PRIMARY_MGR` to send messages to different management servers. This attribute can denote different systems at different times.

The follow-the-sun concept is not restricted to rules based on the time of day. You can also configure the sending of messages to different management servers based on the day of the week, a specific date or dates, or frequency. For details, see "Time Policies" on page 238 and the *opcmom(4)* manual page.

Competence Centers

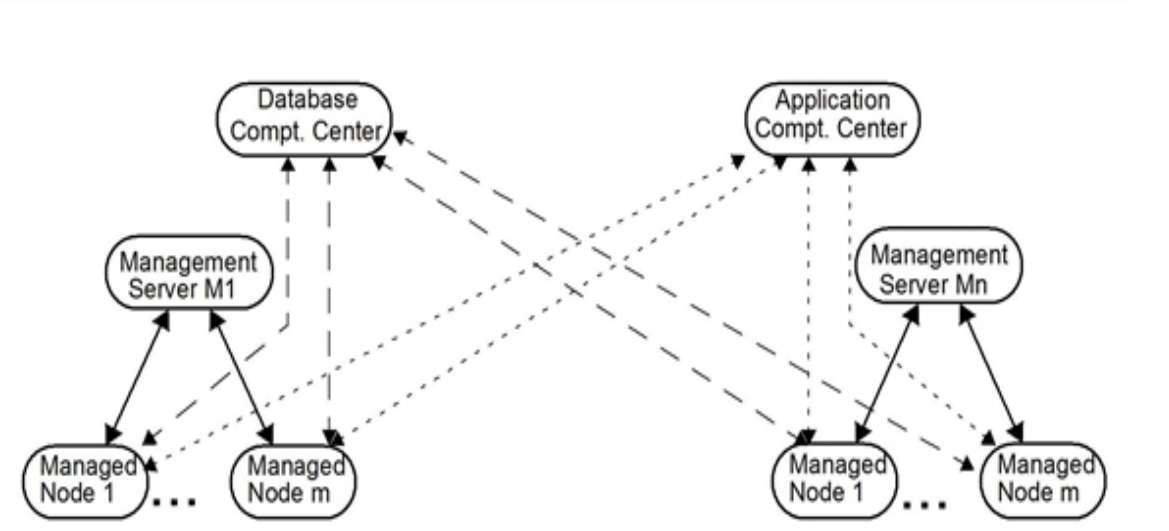
If you operate in a large enterprise with multiple management servers distributed over a wide area, specialist knowledge relating to a specific subject is not always available locally. For this reason, it is helpful to configure managed nodes that communicate with a server other than the primary manager. Other servers in your network may have specialized computing expertise available (for example, expertise about database or spool management). HPOM lets you set up your managed nodes to communicate with the management servers of your choice anywhere in your network.

For example, your enterprise might have a center responsible for all operating system-related problems. In addition, another center of expertise may be responsible for the database, which is used company-wide. Managed nodes can be configured to send messages about operating systems to one center of expertise, and all messages relating to database problems to another. In Figure 49, all managed nodes send all database events to management server M1.

Distributing Responsibility in Competence Centers

Depending on the type of message, a simple competence-center-oriented hierarchy, such as that shown in Figure 49, presents a more flexible environment than a hierarchy based on a central management server.

Figure 49: Communication in a Competence Center-oriented Environment



Unlike a central server hierarchy, a competence center hierarchy distributes responsibility for managed nodes. In a competence center hierarchy, regional managers are not solely responsible for managed nodes. Messages about specific subjects (for example, databases) go directly to a defined management server, where expertise exists to solve problems related to the subject for all managed nodes.

Configuring Competence Centers

The following conceptual syntax describes a sample competence center implementation:

```
IF MSGGRP=databases SEND TO Database Competence Center
IF MSGGRP=finance SEND TO Application Competence Center
IF MSGGRP=cad SEND TO Application Competence Center
```

Note: You can include follow-the-sun capabilities in the configuration by adding time conditions to the relevant competence center conditions.

Backup Servers

Typically in a backup scenario, two HP Operations management servers are configured identically. The main installation is referred to as the primary management server and the other as a backup server.

If the primary manager is temporarily inaccessible for any reason, you can configure HPOM to transfer messages from the managed nodes to one or more designated backup management servers.

Configuring Backup Servers

To respond to these messages, the backup server needs the relevant configuration and policies from its primary manager. That is, before problems arise on the primary manager, you must distribute relevant configurations and policies to designated backup servers and nodes. In addition, you must configure each managed node with specific timeframe and conditions for sending particular messages to particular servers.

Distributing Configurations and Policies

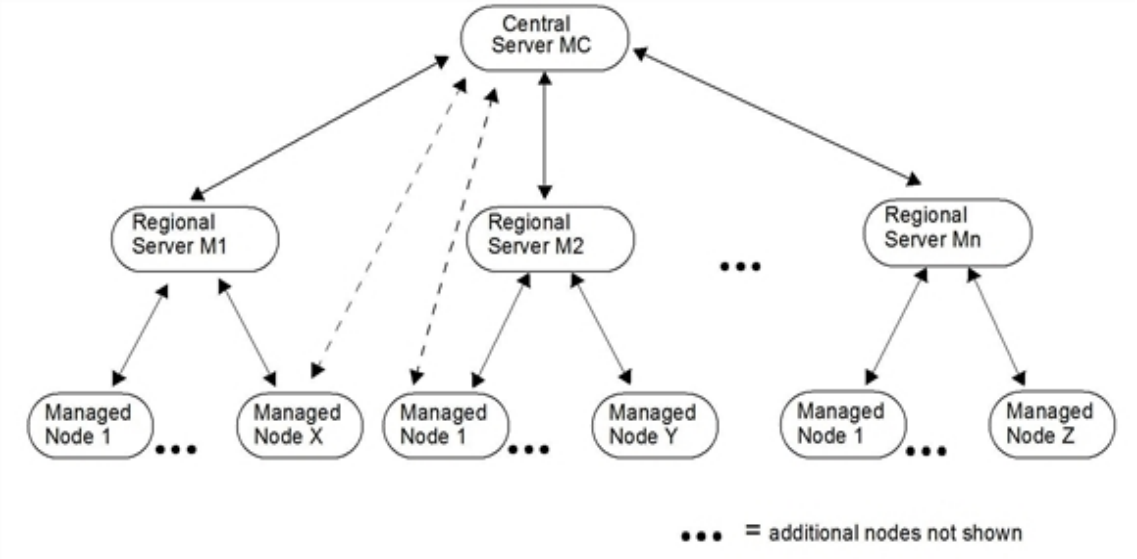
Distributing relevant configurations and policies to all relevant management servers and nodes simplifies centralized product development. You can develop configurations and policies on the central server, and then distribute them to designated servers and managed nodes.

For example, you might want to develop policies at your headquarters, then install or update the policies to several replicated branches. HPOM enables you to distribute configurations, policies, and software by downloading this data to a file and then uploading it to any number of servers.

Management Hierarchies

Typically, manufacturing environments, such as that shown in Figure 50, present a clear hierarchy of management. You can use the responsibilities that are in place with manager-to-manager communication without re-organizing your environment.

Figure 50: Typical Manufacturing Environment and Communication Links



Management Profiles in the Management Hierarchies

The environment illustrated in Figure 50 is very prevalent in the manufacturing industry, where companies have multiple manufacturing sites, which are distributed geographically. This kind of geographical distribution can be compared to replicated site management. All sites usually have a similar management profile. That is, all sites have similar managed objects, policies, and end-user responsibilities.

Setup Ratio in Management Hierarchies

Although the size of such an environment varies significantly, according to the type of industry, a typical setup ratio might look as follows:

- One central management server
- 10 to 20 regional management servers
- 100 to 200 managed nodes (that is, server and multi-user systems) running an HP Operations agent
- Up to 5000 additional managed elements submitting SNMP events

Management Responsibilities in Domain Hierarchies

In hierarchical HPOM domains, each managed node runs an HP Operations agent that manages the system as well as the applications (for example, CAD applications, financial applications, and

databases) running on that system. The managed node reports messages to the regional management server. All managed nodes in a given region have the identical configuration.

Regional Management Servers

Regional management servers (M1-Mn in Figure 50) run the HP Operations management server software as well as a local HP Operations agent. These servers control the systems in that region. Typically, a regional site manages a local area network (LAN) environment to avoid costly wide area network (WAN) traffic. Operators managing this regional site are responsible for keeping the managed nodes and the applications up and running.

Central Server

The central server runs the HP Operations management server software, a local HP Operations agent, the regional management server systems, and the management applications running on those systems (for example, Data Protector, Performance).

From an operational perspective, the HPOM operators at the central site can be compared to specialists in a help desk–type organization. They handle problems that cannot be solved at a regional level. From the administrative perspective, the central site is in charge of developing, distributing, and maintaining the configuration of the regional servers. This is the most sensible configuration because the application know-how is centralized, and the regional configuration is nearly identical.

Configuring the Management Server

In a centralized server environment, the managed nodes report all problems to their regional management server. The regional management server acts as the primary manager for all agents.

Configuring the Central Server as Action-allowed Manager

In a centralized server environment, the central server is configured as **action-allowed manager** for all agents. Configuring the central server as action-allowed manager for all agents allows the central server to perform actions on the distributed managed node. This centralized control of distributed managed nodes enables the central server to manage responsibility switches.

Because the configuration for all managed nodes of a region should be identical, configuring the central server as action-allowed manager requires minimal effort. The HPOM administrator needs to configure just one file on the regional management server. This file holds entries specifying the secondary managers and the action-allowed manager.

Specifying the Central Server as Secondary Manager

When you specify secondary managers, it is good practice to define the central server as a secondary manager, too. In this way, if the primary manager fails, management responsibility can be switched to the secondary manager as a backup.

A sample setup file is located in the following directory:

`/etc/opt/OV/share/conf/OpC/mgmt_sv/tmpl_respmgrs/hierarchy.agt`

If this file is required for use, you must copy it to the following directory:

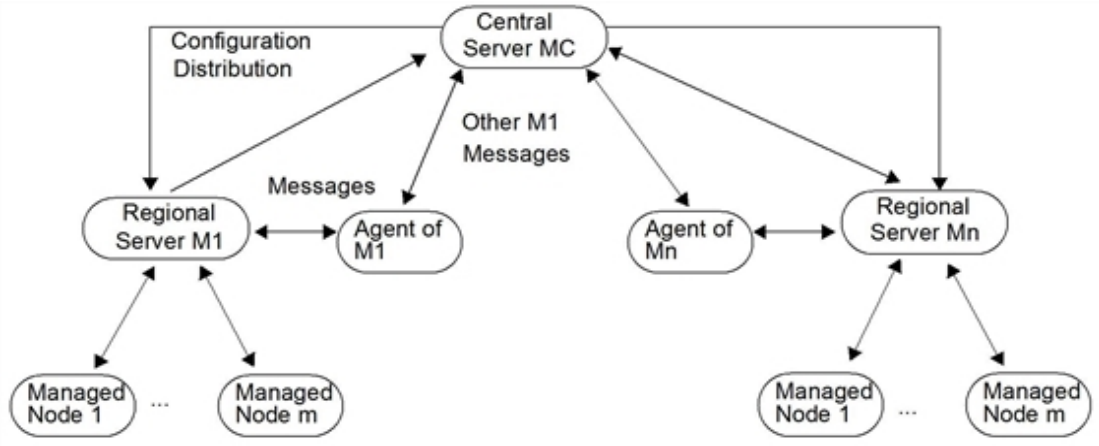
`/etc/opt/OV/share/conf/OpC/mgmt_sv/respmgrs`

From this directory, the file is read and its contents implemented at HPOM startup.

Configuring the Central Management Server

The central management server (MC) in Figure 51 controls regional management server systems (M1 to Mn) as managed nodes. To receive messages, you must configure all the corresponding nodes and assign them to operators.

Figure 51: Central Server Configuration and Communication Links



Storing a complete master configuration (that is, nodes, message groups, operators, and policies) on the central server has some advantages. Since your HPOM experts may be concentrated at one location, it would make sense for them to develop the configuration for each regional site centrally, and then distribute the configuration to those sites.

Configuring Responsible Managers

HPOM enables you to develop a single configuration for responsible managers. You create a file describing the configuration, then storing the file in the `respmgrs` directory on the primary management server. The name of the file is determined by the managed nodes you have in your environment. The text defines when, where, and how the messages are conveyed.

Creating a Configuration File

In the configuration file for responsible managers, you need to configure the following:

- Primary and secondary management servers
- Action-allowed management server
- Date-and-time policies for sending messages to various management servers
- Message-attribute rules for sending messages to various managers

For example, if you want the configuration to apply to all nodes in a given environment, you would develop one configuration file for all nodes. You would name this configuration file `allnodes`. If the configuration applies only to a specific node, the file name is the hex form of the IP address of that node and is generated with the command `opc_ip_addr`. If some nodes have the same configuration, you can create links to the node-specific file with the same configurations. If no specific file exists for a node, HPOM uses the configuration it finds in the `allnodes` file.

For more information on file locations and the steps required to set up a configuration for responsible managers, see the online help for HPOM or the *opcmom(4)* manual page.

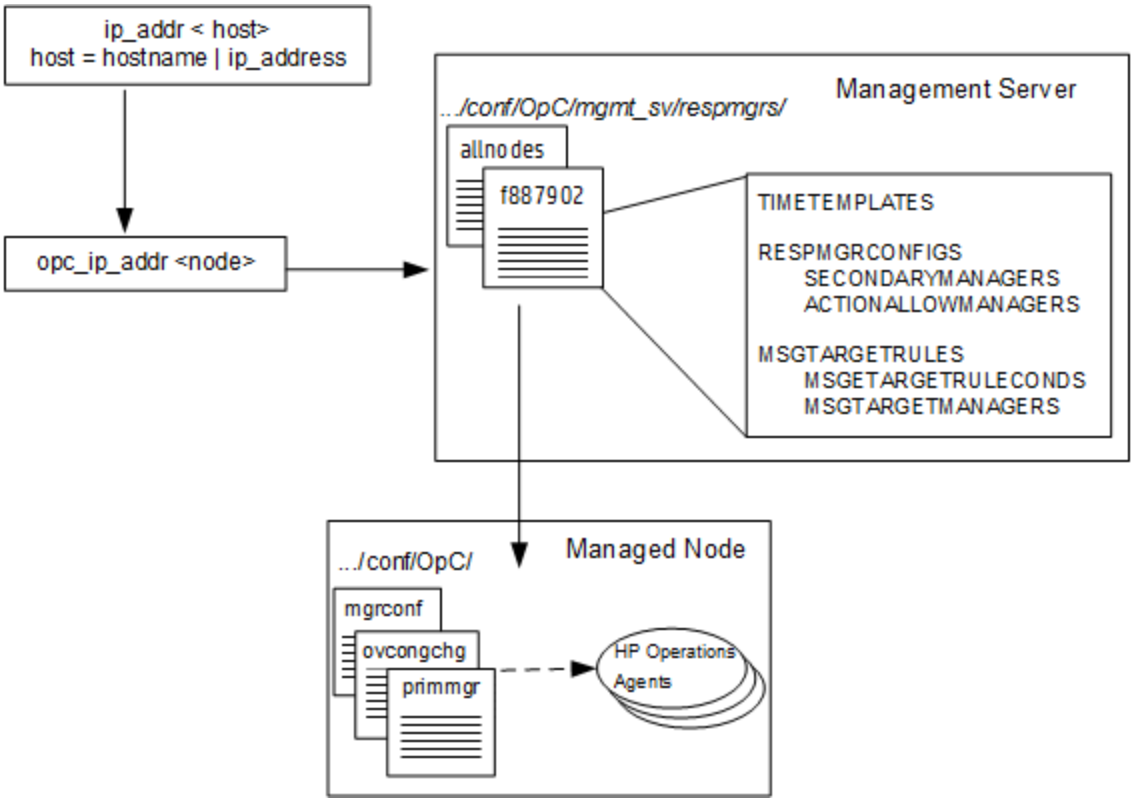
Note: For more information about flexible management in a Japanese environment, see the *HPOM Administrator's Reference*.

Distributing the Configuration File

HPOM automatically distributes the configuration to the managed nodes when it distributes policies. At the same time, HPOM re-initializes the intelligent agents with the new or updated configuration. The responsible manager configuration file `mgrconf` is located on the managed node. The current primary manager hostname is stored in the file, `primmgr`. If `primmgr` does not exist, HPOM uses the `ovconfchg` configuration tool for HTTPS-based managed nodes.

Figure 52 shows responsible manager policies for managed nodes.

Figure 52: Responsible Manager Policies for Managed Nodes



Message Target Rules

HPOM uses a list of **message target rules** to determine whether to send a message to a defined management server and, if so, to which one.

Parts of a Message Target Rule

A message target rule consists of three parts:

- Message attribute rule
- Time policy
- Defined management server

Example of a Message Target Rule for Printing Group

A message target rule for a printing group would have the following conceptual structure:

message group = "printing"

current time fits *time template 2 .....(message) --> mgr 2*

current time fits *time template 1 .....(message) --> mgr 1*

current time fits *time template 3 .....(message) --> mgr 3*

In this example, all messages with the message group “printing” that meet the time conditions in policy 1 are forwarded to HP Operations management server 1. All messages that meet the time conditions in policy 2 will be forwarded to HP Operations management server 2. Time policy 3 functions the same.

Example of a Message Target Rule for a Database Group

A message target rule for a database group would have the following conceptual structure:

message group = “database”

current time fits *time template 1 .....(message) --> mgr 2*

current time fits *time template 2 .....(message)--> mgr 3*

current time fits *time template 3 .....(message)--> mgr 1*

In this example, all messages with the message group “database” that meet the time conditions in policy 1 are forwarded to HP Operations management server 2. All messages that meet the time conditions in policy 2 are forwarded to HP Operations management server 3. And so on.

Time Policies

A **time policy** is a set of conditions (or rules) that tells the agent to which management server and at what time a given managed node should send specific messages. You create time conditions and save them in time policies. You can combine simple rules to set up more complex constructions (for example, “on Monday, Wednesday and Thursday from 10 am to 11:35 am from January to March”). Time conditions are defined by using the 24-hour clock notation (for example, for 1:00 p.m., you would enter “13:00”).

Setting Time Intervals

HPOM enables you to set several different time intervals as follows:

- **No Time**

  If you specify no particular time, day of the week, or year, HPOM assumes you want the condition to be true from 00:00 to 24:00 every day of the year, every year. If you specify a condition, HPOM assumes the condition should apply continually for the time and day specified.

  For example, specifying “Tuesdays” starts a condition every Tuesday from 00:00 to 24:00 throughout the year, every year.
- **Span of Time**

Specify a time range (for example, “from 7:00 to 17:00”).

- **Wildcard (*) Date or Period**

  Use wildcards (*) in dates or periods of time (for example, to set a condition for January 31 every year, you would enter “1/31/*”).

Configuring Time-Indifferent Policies

HPOM requires that you set up a time policy for the message target rules, even if your scheduled action is time-indifferent. HPOM provides the variable `OPC_ALWAYS` to configure time-indifferent policies.

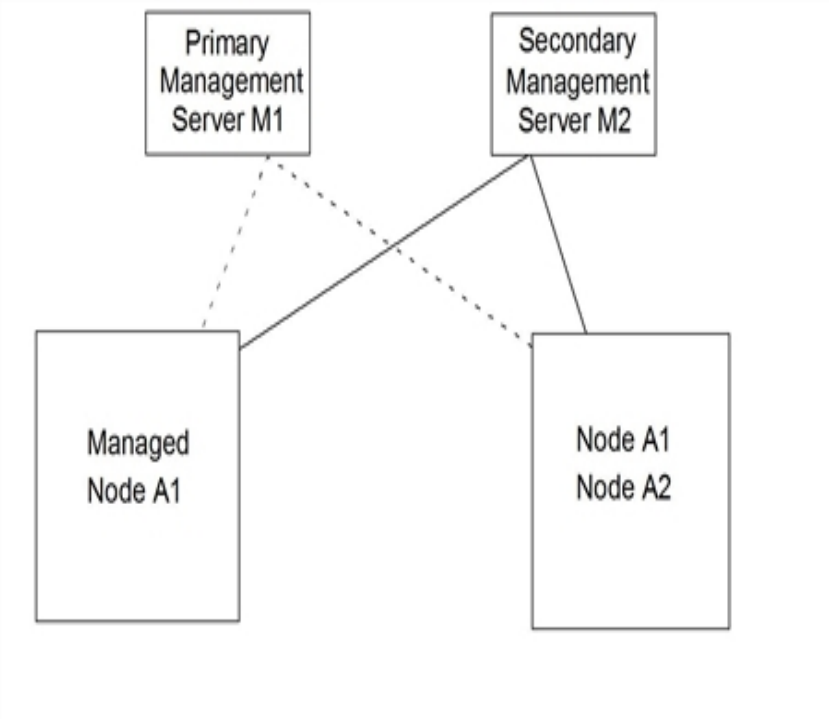
Specifying the Primary Manager

By default, all messages are consolidated on the HP Operations management server from which you originally installed the HP Operations agent software. In the default configuration, one HP Operations management server manages the HP Operations managed node. However, HPOM enables you to spread management responsibility for nodes over multiple HP Operations management servers.

Switching to a Secondary Manager

When you switch primary management responsibility to a secondary management server, you give the secondary management server responsibility for the HP Operations nodes formerly managed by the primary manager.

Figure 53: Switching Primary Management Responsibility



You initiate a management responsibility switch on the secondary management server by using the `opcragt -primmgr` command. The command accepts as parameters a list of managed nodes (hostnames or IP addresses) or node-group names. For details, see the *opcragt(1m)* manual page.

Note: If `OPC_PRIMARY_MGR` is not set or is invalid, the HP Operations management server is denoted by the `MANAGER` setting. Invalid means that the `OPC_PRIMARY_MGR` is not specified as a secondary manager nor as an action-allowed manager, nor as the initial manager.

The `OPC_PRIMARY_MGR` is a message related setting. It maps to the `$OPC_PRIMARY_MGR` variable, which may be used in message target rules of the responsible manager policy to enable sending messages to the specified HP Operations management server.

Enabling a Secondary Manager to Execute Actions

If you want to allow the secondary manager to execute actions on the nodes for which it is to assume responsibility, you must add a keyword to the configuration file.

Or add the following line to allow each current primary manager to become an action-allowed manager:

```
ACTIONALLOWMANAGER $OPC_PRIMARY_MGR
```

Reversing a Manager Switch

Primary management responsibility does not automatically revert to the original primary manager after the new primary management server gives up primary management responsibility. To reverse a primary management responsibility switch, you need to reconfigure the original primary management server as a secondary management server, as well as an action-allowed manager, and then switch the responsibility back manually by using `opcragt -primmgr`.

Delegating Manager Responsibilities

HPOM Installation Manager responsibilities, such as heartbeat polling and license counting, can be delegated to the primary server by executing the *opchbp(1m)* or *opcsw(1m)* command. For more information, see the corresponding manual pages.

Switching to a Backup Manager

Switching primary management responsibility can work as a fail-safe for system shut downs or factory-wide power failures. For example, you could set up a secondary management server to monitor the health of a primary management server by using interval polling. The secondary management server would then notify the HPOM administrator if a system failure occurred at the primary management server. The HPOM administrator would then switch primary management responsibility to the secondary HP Operations management server, which would assume control of all the managed nodes previously managed by the failed management server.

Specifying Action-Allowed Managers

A managed node allows only those management servers defined as **action-allowed managers** in its `mgrconf` file to carry out actions on that managed node. Consequently, a secondary management server that assumes primary management server responsibility for a given node (or nodes) must be configured as an action-allowed manager on the managed nodes for which it has assumed responsibility. Otherwise, it cannot carry out actions on those nodes.

By default, the only HP Operations management server that is allowed to execute actions on a managed node is the HPOM **Installation Manager**. To provide you greater operational flexibility, HPOM lets you configure several HP Operations management servers so that they can also execute actions on shared managed nodes.

Note: The primary manager should be configured as action-allowed manager. Add the following line to the responsible-manager configuration:

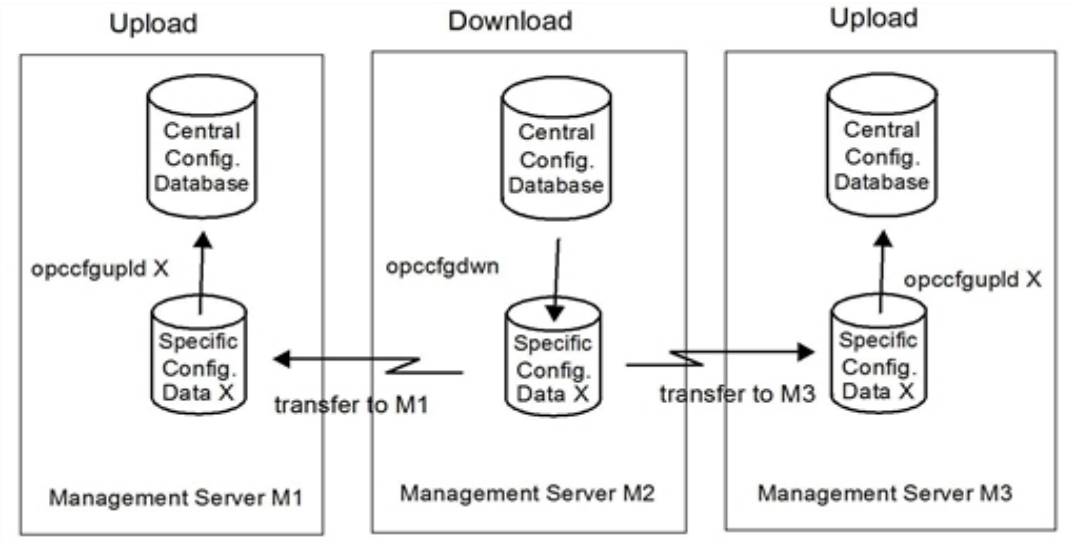
```
ACTIONALLOWMANAGER $OPC_PRIMARY_MGR
```

Distributing Configurations to Other Servers

If your environment consists of multiple HPOM domains acting as replicated sites, it is useful to develop configurations centrally, then distribute these configurations to the various management servers. For example, an HPOM administrator in a head office might want to define the configuration or application at your site, then download this data to a set of files to be picked up at another site.

After HPOM has distributed the configuration files to other servers, administrators at the remote locations can then upload the files to their databases for use, as shown in Figure 54.

Figure 54: Downloading and Uploading Configuration Files



Distributing configurations to different management servers involves three major steps:

1. **Configuration Parts Download**

   This information is stored in an index file used by HPOM when you run the download the `opccfgdwn` command. You can re-use an old specification to download the same configuration type.

2. **Distribute Downloaded Files**

   Distribute downloaded files to another HP Operations management server. If a trust relationship is established between the local and the remote management server, use the following procedure:

   a. Create a tar file containing the downloaded configuration.

   b. Use the `ovdeploy` command-line tool to securely copy the tar package to the remote management server, for example:

   ```
   ovdeploy -upload -file config.tar -targetdir /tmp -host remote.server.com
   ```

Note: If a trust relationship between management servers is not established, use some other method, for example, `FTP`, `rcp`, `scp`, and so on.

3. **Upload Files into Database**

   The administrator at the receiving HP Operations management server uploads the files into the local database by using the `opccfgupld` command. The administrator who wants automatic uploads schedules them by using the `Scheduled Actions`.

Caution: Do not change configuration data (for example, policies, operators, and so on) while uploading. Otherwise, the uploaded data may be corrupted.

The administrator can use the `contents` option with the upload command `opccfgupld` to check the data contained in a configuration. For more information about other upload options, see the *opccfgupld (1m)* manual page.

Uploading Configuration Data on the Local Management Server

After configuration data have been downloaded and distributed to another management server, you need to upload that file to the database on the local management server. Upload configuration data by using the following command:

```
/opt/OV/bin/OpC/opccfgupld <upload_directory>
```

For more information about configuration upload, see "Synchronization of the Configuration Changes " on page 80. For information about the parameters of this command, see the *opccfgupld(1m)* manual page.

Note: If you want HPOM to upload the configuration or application automatically, schedule the `opccfgupld` command by using `at` or `cron`.

Forwarding Messages Between Management Servers

Message forwarding is a feature that allows you to forward messages from one HP Operations management server to another, inform other management servers of problems, and allow another management server to carry out actions associated with the forwarded message. The idea is to increase flexibility by notifying other management servers of messages that may be relevant to them. In addition, HPOM enables you to pass control of messages from one management server to another management server, or even to several management servers. Forwarded messages are processed as if they were normal HPOM messages on the target management server. For example, forwarded

messages are processed by the MSI if it is configured, or the messages are forwarded to trouble ticket systems or notification services.

In HPOM, message forwarding is automatic. That is, you can use a policy to configure the source management server to forward messages. The forwarded message that arrives on the target management server is a copy of the reference message on the source management server.

Message forwarding always happens when a message arrives on the management server. The message cannot be forwarded afterward (that is, forwarding on demand does not exist). You cannot transform notification messages (marked with `n` in the message browser status column) into normal messages and vice versa using the GUI or the command line.

Normal and Notification Messages

All messages are sent as normal messages unless you explicitly define that they should be sent as notification messages. In addition, when in the message forwarding configuration file (`/etc/opt/OV/share/conf/OpC/mgmt_sv/respmgrs/msgforw`), the `MSGCONTROLLINGMGR` or `NOTIFYMGR` keyword is defined in a forwarding condition, you do not have to change anything.

Notification messages are for informational purposes only and allow a limited set of operations. You can define any number of target management servers to receive notification messages. Keep in mind, however, that forwarding notification messages to other management servers is disabled by default (that is, the `OPC_FORWARD_READONLY_MSGS` server configuration variable is set to `FALSE`). If you set it to `TRUE`, notification messages can be forwarded to other management servers. For detailed information about server configuration variables, see the *HPOM Server Configuration Variables* document.

Note: HPOM for Windows does not support notification messages. Operations Manager i (OMi), however, does support them.

Operations on normal messages are always forwarded to all management servers holding these messages. On the other hand, some of the operations on notification messages have only a local impact and are not forwarded any further. Table 13 shows which operations you can or cannot perform on each type of messages.

Table 13: Operations on Normal and Notification Messages

Operation	Normal Messages	Notification Messages
Acknowledge/unacknowledge	√	√ [a]
Add/delete/modify annotation	√	√

[a] This operation has a local effect only (that is, the copies of the message on other management servers are not acknowledged/unacknowledged).

Operations on Normal and Notification Messages , continued

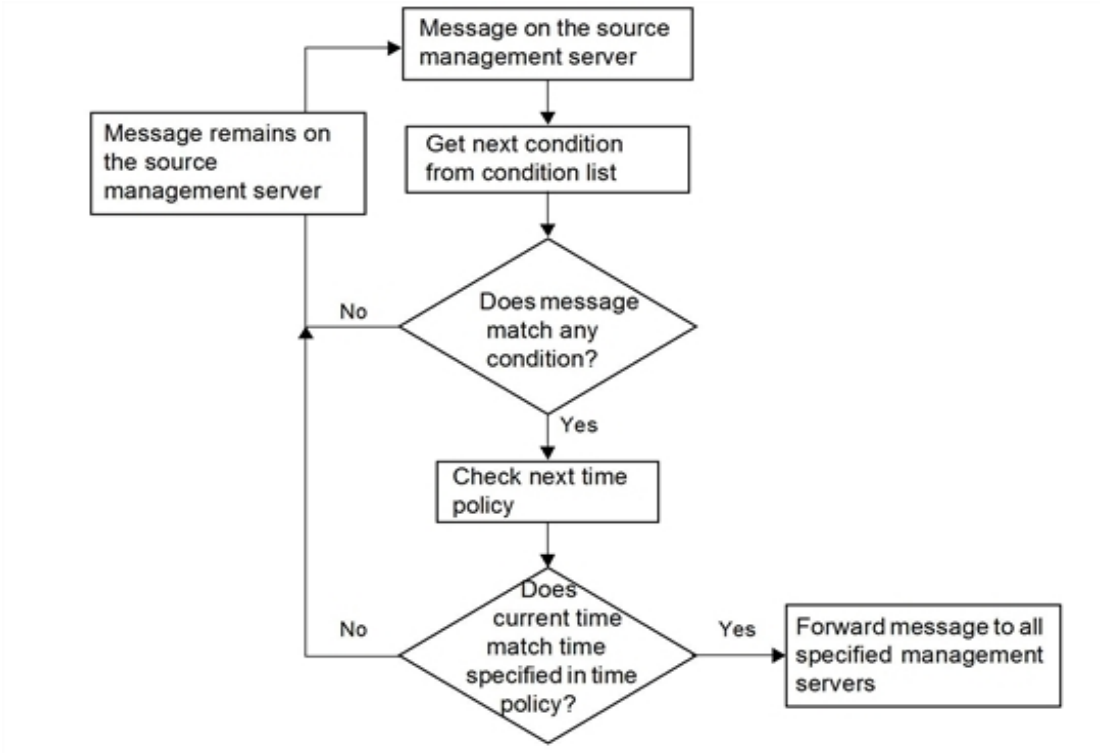
Operation	Normal Messages	Notification Messages
Own/disown	√	-
Modify text/severity	√	√
Add/delete/modify CMA	√	√
Start an operator-initiated action	√	-
Rerun an auto-action	√	-

Message Forwarding Policy

HPOM enables you to create a single message forwarding policy that generates notification messages and switches message control. You assign this policy to the source management server.

Any new message arriving on the source management server is checked before any forwarding action is taken, as shown in Figure 55.

Figure 55: Policy Checking Process



Message Distribution Lists

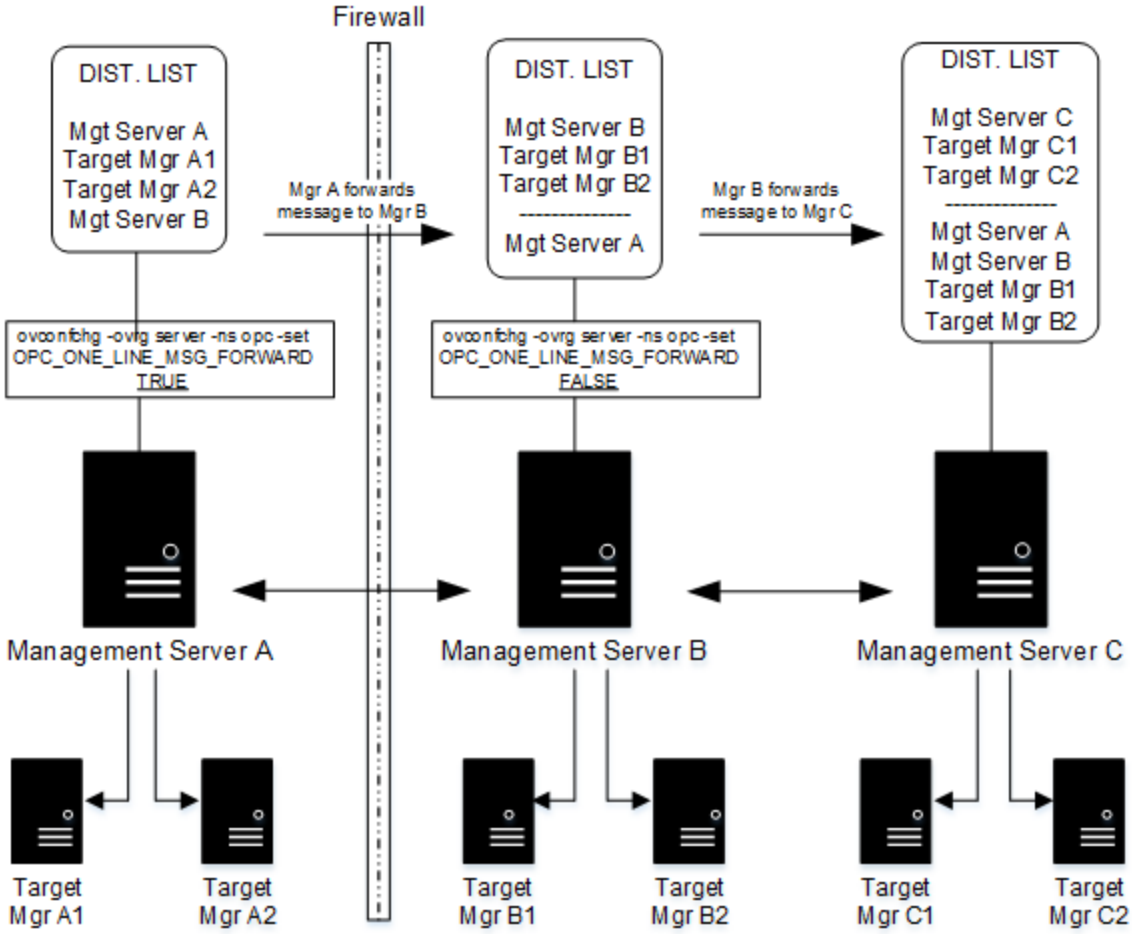
Each forwarded message includes a distribution list of management servers known to the sender of the message. This distribution list is used to inform the listed managers of any subsequent changes that occur to the message (for example, annotations added or actions executed). When a target manager receives a forwarded message, it adds this list to its own distribution list of management servers associated with the message.

Controlling the Size of Distribution Lists

The `OPC_ONE_LINE_MSG_FORWARD` parameter allows you to control the size of the distribution list included with the message as it progresses down a chain of management servers.

In the example shown in Figure 56, if `OPC_ONE_LINE_MSG_FORWARD` is left on its default setting, `FALSE`, management server B would include a list of *all* the target managers it associates with the forwarded message in any message it forwards to management server C. These managers would be added to server C's message-specific distribution list, and the new, larger distribution list would be used if any task were performed by management server C on the message in question. In this case, server B would carry out any necessary action, compare its own distribution list against the one in the message received from server C, and inform only those targets not already informed by C.

Figure 56: Message Forwarding in Larger Hierarchies



Although quicker and more efficient under normal circumstances, this approach has the disadvantage of a message-forwarding operation failing if one or more of the target managers specified in management server B's distribution list is not known to (or is unreachable by) management server C. In the example shown, management server A is known to management server B. However, because of the firewall, management server A is not known to management server C.

However, because management server A is included in the distribution list of management server C, the following is true:

- Management server A does not find out about any changes.
- Management server C will attempt (and fail) to inform A of any changes to messages forwarded originally from A.
- Management server B will assume A has been informed by C.

If the administrator changes the value of the `OPC_ONE_LINE_MSG_FORWARD` parameter on management server B in Figure 56 from `FALSE` to `TRUE`, management server B includes only itself in the distribution list it encloses in any message it forwards to management server C.

Including only management server B reduces the distribution list management server C associates with the same forwarded message from the list shown in Figure 56 to the following:

- Sender of the message (in this case, management server B).
- Target manager of management server C, Target Mgrs C1 and C2.

If management server C subsequently performs a task or adds an annotation to the message, this information is distributed only to the reduced list of management servers specified in server C's distribution list. When management server B receives notification (from management server C) of a change to the message, it carries out any actions specified and then notifies in turn the servers *it* has listed in *its* message-specific distribution list. The disadvantage of this approach is that the chain of servers to be contacted is linear and, as a result, longer and more prone to interruption by network configuration and reliability.

Note: If management servers A and B set `OPC_ONE_LINE_MSG_FORWARD` to `FALSE`, management server C's distribution list includes all the target management servers known to A and B. As a consequence, C attempts to contact all the management servers in the list if any change to the message's state occurs in C's domain.

Connecting Management Servers to Trouble Ticket Systems

Because more than one management server may be connected to the same trouble ticket system, it is possible that the same message may be sent to the same trouble ticket server by more than one management server. This would happen, for example, if the trouble ticket was specified in the policy for a given message, and this message then forwarded to another management server.

Note: If the trouble ticket is enabled for the message from the managed node, a copy of the message is forwarded automatically to the trouble ticket system as soon as it arrives at the management server. However, HPOM allows you to control the passing of both notification and normal messages to the trouble ticket system on any management server to which this message is subsequently forwarded with the parameters `OPC_FORW_NOTIF_TO_TT` (default=`FALSE`) and `OPC_FORW_CRTL_SWITCH_TO_TT` (default=`TRUE`).

Managing Forwarded Messages

Message forwarding is a powerful and integral part of HPOM flexible management. How you configure message forwarding directly affects how it runs in your environment. There are a number of issues you should consider when planning your message forwarding strategy.

Planning Your Message Forwarding Strategy

When planning your message forwarding strategy, consider the following:

- **Managed Nodes**

  All management servers participating in message forwarding must have all managed nodes set up in their respective node banks.

- **Target Managers**

  Before you can execute an operator-initiated action on a normal message, you need to define the target manager to which the message is forwarded as an action-allowed manager on the managed node.

- **Message Ownership**

  When a message is owned or disowned on one management server, the operators on the other management servers are informed only if all operators exist on all management servers.

  If you do not want a management server to send or receive own and disown events, set the following variables on the management server by using the `ovconfchg` command-line tool:

  - `OPC_SEND_OWN_DISOWN FALSE` (default is `TRUE`)
  - `OPC_ACCEPT_OWN_DISOWN FALSE` (default is `TRUE`)

  To find out how to set these variables by using the `ovconfchg` command-line tool, see the *ovconfchg* manual page.

- **Duplicate Messages**

  You can prevent duplicate messages arriving at the trouble ticket server by setting parameters using the `ovconfchg` command-line tool on the management servers. For details, see "Scalable Architecture for Multiple Management Servers" on page 226 as well as the online help for HPOM. For information on `ovconfchg` usage, see the *ovconfchg* manual page.

- **Distribution Lists**

  If the distribution lists for a message contain references to unknown or unreachable management servers, part or all of a message forwarding operation may fail. If you want HPOM to buffer messages until the servers are reachable again, set the `OPC_MSGFORW_BUFFERING` variable to `TRUE` (default is `FALSE`) by using the `ovconfchg` command-line tool. For details about `ovconfchg`, see the *ovconfchg* manual page.

  Caution: Buffer message only in environments with two management servers. If the distribution list contains references to unknown servers, messages are buffered continuously. For details, see "Scalable Architecture for Multiple Management Servers" on page 226.

- **Infinite Loops**

  Never create infinite message loops between servers when implementing message forwarding.

- **Identical Instructions**

  Copies of the same message can sometimes display on different HPOM managers. You need identical instructions, including instruction interface settings on the management servers, to ensure

the correct output of message instructions. One way to ensure identical instructions is to download the instructions from management server A and upload it on management server B.

- **Synchronization Problems**

  Because control switching enables more than one management server to assume responsibility for a message at any one time, the following synchronization problems can occur:

  - *Concurrent Instances*

    Concurrent instances of an operator-initiated action may exist if operators on different management servers trigger the same action at the same time.

  - *Premature Acknowledgement*

    A message on which you are working may be acknowledged by someone else on another management server before you have finished your task.

  - *Unwanted Annotations*

    You may discover message annotations that you did not add.

  - *Unforwarded Annotations*

    The start annotation information for operator-initiated actions is not forwarded between management servers. This annotation contains the start time of the action as well as information about the action itself.

  To avoid these problems at least partially, make sure to own a message before you begin working on it.

- **Communication Failures**

  Communication failures between source and target management servers may affect additional systems further down the communication chain from the target manager. Similar problems may result where a message has already been downloaded from the target manager or has been absorbed into the message-stream interface.

Troubleshooting Problems in the Message Forwarding Policy

If HPOM comes across problems or inconsistencies in the message forwarding policy, it generates an error message and ignores the rest of the policy contents. HPOM also generates an error message on the source management server if the source management server fails to contact its target management servers.

As a rule, HPOM generates errors when the following occurs:

- Policy is set up incorrectly.
- Network-related problems exist.

- Remote or target management server is unreachable.
- Target management server is not configured to receive forwarded messages.

Scalability Scenarios

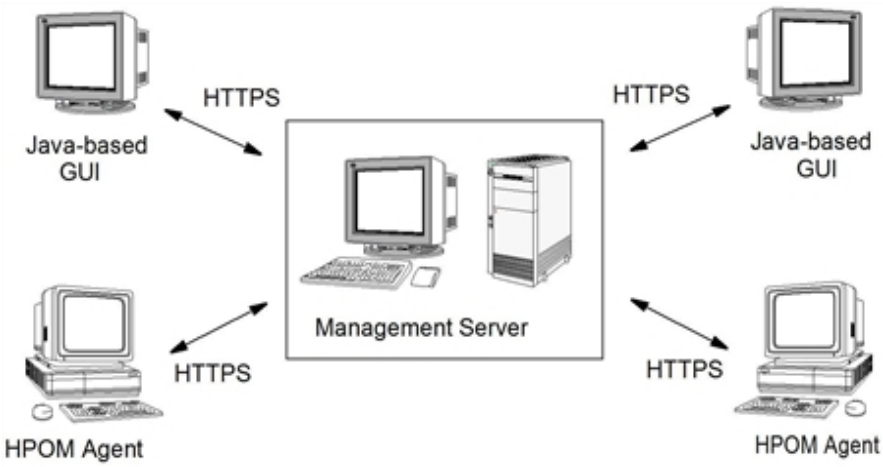
HPOM is designed for deployment in large, complex environments. The flexible architecture of HPOM enables one or more management systems to be combined into a single powerful management solution that meets the requirements of your organizational structure.

This section shows by example how to scale HPOM to meet the particular needs of your organization.

Scenario 1. Single Server Managing a Set of Nodes

The simple scenario illustrated in Figure 57 shows a single HP Operations management server that manages several remote nodes, each running an HP Operations agent. The managed nodes and the management server communicate through the HTTPS protocol. Multiple operators can work together to manage the environment by using the Java-based operator GUI.

Figure 57: Single Management Server Managing a Set of Nodes



This simple architecture provides an efficient solution for managing multiple remote systems from a central location:

- **Variable Thresholds**

  Monitors thresholds of SNMP MIB and custom variables.
- **Message Sources**

  Processes a variety of message sources.
- **Local Events**

Filters local events on managed nodes.

- **Automatic Actions**

  Invokes local automated actions on managed nodes.

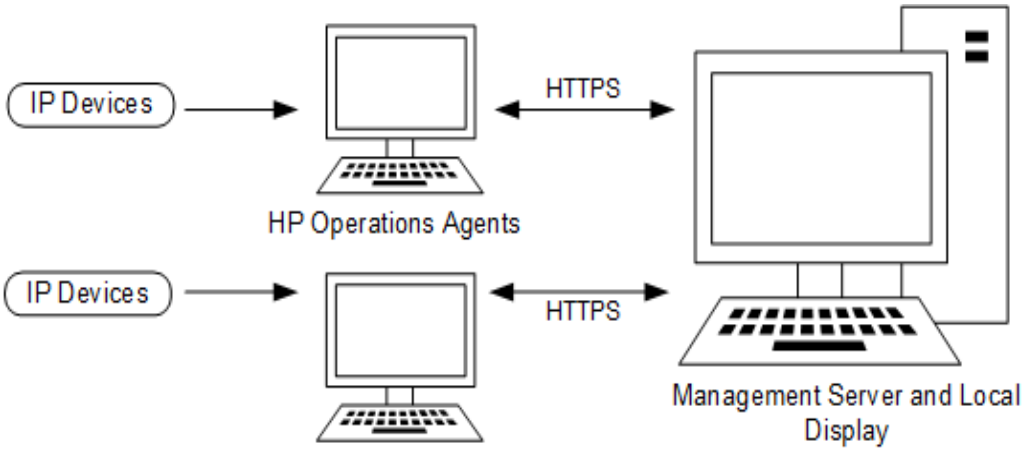
- **Agent Platforms**

  Supports a variety of agent platforms (for example, HP-UX, Linux, AIX, Solaris, and Windows).

Scenario 2. HP Operations Agents Monitoring IP Devices

Figure 58 shows a scenario in which an HP Operations agent acts as a proxy agent, performing SNMP threshold monitoring on remote SNMP devices. In this scenario, a remote managed node with an HP Operations agent can be used to perform threshold monitoring on other SNMP-only devices in the network. Because the HP Operations agent forwards threshold events to the manager only, SNMP polling from the management sever is significantly reduced.

Figure 58: HP Operations Agents Monitoring IP Devices



As part of its standard functionality, HPOM also enables management servers to do the following:

- **Map IP Devices**

  Automatically discover and map any IP device.

- **Poll IP Devices**

  Poll IP devices for their IP status.

- **Receive SNMP Traps**

  Receive SNMP traps from any device.

- **Collect SNMP Trends**

  Collect SNMP trend data (based on MIB variables) from any SNMP device.

Appendix A: Policy Body Grammar

In this Appendix

This appendix provides policy body grammar for the default policy types. The default policy types include:

- Open Message Interface (OPCMSG)
- Log File Entry (LOGFILE)
- Measurement Threshold (ADVMONITOR)
- SNMP Interceptor (SNMP)
- Event Correlation (ECS)
- Scheduled Task (SCHED)
- Service Process Monitoring (ADVMONITOR)
- Windows Management Interface (WBEM)
- Windows Event Log (LOGFILE)

Consider that you cannot use the policy body grammar described in this appendix for editing the following policy types:

- Service Auto Discovery
- Node Info
- ConfigFile
- Event Correlation Composer
- Flexible Management Policy Type
- Remote Action Security Policy Type
- SiteScope Policy Type
- Subagent Policy Type

Policy Body Grammar

Policy body grammar for editing the default policy types is the following:

```
file:           e |
                SYNTAX_VERSION syntax_number |
                file logsource |
```

```
                file snmpsource |
                file csmsource |
                file monsource |
                file advmonsource |
                file schedsource |
                file ecsource |
                file wbemsource

syntax_number:  1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11

logsource:      LOGFILE <string (name)> DESCRIPTION <string (description)> logdefopts
                conditions

snmpsource:     SNMP <string (name)>  DESCRITPION  <string (description)> snmpdefopts
                snmpconditions

csmsource:      OPCMSG <string (name)>  DESCRIPTION  <string (description)> csmdefopts
                conditions

monsource:      MONITOR  <string (name)>  DESCRIPTION  <string (description)> mondefopts
                monconditions

advmonsource:   ADVMONITOR  <string (name)>  DESCRIPTION  <string (description)>
                advmondefaults advmonsourcedef advmonconditions

schedsource:    SCHEDULE  <string (name)>  DESCRIPTION  <string (description)>
                schedsetopts

ecsource:       ECS <string (name)> DESCRIPTION <string (description)> ecopts ecver
                CIRCUIT_FILE <string (file)> circuit

wbemsource:     WBEM  <string (name)>  DESCRIPTION  <string (description)> wbemdefopts
                wbemconditions

logdefopts:     e | logdefopts logdefault | logdefopts logoption | logdefopts
                sourceoption

logdefault:     stddefault | NODE node

logoption:      LOGPATH  <string (path to logfile)> |
                EXEFILE  <string (path to file to execute)> |
                READFILE  <string (path to file containing logfile paths)> |
                INTERVAL  <string (time between logfile checks)> |
                CHSET  <string (character set of the logfile)> |
                FROM_LAST_POS |
                FIRST_FROM_BEGIN |
                NO_LOGFILE_MSG |
                CLOSE_AFTER_READ

snmpdefopts:    e | snmpdefopts stddefault | snmpdefopts sourceoption

snmpconditions: e |
                snmpconditions MSGCONDITIONS snmpmsgconds |
                snmpconditions SUPPRESSCONDITIONS snmpsuppressconds |
                snmpconditions SUPP_UNM_CONDITIONS snmpsupp_unm_conds
```

```
snmpmsgconds:e |
                snmpmsgconds DESCRIPTION  <string (description)> condsuppdupl condition_id
               CONDITION snmpconds SET sets

snmpsuppressconds:e |
                snmpsuppressconds DESCRIPTION  <string (description)> condition_id CONDITION
snmpconds

snmpsupp_unm_conds:e |
                snmpsupp_unm_conds DESCRIPTION  <string (description)> condition_id CONDITION
                snmpconds

snmpconds:      e |
                snmpconds $e  <string (enterprise)> |
                snmpconds $G  <number (generic trap)> |
                snmpconds $S  <number (specific trap)> |
                snmpconds $(<number (variable)>) pattern |
                snmpconds NODE nodelist

csmdefopts:     e | csmdefopts stddefault | csmdefopts sourceoption

mondefopts:     e | mondefopts mondefault | mondefopts monoption | mondefopts
sourceoption

mondefault:     stddefault | NODE node

monoption:      INTERVAL  <string (time between checks)> |
                MONPROG  <string (path to monitor executable)> |
                MIB  <string (MIB variable)> |
                MIB  <string (MIB variable)>  NODE node |
                EXTERNAL |
                MINTHRESHOLD |
                MAXTHRESHOLD |
                GEN_BELOW_THRESHOLD |
                GEN_BELOW_RESET |
                GEN_ALWAYS |
                AUTOMATIC_MSGKEY

monconditions:  e |
                monconditions MSGCONDITIONS monmsgconds |
                monconditions SUPPRESSCONDITIONS monsuppressconds |
                monconditions SUPP_UNM_CONDITIONS monsupp_unm_conds

monmsgconds:    e |
                monmsgconds DESCRIPTION  <string> condition_id CONDITION monconds SET sets

monsuppressconds:e |
                monsuppressconds DESCRIPTION  <string> condition_id CONDITION monconds

monsupp_unm_conds:e |
                monsupp_unm_conds DESCRIPTION  <string> condition_id CONDITION monconds

monconds:       e |
                monconds THRESHOLD numval duration |
```

```
                monconds RESET numval |
                monconds OBJECT pattern

advmondefaults: e | advmondefaults sourceoption | advmondefaults stddefault |
advmondefaults
                NODE node | advmondefaults advmonoption

advmonoption:   INTERVAL  <string (time between checks)> |
                INSTANCEMODE ALL | INSTANCEMODE SAME | INSTANCEMODE ONCE |
                MULTISOURCE |
                INSTANCERULES |
                AUTOMATIC_MSGKEY |
                AUTOMATIC_MSGKEY  <string (default message key)> |
                MINTHRESHOLD |
                MAXTHRESHOLD |
                GEN_BELOW_THRESHOLD |
                GEN_BELOW_RESET |
                GEN_ALWAYS |
                SCRIPTTYPE  <string (type of script)> |
                DDF DATASOURCE  <string> |
                DDF OBJECT  <string>

advmonsourcedef:e |
                advmonsourcedef PROGRAM  <string (name)>  DESCRIPTION  <string (description)>
                advmonprog |
                advmonsourcedef EXTERNAL  <string (name)>  DESCRIPTION  <string (description)>
                ddf |
                advmonsourcedef NTPERFMON  <string (name)>  DESCRIPTION  <string (description)>
                advmonperfmon |
                advmonsourcedef SNMP  <string (name)>  DESCRIPTION  <string (description)>
                advmonsnmp |
                advmonsourcedef MEASUREMENT  <string (name)>  DESCRIPTION  <string (description)>
                advmonme |
                dvmonsourcedef CODA  <string>  DESCRIPTION  <string (description)> advmonme |
                advmonsourcedef WBEM  <string (description)>  DESCRIPTION  <string (description)>
                advmonwbem

advmonprog:     MONPROG  <string (path to executable)> ddf

advmonperfmon:  OBJECT  <string (name)>  COUNTER  <string>  INSTANCE  <string> ddf

advmonsnmp:     MIB  <string (MIB variable)> ddf

advmonme:       COLLECTION  <string> metrics |
                COLLECTION  <string>  GUID  <string (UUID)> metrics |
                DATASOURCE  <string>  COLLECTION  <string> metrics

advmonwbem:     NAMESPACE  <string>  CLASS  <string>  ATTRIBUTE  <string>
                instancefilter ddf |
                WMI_USERNAME  <string>  WMI_PASSWORD  <string>
                NAMESPACE  <string>  CLASS  <string>  ATTRIBUTE  <string> instancefilter ddf

instancefilter: e | INSTANCE_FILTER  <string>
```

```
ddf             DDF DATASOURCE  <string>  OBJECT  <string>  METRIC  <string>

metrics:        e |
                metrics METRIC  <string> metricguid useforinstance

metricguid:     e | GUID  <string (UUID)>

useforinstance: e | USEFORINSTANCE

advmonconditions:e |
                advmonconditions tMSGCONDITIONS advmonmsgconds |
                advmonconditions tSUPPRESSCONDITIONS advmonsuppressconds |
                advmonconditions tSUPP_UNM_CONDITIONS advmonsupp_unm_conds

advmonmsgconds:e |
                advmonmsgconds instancerule tDESCRIPTION  <string (description)> condition_id
                CONDITION advmonconds advmonmsgsets

instancerule:   e |
                INSTANCERULE  <string>  ID  <string> |
                INSTANCERULE  <string>

advmonmsgsets:e |
                advmonmsgsets SETSTART sets |
                advmonmsgsets SETCONT sets |
                advmonmsgsets SETEND sets

advmonsuppressconds:e |
                advmonsuppressconds DESCRIPTION  <string> condition_id CONDITION advmonconds

advmonsupp_unm_conds:e |
                advmonsupp_unm_conds DESCRIPTION  <string> condition_id CONDITION advmonconds

advmonconds:    e |
                advmonconds THRESHOLD numval duration |
                advmonconds THRESHOLD condscript duration |
                advmonconds RESET numval |
                advmonconds RESET condscript |
                advmonconds OBJECT pattern |
                advmonconds OBJECT condscript

condscript:     SCRIPTTYPE  <string>  SCRIPT  <string> |
                SCRIPT  <string>

duration:       e | FOR  <string (condition duration)>

numval:         <integer number> | <floating number>

schedsetopts:   e |
                schedsetopts DISABLED |
                schedsetopts TEMPLATE_ID  <string (UUID of the template)> |
                schedsetopts VERSION  <number> |
                schedsetopts SCRIPTTYPE  <string (type of script)>  SCRIPT  <string (actual
script)> |
                schedsetopts SCHEDPROG  <string (path to executable to run)> |
                schedsetopts USER  <string (username)> |
```

```
                schedsetopts USER  <string (username)>  PASSWORD  <string (password)> |
                schedsetopts MONTH  <string (month)> |
                schedsetopts MONTHDAY  <string (day in the month)> |
                schedsetopts WEEKDAY  <string (day in the week)> |
                schedsetopts HOUR  <string (hour)> |
                schedsetopts MINUTE  <string (minute)> |
                schedsetopts TIMEZONE_VALUE  <string (timezone)> |
                schedsetopts YEAR  <number> |
                schedsetopts INTERVAL  <string (time between actions)> |
                schedsetopts LOGLOCAL |
                schedsetopts SEND_OUTPUT |
                schedsetopts TIMEZONE_TYPE tz_type |
                schedsetopts BEFORE SET sets |
                schedsetopts FAILURE SET sets |
                schedsetopts SUCCESS SET sets

ecopts:         e |
                ecopts DISABLED |
                ecopts TEMPLATE_ID  <string (UUID of the template)> |
                ecopts VERSION  <number (template version)>  |
                ecopts ECS_LOG_INPUT |
                ecopts ECS_LOG_OUTPUT

ecver:          VERIFIED | UNVERIFIED

wbemdefopts:    e |
                wbemdefopts wbemdefault |
                wbemdefopts wbemoption |
                wbemdefopts sourceoption

circuit:        e | circuit <string>

wbemdefault:    stddefault | NODE node

wbemoption:     NAMESPACE  <string (WBEM namespace)> |
                CLASS  <string (WBEM class)> |
                WITHIN  <string (interval)> |
                WHERE_CLAUSE  <string (where clause)> |
                QUERY_LANGUAGE  <string (language for query)> |
                QUERY  <string (query)> |
                INSTANCE_CREATION_EVENT |
                INSTANCE_MODIFICATION_EVENT |
                INSTANCE_DELETION_EVENT |
                CLASS_CREATION_EVENT |
                CLASS_MODIFICATION_EVENT |
                CLASS_DELETION_EVENT |
                NAMESPACE_CREATION_EVENT |
                NAMESPACE_MODIFICATION_EVENT |
                NAMESPACE_DELETION_EVENT |
                INTERVAL  <string (interval)>

wbemconditions: e |
                wbemconditions MSGCONDITIONS wbemmsgconds |
```

```
                wbemconditions SUPPRESSCONDITIONS wbemsuppressconds |
                wbemconditions SUPP_UNM_CONDITIONS wbemsupp_unm_conds

wbemmsgconds:   e |
                wbemmsgconds DESCRIPTION  <string (description)> condsuppdupl condition_id
                CONDITION wbemconds SET sets

wbemsuppressconds:e |
                wbemsuppressconds DESCRIPTION  <string> condition_id CONDITION wbemconds

wbemsupp_unm_conds:e |
                wbemsupp_unm_conds DESCRIPTION <string (description)> condition_id CONDITION
                wbemconds

wbemconds:      e |
                wbemconds <string (condition name)>  ~= pattern |
                wbemconds <string (condition name)> wbemop wbemval

wbemop:         == |  != | >= | > | < | <=

wbemval:        <string> | <number (floating)> | <number (int)>

condefopts:     e |
                condefopts stddefault |
                condefopts sourceoption

conditions:     e |
                conditions MSGCONDITIONS msgconds |
                conditions SUPPRESSCONDITIONS suppressconds |
                conditions SUPP_UNM_CONDITIONS supp_unm_conds

msgconds:       e |
                msgconds DESCRIPTION  <string> condsuppdupl condition_id CONDITION conds SET sets

suppressconds:  e |
                suppressconds DESCRIPTION  <string> condition_id CONDITION conds

supp_unm_conds: e |
                supp_unm_conds DESCRIPTION  <string> condition_id CONDITION conds

condsuppdupl:   e |
                SUPP_DUPL_COND suppdupl |
                SUPP_DUPL_IDENT suppdupl |
                SUPP_DUPL_IDENT_OUTPUT_MSG suppdupl

conds:          e |
                conds SEVERITY severities |
                conds NODE nodelist |
                conds APPLICATION  <string> |
                conds MSGGRP  <string> |
                conds OBJECT  <string> |
                conds TEXT pattern

suppdupl:       <string> |
                <string>  RESEND  <string> |
                <string>  COUNTER_THRESHOLD  <number> |
```

```
                <string> COUNTER_THRESHOLD <number> RESET_COUNTER_INTERVAL <string> |
                <string> RESEND <string> COUNTER_THRESHOLD <number> |
                <string> RESEND <string> COUNTER_THRESHOLD <number>
                RESET_COUNTER_INTERVAL <string> |
                COUNTER_THRESHOLD <number> |
                COUNTER_THRESHOLD <number> RESET_COUNTER_INTERVAL <string>

stddefault:     SEVERITY severity |
                APPLICATION <string> |
                MSGGRP <string> |
                OBJECT <string> |
                SERVICE_NAME <string> |
                MSG_KEY <string> |
                HELPTEXT <string (instruction text)> |
                HELP <string (instruction UUID)> |
                INSTRUCTION_TEXT_INTERFACE <string> |
                INSTRUCTION_PARAMETERS <string>

sourceoption:   LOGMATCHEDMSGCOND | LOGMATCHEDSUPPRESS | LOGUNMATCHED |
FORWARDUNMATCHED |
                UNMATCHEDLOGONLY | MPI_SV_COPY_MSG | MPI_SV_DIVERT_MSG | MPI_SV_NO_OUTPUT |
                MPI_AGT_COPY_MSG | MPI_AGT_DIVERT_MSG | MPI_AGT_NO_OUTPUT |
                MPI_IMMEDIATE_LOCAL_ACTIONS | ICASE | DISABLED |
                SUPP_DUPL_COND suppdupl |
                SUPP_DUPL_IDENT suppdupl |
                SUPP_DUPL_IDENT_OUTPUT_MSG suppdupl |
                SEPARATORS <string> |
                TEMPLATE_ID <string> |
                TEMPLATE_VERSION <number>

severities:     e | severities severity

severity:       Unknown | Normal | Warning | Critical | Major | Minor

nodelist:       nodelist node | node

node:           IP <string (IP address)> |
                IP <string (IP address)> <string (node name)> |
                OTHER <string (variable or other)>

tz_type:        MGR_LOCAL | AGT_LOCAL | FIX

sets:           e | sets set

set:            SEVERITY severity |
                NODE node |
                APPLICATION <string (application to which message relates)> |
                MSGGRP <string (message group)> |
                OBJECT <string (object to which message relates)> |
                MSGTYPE <string (type of message)> |
                TEXT <string (message text)> |
                SERVICE_NAME <string (name of the service to which the message relates)> |
                MSGKEY <string (message key)> |
                MSGKEYRELATION ACK pattern |
```

```
                    CUSTOM  <string (name of the custom attribute)> <string (value of the
                    custom attribute)> |
                    SERVERLOGONLY |
                    AUTOACTION action |
                    OPACTION action |
                    TROUBLETICKET acknowledge |
                    NOTIFICATION |
                    MPI_SV_COPY_MSG |
                    MPI_SV_DIVERT_MSG |
                    MPI_SV_NO_OUTPUT |
                    MPI_AGT_COPY_MSG |
                    MPI_AGT_DIVERT_MSG |
                    MPI_AGT_NO_OUTPUT |
                    MPI_IMMEDIATE_LOCAL_ACTIONS |
                    HELPTEXT  <string (text of the instruction message)> |
                    HELP  <string (UUID of the stored instruction message)> |
                    INSTRUCTION_TEXT_INTERFACE  <string (name of instruction text interface)> |
                    INSTRUCTION_PARAMETERS <string (parameters for instruction text interface)>

condition_id:   e | CONDITION_ID  <string (UUID)>

action:         <string (path to executable)> actionnode annotate acknowledge msgsendmode
                signature

actionnode:     e | ACTIONNODE node

acknowledge:    e | ACK

msgsendmode:    e | SEND_MSG_AFTER_LOC_AA msgsendok msgsendfailed

msgsendok:      e | SEND_OK_MSG logonly

msgsendfailed:  e | SEND_FAILED_MSG

logonly:        e | LOGONLY

signature:      e | SIGNATURE  <string (signature)>

pattern:        <string> separators icase

separators:     e | SEPARATORS  <string (separators)>

icase:          e | ICASE

chset:          e | ASCII | ACP1250 | ACP1251 | ACP1252 | ACP1253 | ACP1254 | ACP1255 |
                ACP1256 |
                ACP1257 | ACP1258 | NT_ANSI_JP | NT_OEM_JP | ACP874 | NT_OEM_L1 | NT_ANSI_LP |
                NT_OEM_US | NT_UNICODE | OEMCP437 | OEMCP720 | OEMCP737 | OEMCP775 | OEMCP850 |
                OEMCP852 | OEMCP855 | OEMCP857 | OEMCP860 | OEMCP861 | OEMCP862 | OEMCP863 |
                OEMCP864 | OEMCP865 | OEMCP866 | OEMCP869 | OEMCP932 | ROMAN8 | ISO8859 |
                ISO88591 | ISO885910 | ISO885911 | ISO885913 | ISO885914 | ISO885915 | ISO88592 |
                ISO88593 | ISO88594 | ISO88595 | ISO88596 | ISO88597 | ISO88598 | ISO88599 |
                TIS620 | UCS2 | EBCDIC | SJIS | EUC | EUCJP | EUCKR |
                EUCTW | GB2312 | BIG5 | CCDC | UTF8
```

Send Documentation Feedback

If you have comments about this document, you can contact the documentation team by email. If an email client is configured on this system, click the link above and an email window opens with the following information in the subject line:

Feedback on HPOM Concepts Guide (Operations Manager 9.22)

Just add your feedback to the email and click send.

If no email client is available, copy the information above to a new message in a web mail client, and send your feedback to docfeedback@hpe.com.

We appreciate your feedback!


Java™
COMPATIBLE