HP OpenView Service Desk

Support Guide

Software Version: 5.10

For the Windows® and UNIX® Operating Systems

Manufacturing Part Number: None

Document Release Date: December 2006

Software Release Date: December 2006

Legal Notices

Contents

1. Service Desk Architecture and Components

Components Used by Service Desk 16
Protocols 21
Licensing. 23
Service Desk Login Authentication. 25
Object Server 27
  Rules 27
SLM Server and Metric Adapters 30
Impacted Services 33
Helpdesk Deadline Calculation and SLA retrieval 35
  Time Zone 35
  Support Hours. 36
  Duration 36
  Calculation 37
  Helpdesk Deadline Calculation with SLM 37
    Summary 38
  Service and SLA Retrieval Settings for Service Call 38
  Service and SLA Retrieval Settings for Incident 40
  Setting an Impact-Priority Mapping in the SLA 42
  Setting Support Hours in the SLA. 42
  Setting Support Hours per Impact 43
  Support Hours in Support Organization's Time 43
  Support Hours Relative to the Customer in the SLA. 45

2. Performance Tuning

Query Restrictions 48
  Symptoms of Retrieving Many Records. 48
  View Filters 48
  Setting a Default Query Restriction 48
  Setting Role Query Restrictions. 49
Optimizing Forms 51
Database and UI Rules 52
Folders 54
Roles 55
Service Call and Incident Queries. 56
  Queries from Service Calls. 56

Contents

Queries from Incidents . . . 58
Java Heap Size . . . 59
Service Desk Server and Database . . . 61
Service Desk Database . . . 62
Reduce Network Latency . . . 63
Stop Unnecessary Server Processes . . . 64
Service Desk Server . . . 65
Increase the Number of Server Instances . . . 66

3. Logging
User and Environment Variables Related to Log Files . . . 69
Service Desk Installation Log Files . . . 71
Searching Installation Logs for Errors . . . 73
XPL Log Files . . . 74
About Binary and Text Log Files . . . 74
Naming Log Files . . . 75
Relating Log File Names to Processes . . . 76
Tracking Log File Generation . . . 78
Logging in UNIX . . . 78
Logging in Windows . . . 80
Configuring the Level of Logging . . . 81
Controlling the Size of Log Files . . . 82
Other Log File Settings . . . 83
Location of Configuration and Log Files . . . 86
Switching Logging Off and On . . . 87
Viewing the Contents of a Binary Log File . . . 89

4. HP OpenView Tracing Fundamentals
Troubleshooting and Tracing . . . 94
HP OpenView Tracing . . . 95
Tracing Architecture . . . 95
Tracing Tools . . . 96
Trace Configuration Files . . . 97
TCF Version . . . 98
APP . . . 98
SINK . . . 99

TRACE 101
Sample Trace Configuration Files 102
Trace Server Log Information 103
Tracing Methods 104
How to Trace 106

5. Installing and Configuring HP OpenView Tracing

Auto Installation of Tracing Tools 110
Trace Server Startup on UNIX-based systems 110
Trace Server Startup on Windows Operating Systems 111
Setting Up Tracing 112
Local Tracing Options 118
Local Dynamic Tracing Options 120
AppName.tcf File Contents 120
Establishing Local Dynamic Tracing 121
Local Static Tracing Options 121
AppName.tcf File Contents 122
Establishing Local Static Tracing 123
Enabling Static Tracing 124
Enabling Dynamic Tracing 127
Tracing Guidelines 130
Known Limitations 131

6. Tracing Processes

Tracing Processes in Service Desk 134
Tracing on UNIX 134
Tracing on Windows 135

7. Configuring Server Processes

HP OpenView Process Control Component 140
The ovc Control Tool 140
The ovcd Control Daemon 140
Control Configuration 142
Troubleshooting Control Error Messages 143
(ctrl-7) Error in the Target Component 143

Cause 143
Solution 143
Security Error when Executing the <Start|Stop|Status|Notify> Method 144
Cause 144
Solution 145
Communication Error when Executing the <Start|Stop Notify|Status> method. 145
Cause 145
Solution 145
(ctrl-4) No Component Matches Target 145
Cause 145
Solution 145
No Target Component Subscribed for this Event 146
Cause 146
Solution 146
Error in One of the Target Prerequisites. 146
Cause 146
Solution 146
Not Authorized 147
Not Authenticated 147
Control Daemon is Running but Cannot be Accessed Remotely 148
Cause 148
Solution 148
Control Daemon is Being Initialized 148
Cause 148
Solution 148
Error. Check Log File for More Information 148
Cause 148
Solution 148
Using the Config Component 149
Using Certificates 150
Tracing 151

8. Testing and Troubleshooting the Service Desk Agent
Testing the SD Agent. 157
SD Agent Does Not Execute Required Actions. 158
Solution 158

Agent Does Not Launch a Program with a GUI . . . 160
Solution . . . 160

9. Object Loader
Object Loader Internals . . . 162
HTTP Post . . . 163
Import Mappings . . . 164
Value Lists . . . 164
HTTP Response . . . 165
Command-Line Arguments and Configuration File Parameters . . . 165
The ONFAIL Option . . . 166
Queuing . . . 168
Architecture of the Queuing Tools . . . 168
Implementation of the Queuing Tools . . . 169
Implementing Event Forwarding with Object Loader . . . 171
Introduction . . . 171
Installing Object Loader . . . 171
Exit Codes . . . 172
Errors and Logging . . . 172
CLIENT_RESPONSE Field in Log File . . . 173
SERVER_RESPONSE Field in Log File . . . 173
Client Error Messages . . . 173

10. Troubleshooting Data Exchange
Troubleshooting Export . . . 178
Troubleshooting Import . . . 179

11. Troubleshooting the SLM Module
SLM Module Components . . . 184
Runtime Information: the Dump Command . . . 185
Propagation of Changes (Attribute Change Control Management) . . . 187
SLM Module Component-Specific Troubleshooting . . . 191
Metric Adapters . . . 191
Common Metric Adapters . . . 191
OVIS Metric Adapter . . . 194

OVPM Metric Adapter 195
OVSN Metric Adapter 196
OVSD Metric Adapter 197
Open Metric Adapter 197
MA Simulator 198
SLM Reporting 199
Dimension Exporter 199
Reporting Data Feeder 199
Reporting Dimension Manager 200
SLM Report Pack. 200
Service Desk Reporting 201
Entity Exporter 201
Report Pack 201
Configuration Server 202
Calculation Engine / Data Collector / Data Exporter 203
Alarm Engine 205
Business Logic. 206
Service Designer 207
Licensing 207
SLM Module Error Messages 208
Example Case - Troubleshooting SLM and Metric Adapters 215
Customer's Description of the Case 215
Customer Environment 215
Additional Information 216
Initial Checks and Investigations 216
Log Files 217
Attempting to Resolve the Problem 218
Proposed Solution and Test Results 220
Results and Conclusion 222

12. Troubleshooting Service Desk OVO for UNIX Integration

Management Server Does Not Start After Deploying OVO for UNIX Agent 227
Explanation 227
Workaround. 227
OVO for UNIX Agent does not Communicate with the OVO Management Server . . 228
Explanation 228

Workaround. 228
Tomcat Service Conflicts 230
Explanation 230
Workaround. 230

13. Service Desk Coexistence Issues
Coexistence – Service Desk and HP OpenView Operations 235
Service Desk 5.0 235
Service Desk 5.0 Service Pack 1. 236
Service Desk 5.0 SP1 and OVO for UNIX 8.1 on Windows 2000 AS – Unable to Start SD Management Server. 237
Service Desk 5.0 SP1 and OVO for UNIX 8.1 on Windows 2000 AS – Conflicting or Missing Certificates 238

14. Troubleshooting Tools
Service Desk Support Tool. 240
Support Log File 240
Support Zip File 242
Run the Support Tool 243
Other Service Desk Support Tools 244
OVII Inventory Tool 244
Integrity Check Tool 244
Service Desk Version Tool 244
Component Library Version Tool 245

Documentation Updates

This manual's title page contains the following identifying information:

- 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, visit the following URL:

http://ovweb.external.hp.com/lpe/doc_serv/

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

For an overview of the latest information on supported platforms, installation prerequisites, software deliverables, and so on, refer to the *Service Desk 5.10 Release Notes*.

Support

You can visit the HP OpenView support web site at:

www.hp.com/managementsoftware/support

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

HP OpenView online software 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 site to:

- Search for knowledge documents of interest
- Submit enhancement requests online
- Download software patches
- Submit and track progress on support cases
- Manage a support contract
- Look up HP support contacts
- Review information about available services
- Enter 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 log in. Many also require a support contract.

To find more information about access levels, refer to the following URL:

www.hp.com/managementsoftware/access_level

To register for an HP Passport ID, go to the following URL:

**www.managementsoftware.hp.com/
passport-registration.html**

1 Service Desk Architecture and Components

This chapter describes the internal architecture and components of Service Desk.

Components Used by Service Desk

Service Desk uses components from the Common Management Environment (CME), as illustrated in Figure 1-1.

Figure 1-1 Components Used by Service Desk

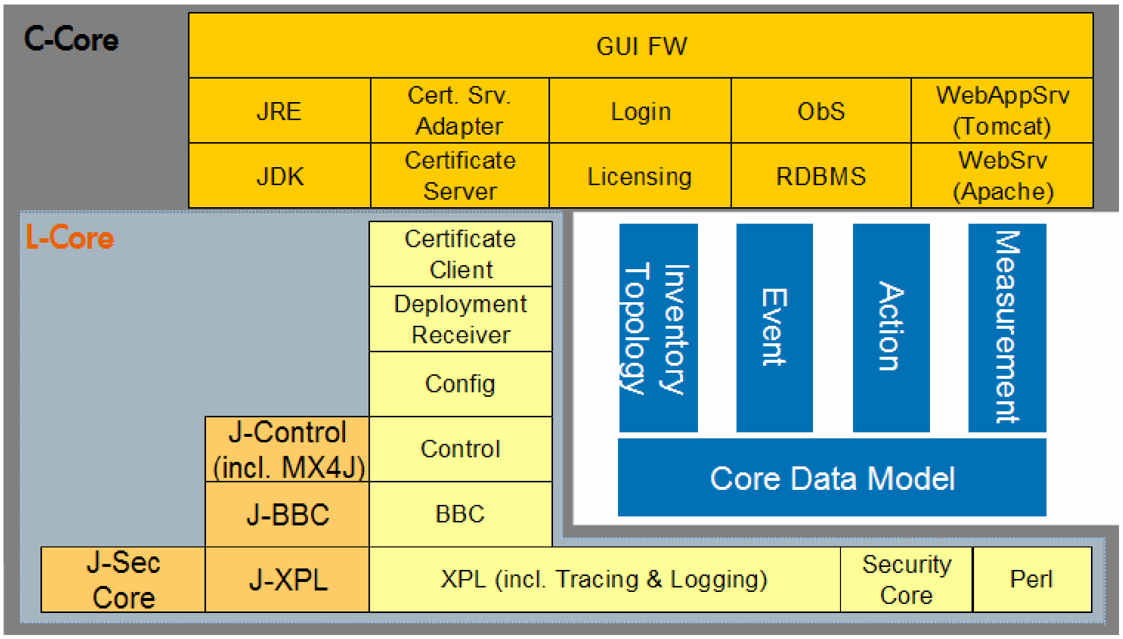


Table 1-1 gives a brief description of the components used in Service Desk.

Table 1-1 Components Used in Service Desk

Component ID	Component Name	Part of Component
`Adm`	Administrator Console	Object Server
Determines your view and the available options for the content pane when you select a workspace from OV Configuration in the Service Desk client console.		
`Apache(A)`	Apache Web Server	Third party
Displays on the Service Desk server as the process `ovapacheA`. Depends on the process `ovtomcatA` (see Tomcat(A) component). Registered as the HP OpenView Apache(A) WebServer service. Used to access the Web portal at **http://server/ovconsole** for installation of the Web Start Service Desk client for the Service Desk online help.		
`BBC`	Black Box Communications	L-Core
Displays on the Service Desk server as the process `ovbbccb` (BBC Communication Broker). Facilitates message communication between components. Uses SnF (store and forward), which can be tuned with `bbcutil` (see the HP OpenView Service Desk online help). Can use SSL for security. Can process SOAP messages, executed with `bbcrpcserver` (BBC Remote Procedure Call server). BBC is primarily used for communication between Metric Adapters and the SLM Server.		
`CDM`	Core Data Model	CME
A module in the Object Server that is used and extended by the Service Desk module. CDM defines generic items like persons, workgroups, organizations, configuration items, services, locations and so on.		

Table 1-1 Components Used in Service Desk (Continued)

Component ID	Component Name	Part of Component
`Ctrl`	Control	L-Core
The `ovc` command-line utility and the HP OpenView Ctrl service (also the HP OpenView Ctrl Daemon (`ovcd`)).		
`GUI FW`	GUI Framework	CME
The Service Desk client GUI.		
`OVII`	HP OpenView Install Initiative	CME
The HP OpenView application installation tool.		
`JdkA`	Java Development Kit	Third party
`Jmc`	Java Mirror Control	part of Control
Java standard interface for starting and stopping Java processes. Used by Control.		
`JRE`	Java Runtime Environment	Third party
`Lic`	Autopass Licensing	CME
Used to manage licenses. Object Server interfaces with this component (copies and reads files from Autopass) and stores license information in the database. Object Server enforces the licenses, Autopass registers them.		
`ObS`	Object Server	CME
The core of the Service Desk server: the management server. Comparable to a J2EE server but with more options (for example, data exchange, Advanced Find, database and UI rules). Part of the Object Server is also used in the Service Desk client. This allows for minimum network traffic (in part provided by the ITP protocol). Other utilities provided by this component are the SD Agent, LoadObject, Exporter and Importer, identified by files with the prefix `OVObs` and located in the installation bin directory.		

Table 1-1 Components Used in Service Desk (Continued)

Component ID	Component Name	Part of Component
`PerlA`	Perl compiler	Third party
`Sched`	Scheduler	CME
Defines the standard schedule. Used when defining Service Hours and Support Hours. It is a base CME component but is not shown in Figure 1-1, “Components Used by Service Desk”.		
`SD`	Service Desk module	SD
A module in the Object Server that defines the business logic and items that comprise Service Desk.		
`SecC`	Security Certificates	L-Core
Displays on the Service Desk server as the process `ovcs`. This process runs on one server only. Installed with Primary Server installation. The `ovcs` process validates certificates received from other Service Desk servers, clients, and agents.		
`SecCo`	Security Core	L-Core
`SecLogin`	Secure Login	CME
Displays on the Service Desk server as the process `ovloginsv`. Used when logging in with Java Authentication and Authorization Service (JAAS), Active Directory, or LDAP. Based on the JAAS standard as the extension interface, SEC Login adds the ability to distribute the collection and validation of user credentials across different systems, overcoming the limitations and security issues when using JAAS on an untrusted client machine.		

Table 1-1 Components Used in Service Desk (Continued)

Component ID	Component Name	Part of Component
`SLM`	Service Level Management module	SD
1) A module in the Object Server that defines the business logic for Service Designer, Service Manager (the Metric items), Customer Relationship Manager, and SLM Administrator. 2) Displays on the Service Desk server as the process `ovsdslm`. Collects information sent by metric adapters. Runs on one server only. 3) Metric Adapter processes displayed on the Service Desk server as `ovsdma`, `openma`, `ovsnma`, `ovpmma` and `ovisma`. These processes must be suitably configured. Metric Adapters (monitoring applications) collect operational data from various sources. 4) Displays on the Service Desk server as the process `ovsdreport`. Runs on one server only. Communicates with OVPI server to provide reporting data.		
`Tomcat(A)`	Tomcat servlet container	Third party
Displays on the Service Desk server as the process `ovtomcatA`. Also registered as HP OpenView Tomcat(A) Servlet Container service. Used by Apache(A).		
`XPL`	Cross Platform Library	L-Core
Defines the interface for tracing and logging (uses the standard Java logging interface). Displayed on the Service Desk server as the HP OpenView Shared Trace service. Trace GUI is also part of this component.		

Protocols

Figure 1-2 shows an overview of the network protocols used by Service Desk components.

Figure 1-2 Network Protocols

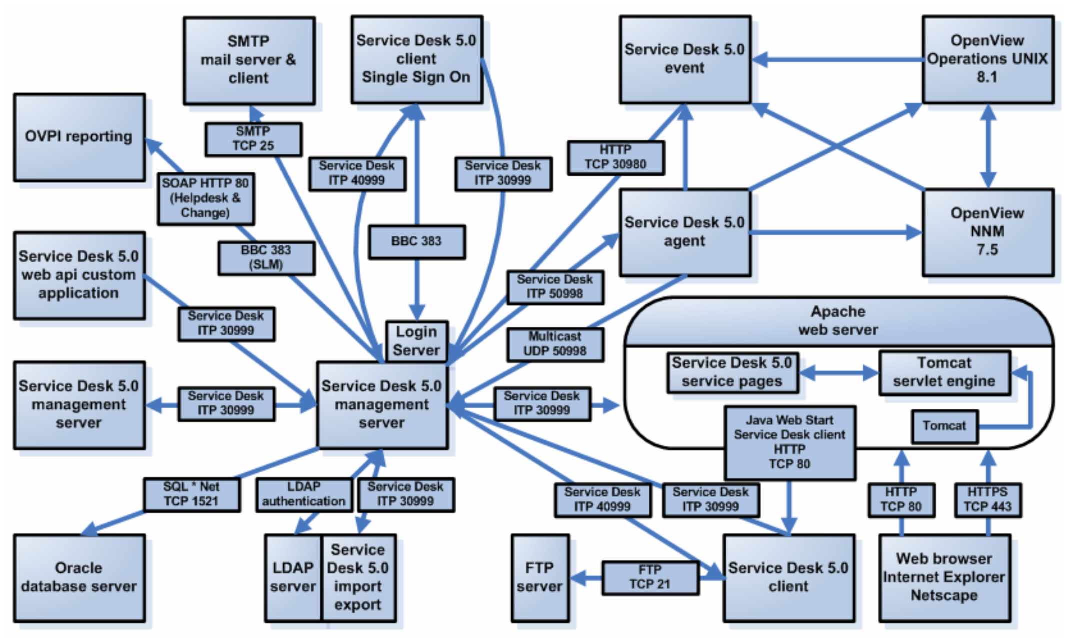
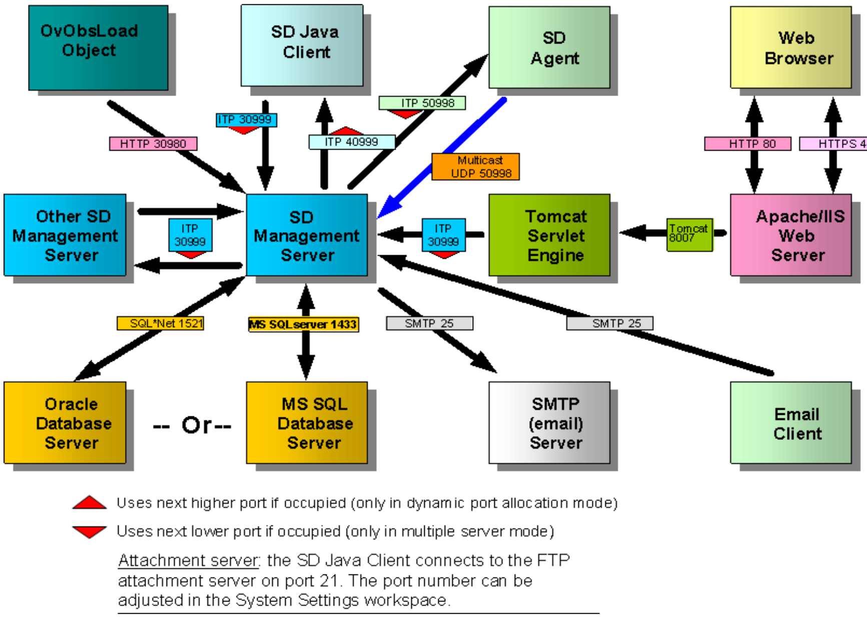


Figure 1-3 shows port number allocations for Service Desk in a firewall environment.

Figure 1-3 Service Desk in a Firewall Environment



Most port numbers can be configured, except the port for the SD agent. The firewall environment must be adapted for non-default port numbers. The SD Java Client connects to the FTP attachment server on port 21.

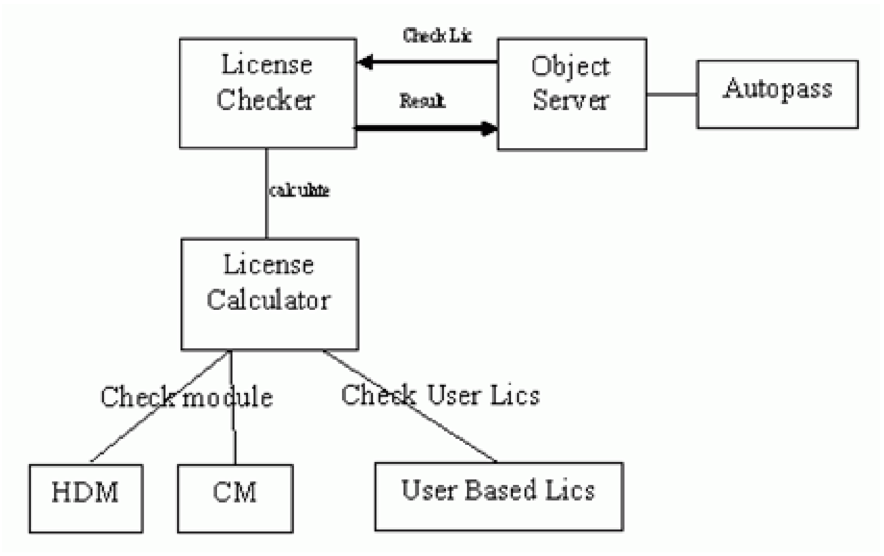
To configure port numbers, open the workspace:

Attachment Settings → **System Settings**

Licensing

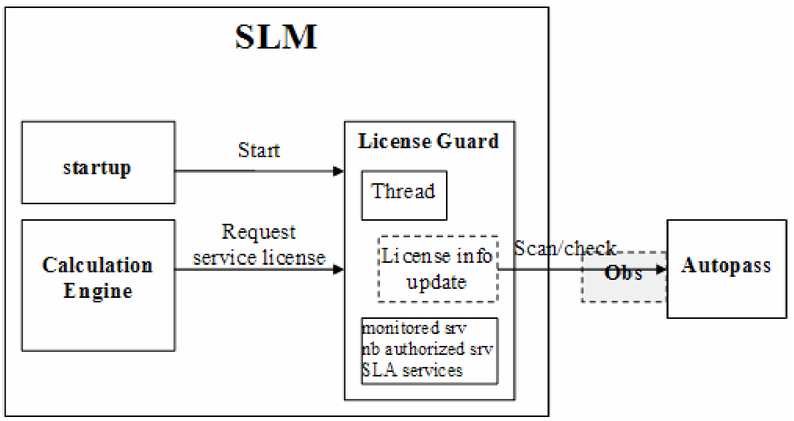
License management is handled by the Autopass component. The Object Server reads the licenses registered by Autopass and enforces them, as illustrated in Figure 1-4.

Figure 1-4 **Autopass License Management**



SLM licenses perform additional checks, as illustrated in Figure 1-5.

Figure 1-5 **SLM Licensing**



The following checks are executed to check the number of licenses used:

Service license criteria:

- Associated to an SLA
- Service is Active:
  - Management status = managed
  - Hierarchy status = configured
  - Management status of the SLA = managed
- Monitored by at least one non-OVSD metric

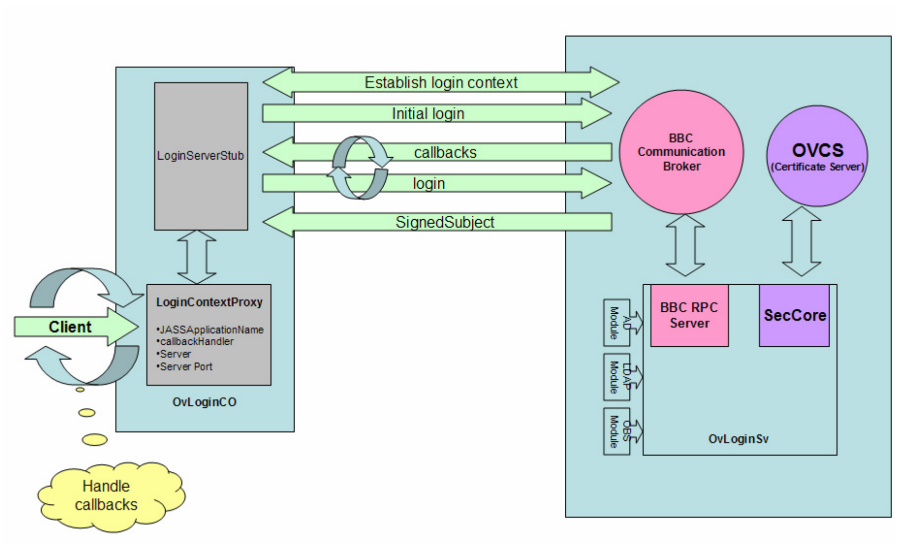
The Service license criteria are computed by the SLM server process `ovsdslm` at startup.

NOTE Open Metric Adapter and HP OpenView Metric Adapters do not need a license.

Service Desk Login Authentication

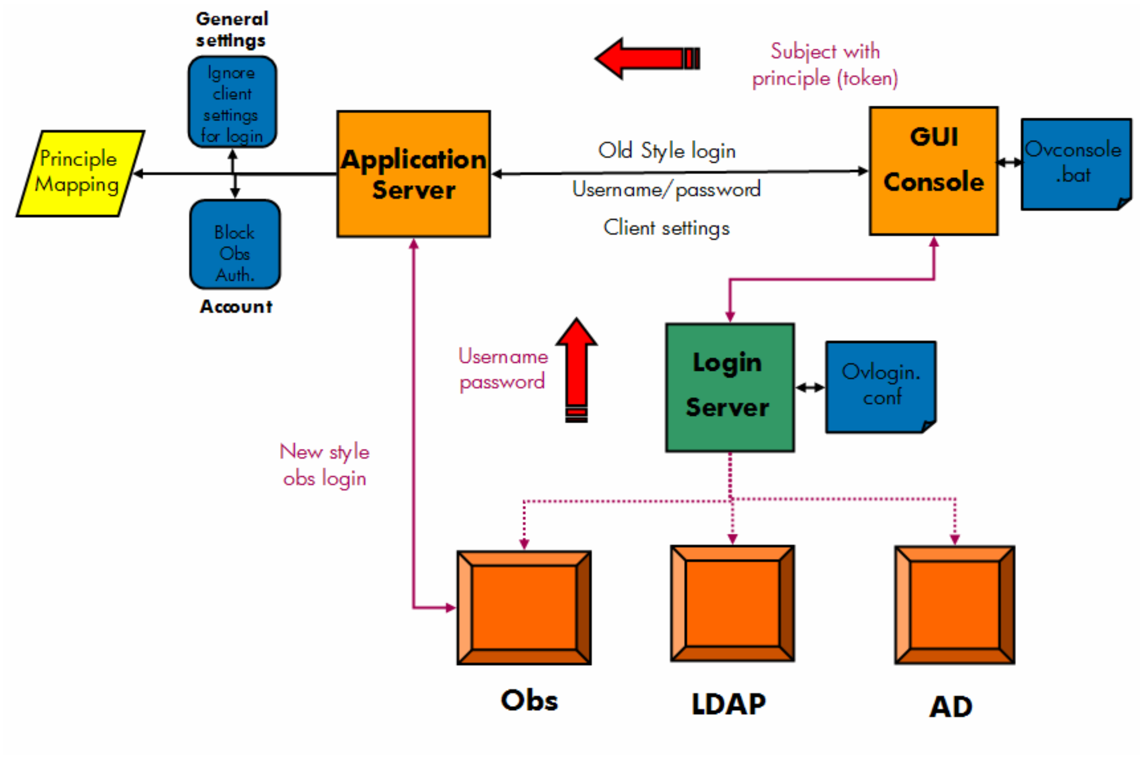
Figure 1-6 shows Service Desk login authentication using JAAS, Active Directory or LDAP, and the associated components involved.

Figure 1-6 **Service Desk Login**



The above process is followed only if the Ignore Client Settings for Login option (in the General Settings form) is selected. If this option is *not* checked, the Service Desk client logs directly onto the Object Server using the credentials specified in the Client Settings editor, as illustrated in Figure 1-7.

Figure 1-7 Using Client Settings for Logon



For more information about configuring login authentication, go to the HP OpenView Service Desk online help, click:

Information for Administrators → Users & Security → User Authentication

Object Server

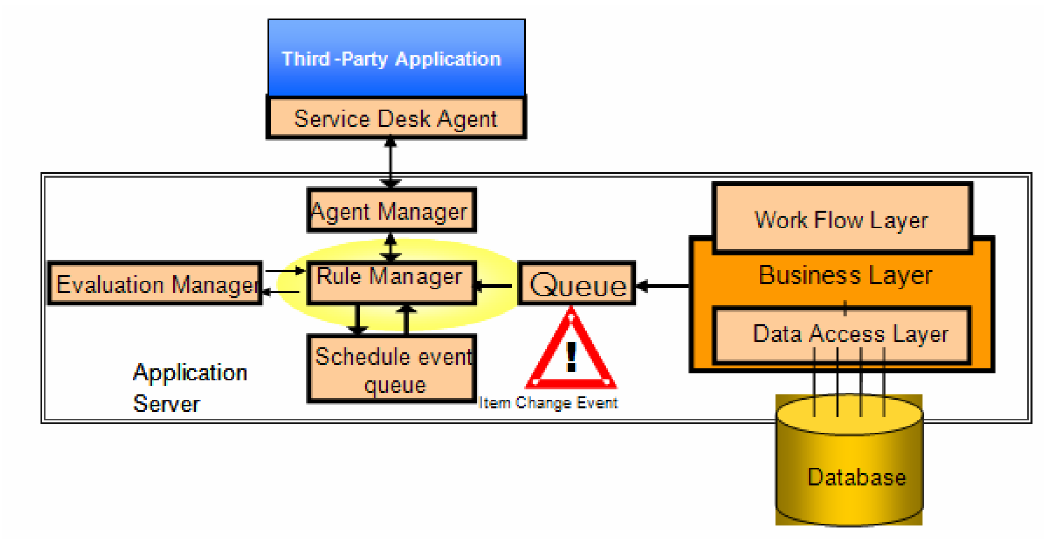
This section describes the Object Server.

Rules

Rules have debugging facilities. To debug rules, select the Debug option in the Rules Wizard when naming the rule. Debugging information is saved to the `system` and `rule` log files in the `$intall_dir\data\log` directory.

Rules are processed as illustrated in Figure 1-8.

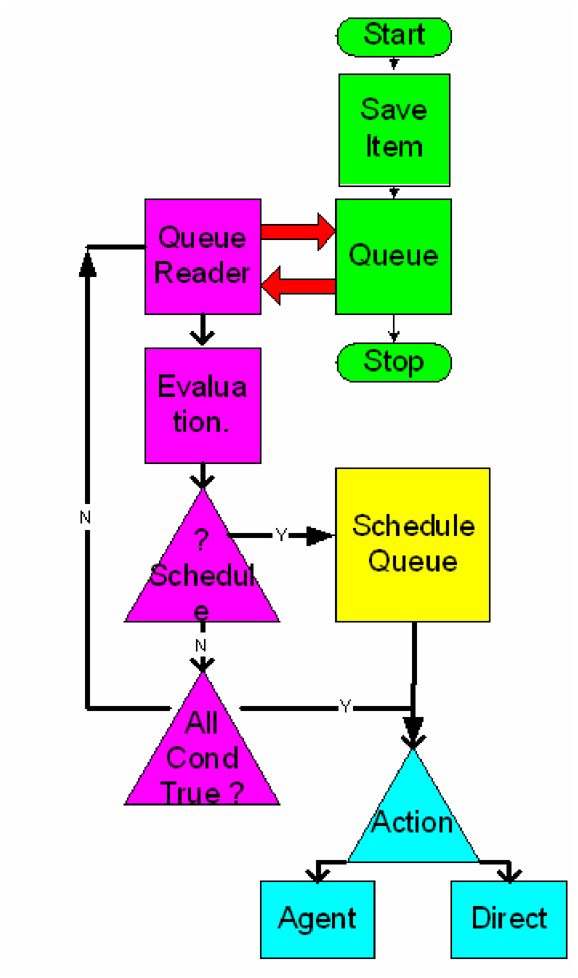
Figure 1-8 How Rules are Processed



The size of the rule queue is displayed in the Server Monitor on the Queues tab, Event queue.

Rules can be scheduled for later evaluation and execution, as illustrated in Figure 1-9.

Figure 1-9 **How Rules are Queued**



Scheduled tasks are stored in the database and in memory. Every management server has its own task queue.

The management servers handle scheduled tasks simultaneously. Each server works independently from the other. When starting a management server, all scheduled tasks for that server are retrieved from the database.

Rules waiting to be scheduled are placed in the Task Manager queue (see the Queues tab in Server Monitor), which interfaces with the Timer queue. Rules waiting for execution are placed in the Scheduled task queue.

Scheduled tasks can be managed using:

OV Configuration workspace → **Scheduled Tasks** → **Scheduled Rule Task**

Right-click a task to reschedule it. When you reschedule a task, another server is assigned to handle the evaluation and execution of that task. The time that a scheduled task is started is determined by the Scheduled Rule.

When another server reschedules a scheduled execution, the time is not changed, only the hosting server that executes the condition or action at the scheduled time.

If a management server goes down, tasks that are scheduled for that server are rescheduled on another server.

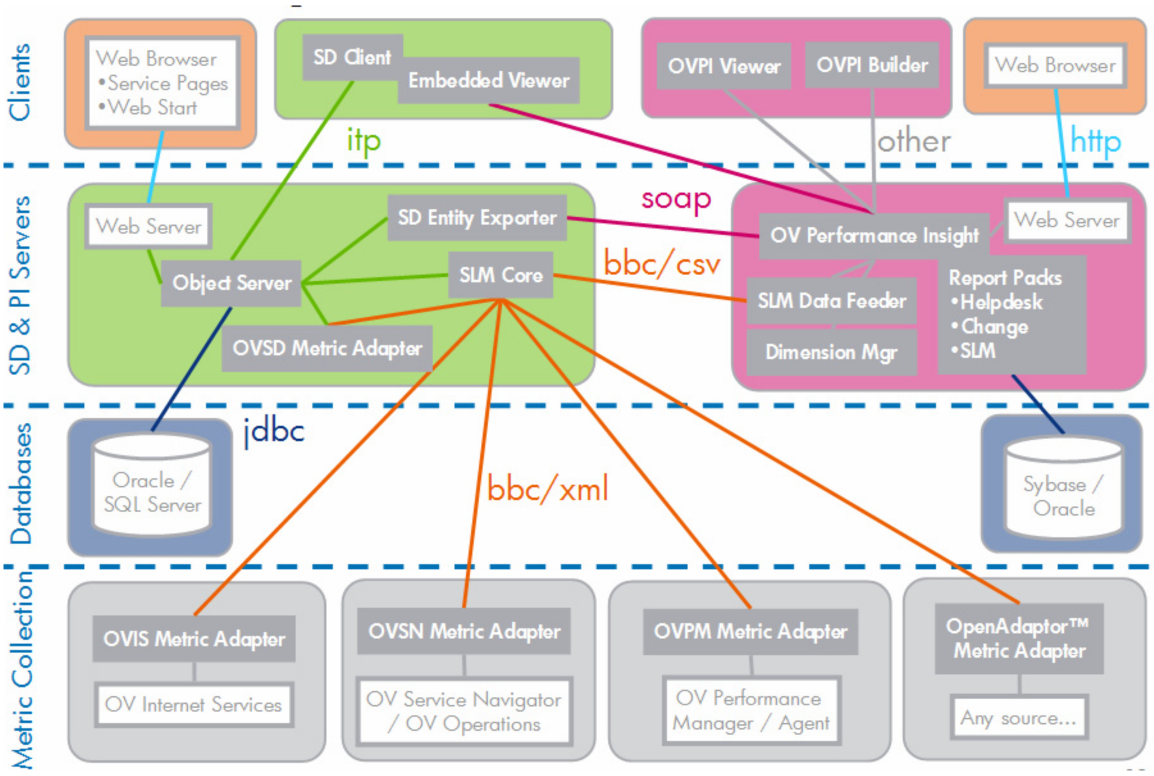
Task rescheduling is handled as follows:

- All scheduled tasks are read from the database.
- Every task is checked to see if the server executing the task is active.
- If the server is not active, the task is rescheduled on another server by the server that is performing rescheduling.

SLM Server and Metric Adapters

Figure 1-10 illustrates the main data flow for the SLM module.

Figure 1-10 SLM Data Flow



OV Performance Insight:

- Is built on a data warehouse.
- Embeds SLM report pack with predefined out-of-the-box reports.
- Offers a graphical builder to customize reports.

Object Server / HP OpenView Console:

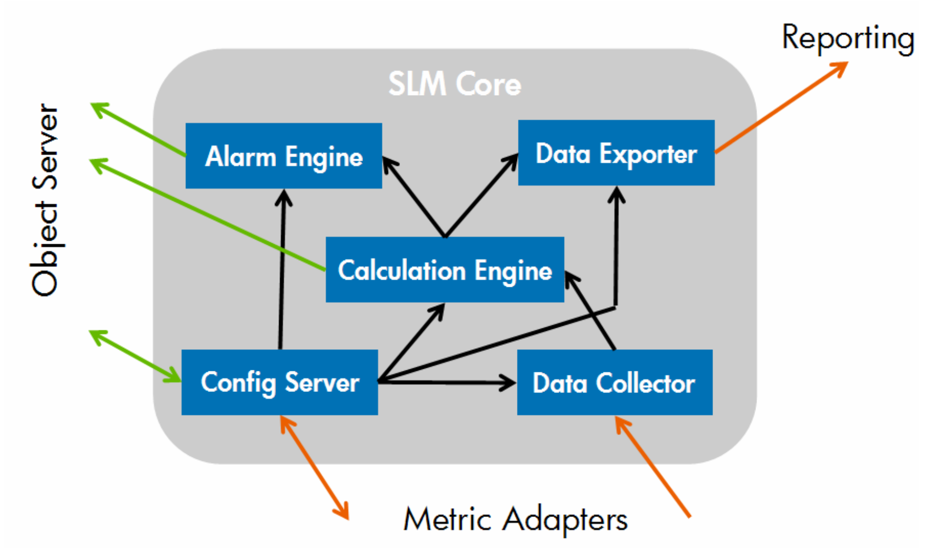
- Acts as an application server and a generic UI for the SLM calculation engine.

SLM Core:

- Collects performance data from metric adapters and calculates compliance and availability statuses.
- Exports performance data to OVPI.

Figure 1-11 illustrates the architecture of the SLM Core.

Figure 1-11 **SLM Core**

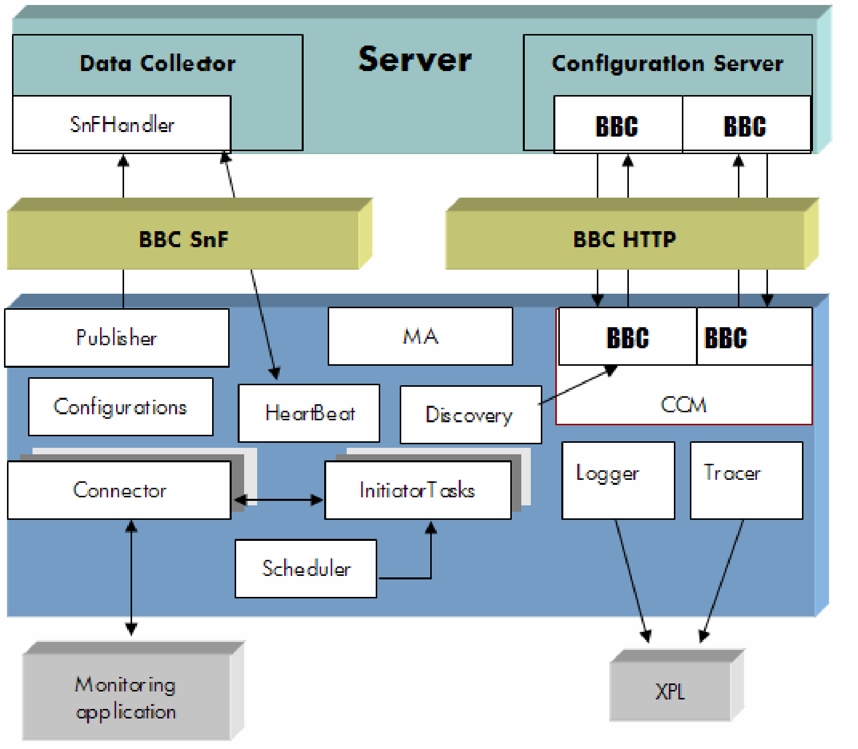


Metric Adapters:

- OVSD – Service Desk adapter.
- OVSN – Service Navigator adapter.
- OVPM – Performance Manager adapter.
- OVIS – Internet Services adapter.
- OpenAdaptor™ based on OpenAdaptor™ technology to address potentially any source.

Figure 1-12 illustrates the architecture of a Metric Adapter (MA, indicated in blue) and communication with the SLM server (`ovsdslm` process, denoted by "Server").

Figure 1-12 **Metric Adapter Architecture**



The Metric Adapter collects the data from the monitoring application, and sends it to the SLM server (Data Collector sub-module).

Impacted Services

In Service Desk 5.0 and higher, you have the option to retrieve impacted services related to an incident. To do this, choose from the SD Incident form:

Action → Relate Impacted Services to Incident

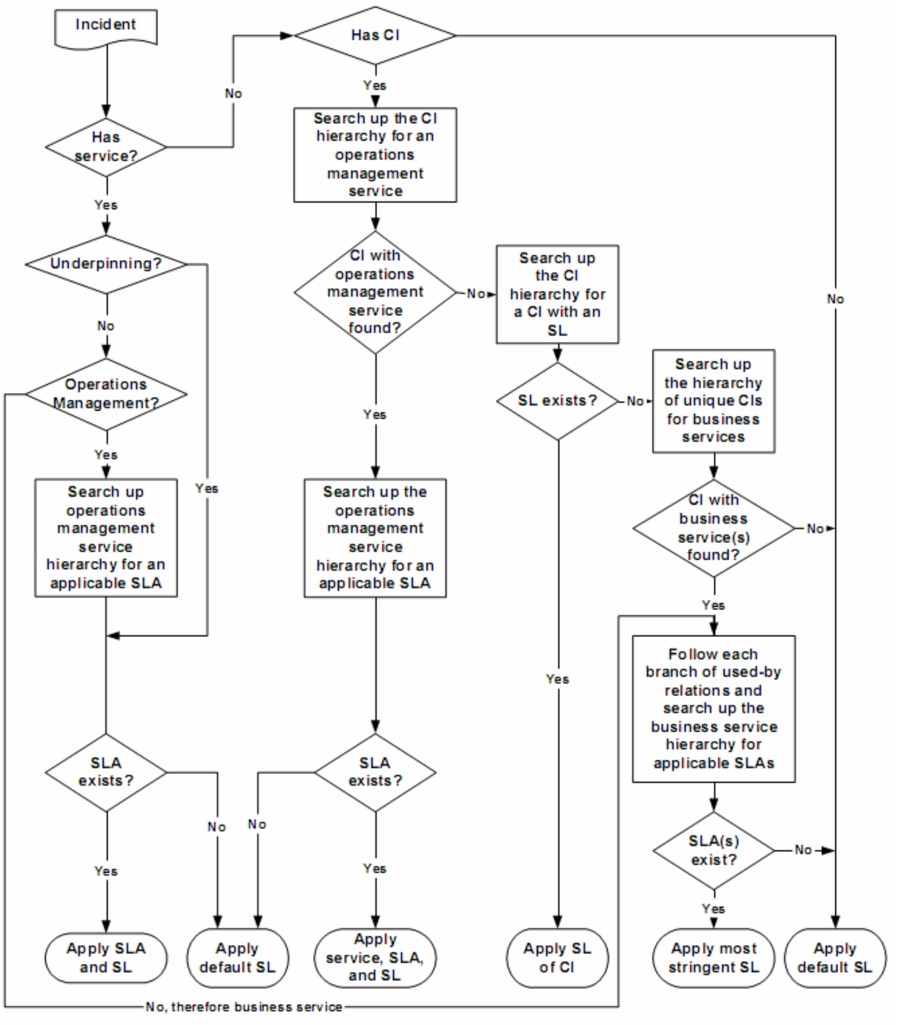
To view impacted services, choose:

Action → View Services Related to Incident

At least one service must be related to the incident as `Initially Identified`.

Figure 1-13 shows the process flow used to apply an Initially Identified Service to an Incident. The process flow illustrates how Service Desk finds impacted services by searching up through the service hierarchy, following each branch of Used By relations, beginning with Initially Identified services. It filters the impacted services using the settings specified in the Impacted Services Settings dialog (System Settings workspace).

Figure 1-13 **Initially Identified Impacted Services**


Incident
Has service?
No
Has CI
Yes
Search up the CI hierarchy for an operations management service
Yes
Underpinning?
No
Operations Management?
Yes
Search up operations management service hierarchy for an applicable SLA
Yes
CI with operations management service found?
No
Search up the CI hierarchy for a CI with an SL
SL exists?
No
Search up the hierarchy of unique CIs for business services
CI with business service(s) found?
No
Yes
Follow each branch of used-by relations and search up the business service hierarchy for applicable SLAs
SLA(s) exist?
No
Yes
Search up the operations management service hierarchy for an applicable SLA
SLA exists?
No
Yes
SLA exists?
No
Yes
Apply SLA and SL
Apply default SL
Apply service, SLA, and SL
Apply SL of CI
Apply most stringent SL
Apply default SL
No, therefore business service

Key: SL = Service Level; SLA = Service Level Agreement; CI = Configuration Item

For more information on the use of different types of services, refer to the the *HP OpenView Service Desk Concept's Guide*.

Helpdesk Deadline Calculation and SLA retrieval

Deadlines for solving reported problems and fulfillment of service requests are set by the helpdesk in a Service Call or Incident.

You can configure Service Desk to calculate deadlines automatically when you enter a service call or incident using SLA and support hour settings.

The basic deadline calculation can be used for support organizations that use fixed support hours. Deadline calculation in Service Desk requires the following:

- Start date
- Time zone
- Support hours
- Duration

The examples in this section use service calls. However, the same rules can be applied for incidents.

Time Zone

Support hours are stored without time zone information. You need to specify the time zone of the support organization.

To set the time zone of the support organization, from the `OV Configuration` workspace group:

1. Select **System Settings**.
2. Double-click the **Regional Settings** icon.
3. Click the **Time Zones** tab.
4. Choose a time zone from the **Primary Time Zone** drop-down list.

Support Hours

To set support hours:

1. Click the **Schedule Element** shortcut you created in your new workspace.
2. Create a new Schedule Element that reflects your normal support hours.

   For example, Monday to Friday, 09:00 to 17:00. Do not set a time zone. You can also create several Schedule Elements that together reflect your normal support hours.
3. Create another Schedule Element that reflects your holidays. Do not set a time zone. You can also create several Schedule Elements that together reflect your holidays.
4. Click the **Schedule** shortcut.
5. Create a new schedule.
   a. In the **Included Schedules** tab, relate all Schedule Elements that reflect your normal support hours.
   b. In the **Excluded Schedules** tab, relate all Schedule Elements that reflect your holidays.
6. Enter the new schedule.
   a. From the **OV Configuration** workspace group, select

      System Settings and double-click the **General Settings** icon.
   b. Select the **Application** tab and select a **Default Support Hours Schedule** from the drop-down list.

TIP To see how support hours affect deadline calculations, set support hours to Saturday 18:00 to 00:00.

Duration

The duration used in the deadline calculation is derived from the Priority set for the Service Call or Incident. The duration must be mapped to each available Priority. For example, Top priority = 1 hour.

To map a duration to a priority, edit the priority codes From the OV Configuration workspace group as follows:

- For service calls, select:

  Data → Codes → Service Call → Service Call Priority Duration Setting

- For incidents, select:

  Data → Codes → Service Call → Incident Priority Duration Setting

Calculation

To calculate the deadline:

1. Open a new Service Call.
2. Set the Priority.

Service Desk calculates the deadline using your previous settings. The start date is the date the object was created in the service call (see the Registration date in the History tab). This is the same for incidents.

If you logged on with an account that has a different time zone than the Primary Time Zone, the deadline date is shown in your account time zone.

Helpdesk Deadline Calculation with SLM

An impact-priority mapping is related to a service level using the Impact Priority Setting tab on the Service Level form.

You can configure SLM to calculate important deadlines for customers and services using the support hours and impact-priority mappings. Support hours and impact-priorities are retrieved dynamically by the Service Level Agreement (SLA) that is entered in the service call or incident.

Impact-priority mapping allows Service Desk to determine the priority of a service call or incident and calculate the maximum duration for the deadline using the priority duration settings.

The impact value is related to the impact of the service call or incident at the customer site.

The priority given for the service call or incident depends on the importance of the service that is disrupted and the service level expected by the customer. This information is stored in an SLA. Impact-priority mappings are part of this information.

You cannot set an SLA in a service call or incident directly. An SLA is entered automatically after you have entered a value.

Summary

A service level comprises the following:

- Impact-priority mappings
- Priority-duration settings.
- Support hours

These variables are retrieved from the SLA.

If no SLA is entered in the service call or incident, or there is no default service level available, Service Desk uses the default impact-priority mappings. The following sections show how services and SLAs are set and included in new service calls and incidents.

Service and SLA Retrieval Settings for Service Call

To determine how services and SLAs are retrieved automatically for service calls:

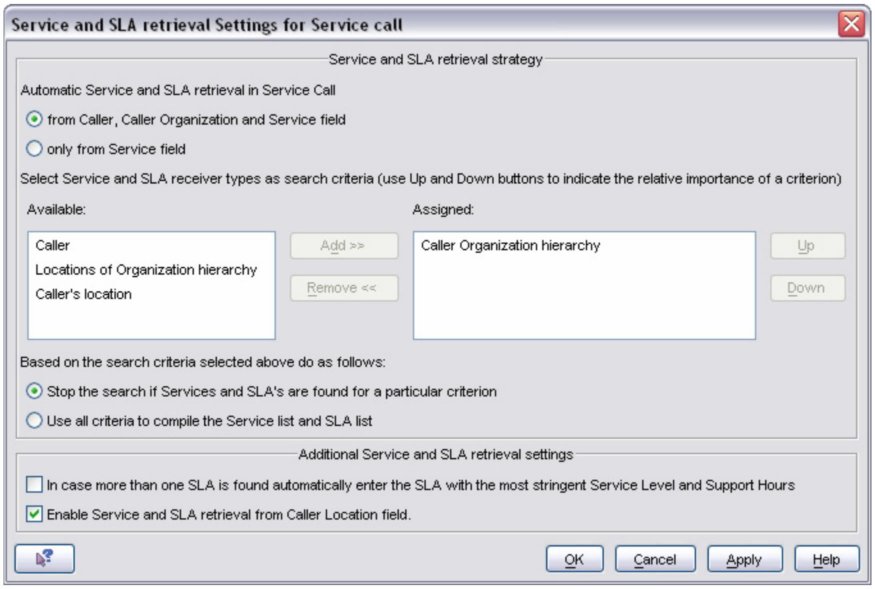
1. From the **OV Configuration** workspace group:
    a. Select **System Settings**.
    b. Double-click the **Service and SLA Retrieval Settings for Service Call** icon.

       The Service and SLA retrieval Settings for Service Call dialog is displayed.
2. Enable the following settings (these are also shown in Figure 1-14 on page 39):
    - **From Caller, Caller Organization and Service field**
    - **Stop the search if Services and SLAs are found for a particular criterion**
    - **Enable Service and SLA retrieval from Caller Location field**

3. For the `Select Service and SLA Types as Search Criteria` option, make sure the following criterion are in the Available list:

   - **Caller**
   - **Locations of organization hierarchy**
   - **Caller's location**

Figure 1-14 Service and SLA retrieval Settings for Service call



Service and SLA retrieval Settings for Service call

Service and SLA retrieval strategy

Automatic Service and SLA retrieval in Service Call

[✓ from Caller, Caller Organization and Service field]

only from Service field

Select Service and SLA receiver types as search criteria (use Up and Down buttons to indicate the relative importance of a criterion)

Available:
Caller
Locations of Organization hierarchy
Caller's location

Assigned:
Caller Organization hierarchy

Based on the search criteria selected above do as follows:

[✓ Stop the search if Services and SLA's are found for a particular criterion]

Use all criteria to compile the Service list and SLA list

Additional Service and SLA retrieval settings

☐ In case more than one SLA is found automatically enter the SLA with the most stringent Service Level and Support Hours

☑ Enable Service and SLA retrieval from Caller Location field.

To see how Service Desk uses these settings in a service call:

1. Create an organization named **`TestOrg`** with a different time zone than the primary time zone.
2. Create a service named **`TestServiceEmail`** and relate `TestOrg` as the receiving organization.
3. Create an SLA named **`TestSLAEmail`**:

   - Relate `TestServiceEmail` as the **Service**.
   - Relate TestOrg as the receiving **Organization**.
   - Set **Applied Timezone** to **Provider Timezone**.

4. Create a new service call:
   - Select **TestOrg** as the caller organization in **Caller Info**.

The Service and SLA of a service call are automatically retrieved as TestServiceEmail and TestSLAEmail. The SLA is entered in the service call because the caller organization is related as a receiver to both the service and the SLA. Service Desk can then calculate the deadline.

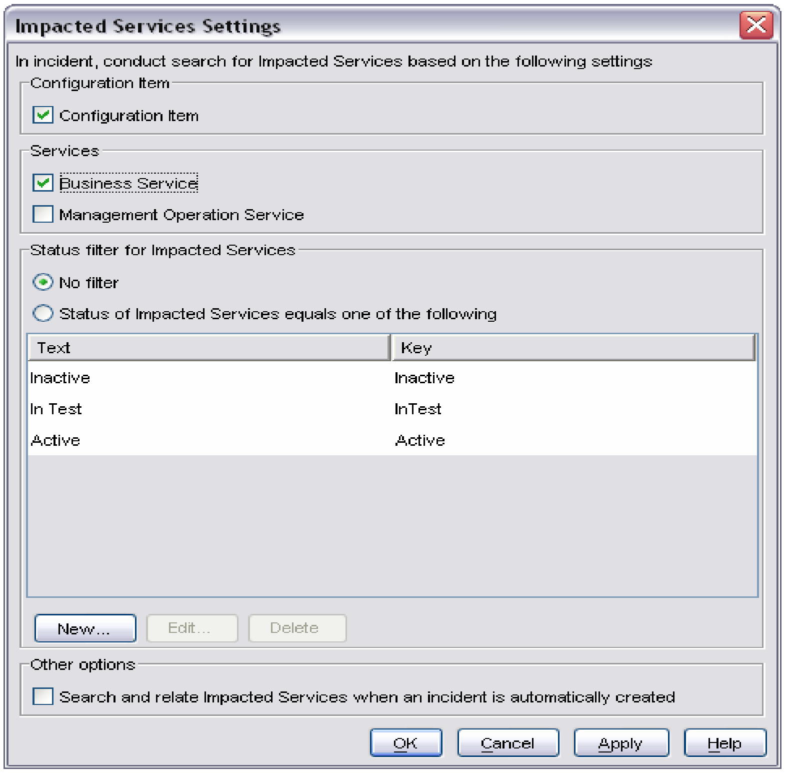
Service and SLA Retrieval Settings for Incident

To set how impacted services are retrieved for incidents:

1. From the OV Configuration workspace group:
   a. Select **System Settings**
   b. Double-click **Impacted Service Settings**.

      The Impacted Service Settings dialog is displayed.
2. Enable the following settings as shown in Figure 1-15:
   - **Configuration Item**
   - **Business Service**
   - **No filter**

Figure 1-15 **Impacted Services Settings**



In incident, conduct search for Impacted Services based on the following settings

Configuration Item

[✓] Configuration Item

Services

[✓] Business Service

[ ] Management Operation Service

Status filter for Impacted Services

[✓ No filter]

Status of Impacted Services equals one of the following

Text	Key
Inactive	Inactive
In Test	InTest
Active	Active

Other options

[ ] Search and relate Impacted Services when an incident is automatically created

To see how Service Desk uses these settings for an incident:

1. Open the **TestServiceEmail** service and select the **Used CIs** tab.
2. Click **New** and name the CI **`TestCI`**.
3. Select the **Details** tab, and select **Unique**.
4. Save and close the CI and the service to which it is now related.
5. Create a new incident:
   - Select **TestCI** as the Configuration Item.
   - Save the incident (but do not close) to see Service Desk calculate the deadline.

The impacted service of the incident is set to TestServiceEmail. The SLA of the service is TestSLAEmail.

Setting an Impact-Priority Mapping in the SLA

To set an impact-priority mapping in the SLA:

1. From the Service Designer workspace group, click the **Service Level** icon and create a new service level named **`SLTest`**.
2. From the Impact Priority Setting tab, create a new **Impact Priority Configuration**.
3. Save the service level with the new impact-priority configuration.
4. Open the TestServiceEmail service and set the Service Level to **SLTest**. Save and close.
5. Create a new service call and set caller organization to **TestOrg**.

The service of the service call is set to TestServiceEmail. The SLA is set to TestSLAEmail, the Service Level is set to SLTest.

When you set an impact, the priority is set according to the impact-priority mappings in the service level.

Service Desk then calculates the deadline.

Setting Support Hours in the SLA

The more support hours in an SLA results in earlier deadlines, for example, every day from 07:00 to 19:00 instead of every weekday from 09:00 to 17:00. Additional support hours related to the SLA indicate an important customer or service or both.

To set support hours in the SLA:

1. Create support hours for TestOrg. See “Support Hours” on page 36.
2. Open TestSLAEmail:
    - In the Schedules tab, select **Schedules**, click **New**, and set the Schedule to the support hours schedule for TestOrg.
    - Set the Type to **Support Hours**.
    - Save and close.
3. Create a new service call and select **TestOrg** as the caller organization.

The service of the service call is set to TestServiceEmail and the SLA is set to TestSLAEmail.

Service Desk calculates the deadline using the support hours that are related to TestSLAEmail.

TIP To see how support hours affect the deadline calculation, set support hours to Sunday 12:00 to 18:00.

Setting Support Hours per Impact

For Service Desk 5.0 and higher, you can set the support hours by impact.

Not every disruption of service must be solved within the same support hours. For example, the Email service has support hours set to weekdays 09:00 to 17:00 for normal issues (for example, password forgotten for Email account). Under normal circumstances, the Email service is available 24 hours a day.

To create a 24x7 schedule:

1. Open TestSLAEmail:
   a. In the **Assignments & Support Hours** tab, **Support Hours per impact**, click **New** and set **Support Hours** to the 24x7 schedule.
   b. Set the **Impact** to **Top**.
   c. Save and close.
2. Create a new service call and select **TestOrg** as the caller organization.

The Service of the service call is set to TestServiceEmail and the SLA is set to TestSLAEmail.

Service Desk calculates the deadline using the support hours that are related to TestServiceEmail. When you set the impact to Top, Service Desk uses the 24x7 schedule to calculate the deadline.

Support Hours in Support Organization’s Time

To maintain low maintenance costs for a service where the support hours should be in the time zone of the support organization, you need to set the time zone in the SLA to the provider time zone and related provider organization.

To set the time zone in the SLA to the provider time zone and organization:

1. In the SLA:
   a. Set the Applied Timezone option to **Provider Timezone**.
   b. Relate a provider organization (or support organization) to the SLA.
2. Create a new organization named **`TestOrgSupport`** with a time zone that is different than the primary time zone (for example, **`UTC+2`**).
3. Open **TestServiceEmail** and relate it to **TestOrgSupport** as the provider organization.
4. Create a new service call and select **TestOrg** as the caller organization.

The service call's Service is set to TestServiceEmail, and the SLA is set to TestSLAEmail.

NOTE If the applied time zone is set to the Provider time zone, but there is no provider organization related to the SLA, Service Desk uses the primary time zone as the provider's time zone.

Service Desk calculates the deadline using the Support Hours that are related to TestServiceEmail, and uses the time zone of TestOrgSupport. In the following example, we see how the deadline is calculated.

The following conditions apply:

- Console display time zone is set to UTC.
- TestOrgSupport time zone is set to UTC+2.
- Support hours are 09:00 to 17:00 next day.
- Duration is 4 hours.

The support organization will solve the problem before 09:00 + 4 hours = 13:00 on the next day.

Service Desk calculates the deadline as 13:00 - 2 hours = 11:00 on the next day.

Support Hours Relative to the Customer in the SLA

To set the time zone in the SLA to the customer’s time zone:

1. Open **TestSLAEmail** and set the Applied Timezone to **Consumer Timezone**.
2. Create a new service call and select **TestOrg** as the caller organization.

The service call's Service is set to TestServiceEmail, and the SLA is set to TestSLAEmail.

Service Desk calculates the deadline using the support hours that are related to TestSLAEmail and the time zone of TestOrg.

The following example, we see how deadline is calculated.

The following conditions apply:

- The console display time zone is set to UTC.
- TestOrg time zone is set to UTC+4.
- Support hours are 09:00 to 17:00 next day.
- Duration is 4 hours.

The customer expects the problem to be solved before 09:00 + 4 (duration) hours = 13:00 on the next day.

The deadline is calculated as 13:00 - 4 (time zone) hours = 09:00 on the next day.

2 Performance Tuning

This chapter describes how to optimize Service Desk performance. Some of the topics in this chapter can be applied generally, and some are relevant to specific situations only. It is therefore useful to become familiar with Service Desk performance tuning as a whole.

Query Restrictions

Views and queries created with Advanced Find can return many hundreds of records, depending on the criteria and size of the database. This volume of data creates higher network traffic and possibly server overload. The following subsections describe how to control the number of records in a view, or the number of records displayed from a search.

Symptoms of Retrieving Many Records

Simultaneously retrieving many records include heavy network loads and high response times from the Service Desk client.

View Filters

A typical Service Desk user needs to see about 40 records in each view. Try to make sure that filters in views select only the records that the user actually needs to work on. The same applies to columns shown in a view: show only columns in a view that the user needs to see. Details can always be shown by opening the record.

Also consider creating more views for different tasks or creating views that can be navigated, similar to the explorer view.

Setting a Default Query Restriction

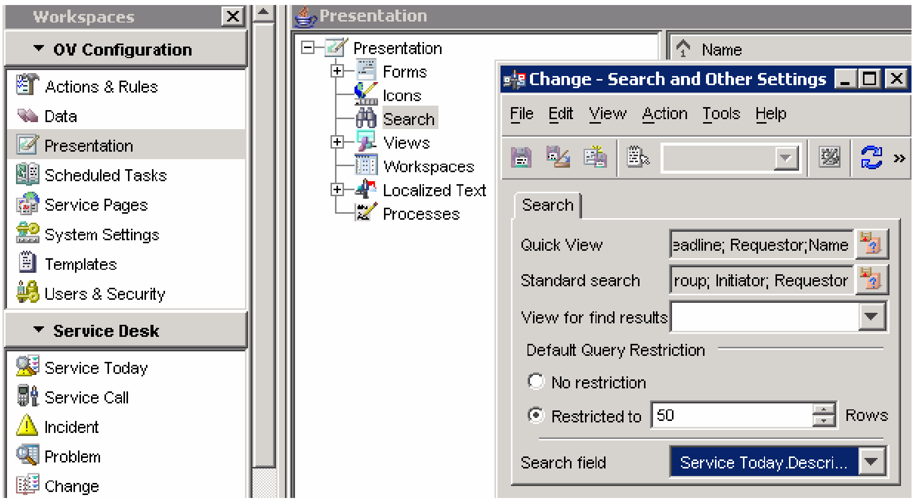
A user who customizes a view or uses Advanced Find can still retrieve an inappropriately large number of records by setting non-selective filters or search criteria. This will unnecessarily increase the load on the application server and database.

To avoid this increase in load, set a default query restriction as follows:

1. Open the **Presentation** workspace and select **Search**.
2. Select the object for which you want to restrict the amount of returned records.

3. Open the **Search and Other Settings** dialog and enter an appropriate value in the Default Query Restriction radio box, preferably in the range 20 to 80 (see Figure 2-1). The number you enter specifies the maximum number of records shown in a view or in an Advanced Find for the selected object.

Figure 2-1 **Optimizing Views and Queries**



Quick View: eadline; Requestor;Name
Standard search: roup; Initiator; Requestor
View for find results:
Default Query Restriction
○ No restriction
● Restricted to 50 Rows
Search field: Service Today.Descri...

Setting Role Query Restrictions

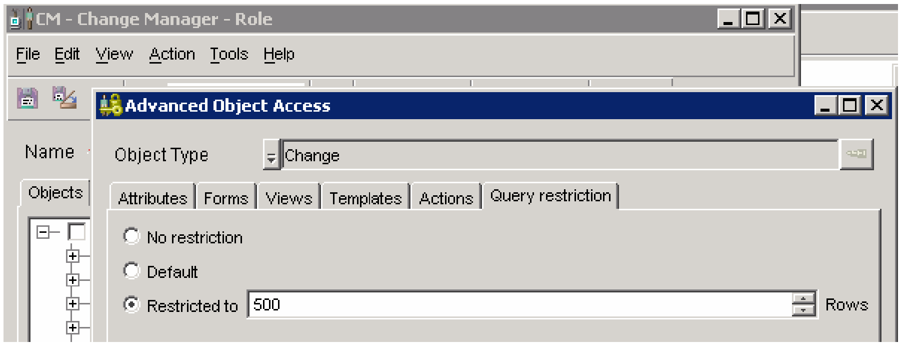
The default query restriction described in “Setting a Default Query Restriction” on page 48 restricts the potential for disruption to the Service Desk environment. However, this measure may seriously impair the work of some Service Desk users. For example, a helpdesk manager may require reports, in chart view, that show statistics aggregated from a large number of records.The Role Query restriction can be used to accommodate these users. The query restriction specified for a Role overrides the default query setting.

To set query restrictions for selected user roles, complete the following steps:

1. From the **Users & Security** workspace, select **Access** → **Roles**.
2. Select a role, and edit this to open the **Role** dialog.

3. Select an object from the **Objects** tab and click **Advanced...** to display the **Advanced Object Access** dialog.
4. Select the **Query restriction** tab
5. Select the **Restricted to** option and enter the required number of records in the associated field. See figure Figure 2-2.

Figure 2-2 **Role Query Restriction Settings**



Object Type: Change

No restriction [✓ Restricted to] Default

Restricted to 500 Rows

Optimizing Forms

You can also optimize forms. For example, you can show only fields that are needed or create different forms for different users.

See the HP OpenView Service Desk online help for more information.

Database and UI Rules

Database and UI rules are executed in sequential order, which can result in a higher load on the Service Desk server.

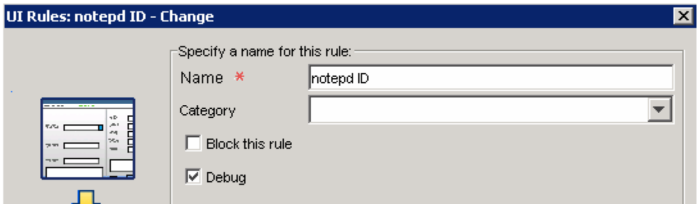
Symptoms

High CPU load on the Service Desk server, high response times from clients, average load on the database.

Solution

Minimize, where possible, the amount of database and UI rules running on the server:

- Block rules that are not used. Select the **Block this rule** option in the Rules Wizard.


UI Rules: notepd ID - Change

Specify a name for this rule:
Name * notepd ID
Category
[ ] Block this rule
[✓] Debug

- To avoid a multiple of related scheduled tasks (for example multiple notification rules that trigger at 2 hours, 1 hour, 30 min., and 5 min.) before a deadline, allow scheduled rules to trigger each other. For example, when the first “2 hour rule” evaluates to true, send the notification and schedule the next trigger for the “1 hour rule”.

  However, prevent rules from triggering each other in the same action. UI rules are also executed on the server. A database rule can trigger the execution of a UI rule.

- Move time-consuming tasks to another server. For example, use the Service Desk agent to have database rules execute programs on another server.

- Select the **Debug** option in the Rules Wizard to examine the behavior of rules. See the system log files for statements logged by rules.

- Increase the number of Service Desk server instances (see page 66).

- Remember that scheduled rules (tasks) are actually scheduled *conditions*. At the scheduled time, all conditions are evaluated, including conditions that are not scheduled for this time. When the composite condition evaluates to true, the actions are executed. To avoid a large build-up of scheduled tasks, consider adding an extra condition on the same field of the scheduled condition, as shown in Example 2-1.

Example 2-1 Avoiding Queue Build-up by Adding Extra Conditions to a Rule

The initial rule is as follows:

When the Service Call is created or modified

If field1 = A AND field2 = B AND field3 = C after 2 weeks > execute action 1, action 2 and so on.

Saving Service Call No. 123 with field1 = A, field2 = B and field3 = D will result in a scheduled task, which is queued for 2 weeks, but it will then evaluate to false.

A more efficient rule definition would be follows:

When the Service Call is created or modified

If field1 = A AND field2 = B AND field3 = C AND field3 = C after 2 weeks > execute action 1, action 2 and so on.

Folders

Security Folder permissions can result in complex queries to the database.

Symptoms

High CPU load on the database, and high response times from Service Desk clients.

Solution

Reduce the number of folders to which users have access by using other authorization options. For example, you can replace folder permissions by restricting access to specific templates, forms and views.

Roles

In Service Desk you can include roles within roles. This makes it possible to create roles that can be re-used for different users. However, roles that include several levels of roles results in a higher load for the Service Desk server. Changing roles that contain complex role hierarchies causes higher network traffic between the client and server.

Symptoms

Startup of Service Desk clients becomes slower.

Solution

Reduce the number of included roles to one level deep. Try to include all information into one role where possible.

Service Call and Incident Queries

Service calls and incidents perform complex queries on services and SLAs to retrieve contextual information. For example, after you enter a Caller Location in a service call, or a Configuration Item or Service in an incident, Service Desk queries the database to retrieve the applicable services and SLAs. This can result in slow response times.

Symptoms

Service Desk response times become longer.

Solution

Restrict data retrieval to data that is relevant to your organization.

For incidents, you specify the relevant settings in the Impacted Services Settings form:

System Settings → **Impacted Services Settings**.

For service calls, you specify this in the Service and SLA retrieval Settings for Service call form:

System Settings → **Service and SLA retrieval Settings for Service call**.

Both options are described below.

You can also remove the option of searching for a caller's organization, as follows:

- In the **System Settings** → **General Settings**, **Application** tab, deselect the **Search Caller person in Organization tree** option.

Queries from Service Calls

You can minimize the impact of queries from service calls by specifying the criteria to determine the SLA and service information searched for and retrieved when a user enters information on a service call form.

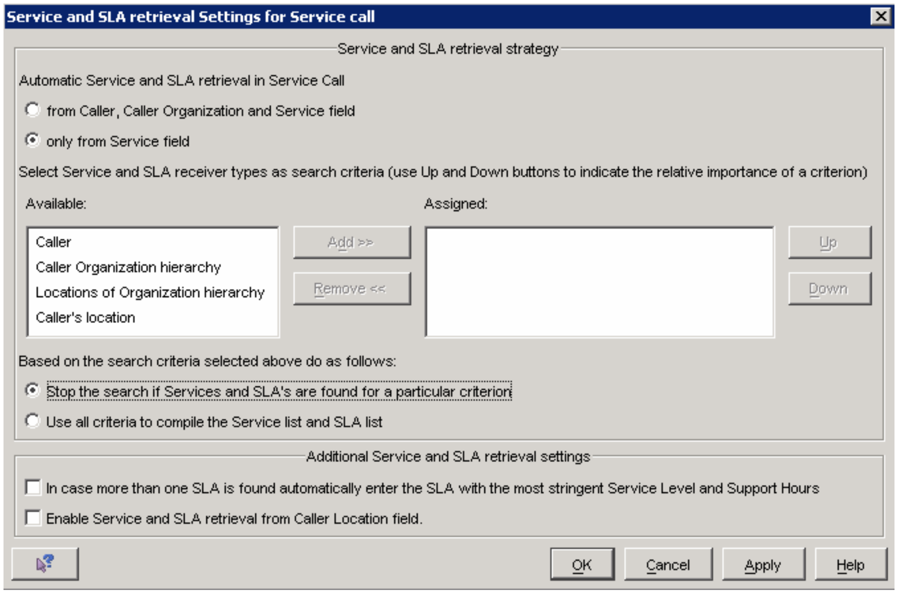
To specify the criteria used to query an SLA, complete the following steps and refer to Figure 2-3:

- Select **System Settings** → **Service and SLA retrieval Settings for Service call**.

The Service and SLA retrieval Settings form is displayed. Do the following on this form:

- Select **only from Service field**.
- Select **Stop the search if Services and SLAs are found for a particular criterion**.
- Move **Caller Organization hierarchy** to the Available column.
- Deselect **Enable Service and SLA retrieval from Caller Location field**.

Figure 2-3 **Retrieval Settings for Service Calls**



Service and SLA retrieval Settings for Service call

Service and SLA retrieval strategy

Automatic Service and SLA retrieval in Service Call

( ) from Caller, Caller Organization and Service field
[✓ only from Service field]

Select Service and SLA receiver types as search criteria (use Up and Down buttons to indicate the relative importance of a criterion)

Available:	Assigned:
Caller	
Caller Organization hierarchy	
Locations of Organization hierarchy	
Caller's location	

Based on the search criteria selected above do as follows:

[✓ Stop the search if Services and SLA's are found for a particular criterion]
( ) Use all criteria to compile the Service list and SLA list

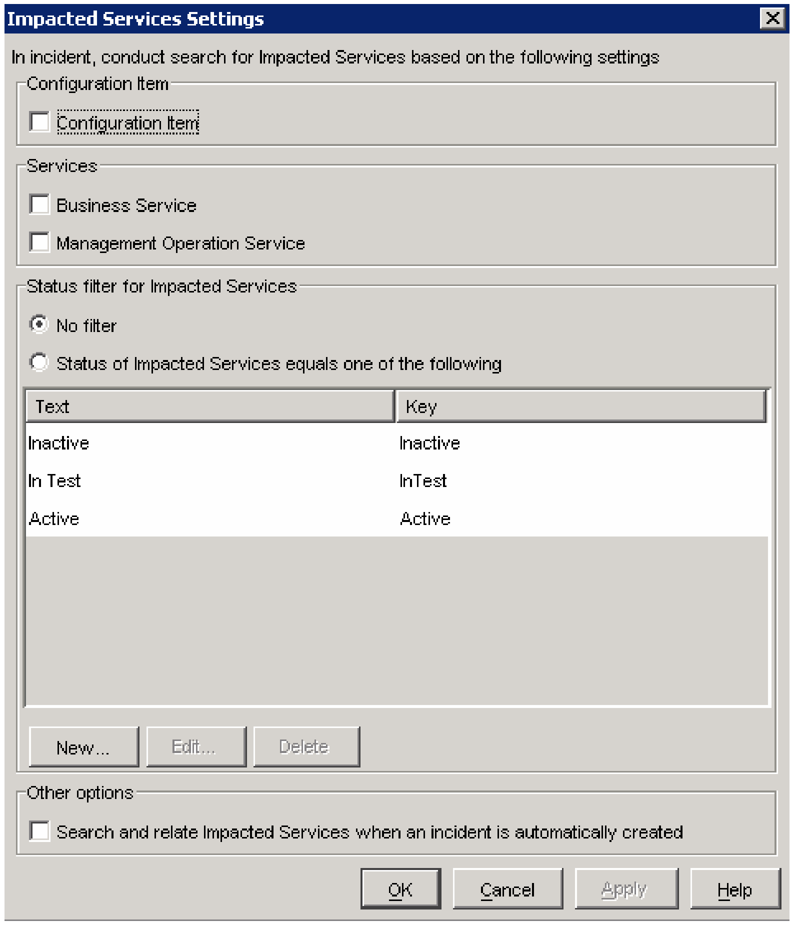
Additional Service and SLA retrieval settings

☐ In case more than one SLA is found automatically enter the SLA with the most stringent Service Level and Support Hours
☐ Enable Service and SLA retrieval from Caller Location field.

Queries from Incidents

You can also minimize the impact of queries from incidents, as shown in Figure 2-4. For information on this form, refer to the HP OpenView Service Desk online help (Define Impacted Services Search Settings).

Figure 2-4 **Retrieval Settings for Incidents**



Impacted Services Settings

In incident, conduct search for Impacted Services based on the following settings

Configuration Item

- [ ] Configuration Item

Services

- [ ] Business Service
- [ ] Management Operation Service

Status filter for Impacted Services

[✓ No filter]
Status of Impacted Services equals one of the following

Text	Key
Inactive	Inactive
In Test	InTest
Active	Active

Other options

- [ ] Search and relate Impacted Services when an incident is automatically created

NOTE For more information on the Service Call and Incident Query dialogs, see the HP OpenView Service Desk online help.

Java Heap Size

A prerequisite for Service Desk is that you suitably configure the Java heap size of the Java Virtual Machine (JVM).

Symptoms

- Object Server (`ovobs`) produces out of memory errors.
- Server regularly shows high CPU load (because of garbage collection). During this period of high CPU load, the Service Desk clients experience low response times from the server and the server memory usage does not decrease over time. View this in the Performance tab:

  HP OpenView Server Monitor → Performance

Solutions

The server usage (for example, steady load, peak load, email processing, and data exchange tasks) determines the configuration needed for the Java heap size and garbage collection. General guidelines are documented in the *HP OpenView Service Desk Installation Guide*.

Use HP JTune to configure the Java heap sizes. HP JTune is available at the following site:

http://www.hp.com/products1/unix/java/java2/hpjtune/index.html

For more information on setting the Java heap size, visit the following site:

http://www.javaperformancetuning.com/tools/hpjtune/index.shtml

To set the heap size for the Object Server (the core Service Desk server process), complete the following steps:

1. Edit the file `%OvDataDir%conf/obs/OvObs.xml` and add the following heap size settings after Dctrl=START:

   ```
   Dctrl=START -XX:MaxNewSize=128M -XX:NewSize=128M -Xms256M
   -Xmx1024M
   ```

NOTE `%OvDataDir%` is your installation data directory.

2. Stop the server processes:

   ```
   ovc -stop
   ```

3. De-register the Object Server:

   ```
   ovcreg -del ovobs
   ```

4. Re-register the Object Server:

   ```
   ovcreg -add%OvDataDir%conf/obs/OvObs.xml
   ```

5. Re-start the Object Server:

   ```
   ovc -start ovobs
   ```

6. Check if re-registering the Object Server is successful:

   ```
   ovc -status ovobs -level 2
   ```

Check the allocated memory display in the HP OpenView Server Monitor, Performance tab. Allocated memory should be between 256000 and 1024000 kilobytes.

Service Desk Server and Database

The Service Desk servers must communicate frequently with each other and with the database.

Symptoms

Low response from servers or databases that have a low CPU load is experienced.

Solution

Put the Service Desk servers and database on a separate network. Make sure that communication from the servers to the database and between servers is not disturbed by other network activity, such as probing, and scanning. If you use an FTP server for attachments, and most users connect using the Service Desk client installed using Web Start, consider placing this FTP server on a separate network. Installed clients or clients started using Java Web Start connect to the FTP server directly to retrieve or store attachments.

Service Desk Database

The Service Desk servers regularly send complex queries to the database. The database must be configured to execute queries efficiently.

Symptoms

Low response from servers with low CPU load, high CPU load on database.

Solution

Oracle and MS SQL Server have multiple tuning options. Some of these options can degrade performance when Service Desk queries are executed (for example, Oracle can use other query optimization schemes, some of which are counter productive). Ask your DBA to optimize the database for Service Desk queries.

To view queries that are sent to the database:

1. While running Service Desk, start the HP OpenView Server Monitor:

   Start → **Programs** → **HP OpenView** → **Server Monitor**

2. From the **Database** tab, click **Start Logging**.
3. Click **View Log** to view the Query Log information.

The Query Log generates the `OvObsQuery.log` file in the `$installdir\data\log` directory.

Reduce Network Latency

The Service Desk client sends multiple requests to the server. Each request takes time to travel over the network.

Symptoms

Low response from client, even though there is low CPU load on clients, servers and the database.

Solution

Reduce the distance between the server and client. Users can then login remotely with, for example, VNC or Remote Desktop Connection or Citrix server.

If the user has a limited set of tasks, use Service Pages, the Web Console (not from Web Start), the email interface, or create your own interface with the Web API.

Stop Unnecessary Server Processes

The Service Desk server is installed, and several unnecessary processes are running.

Symptoms

The Service Desk server is generating a high CPU load for small tasks.

Solution

1. At the command line, use **`ovc -status`** to display the current status of processes on the server.
2. Use **`ovc -stop`** to stop all processes. Then start only the processes you require. For example, to run the Service Desk server without the SLM server, enter the following at the command line:

   ```
   ovc -start ovobs
   ```

   All processes that `ovobs` needs are started automatically.
3. To stop unwanted processes from starting up at system boot-time, use `ovcreg`. For more information, refer to “Control Configuration” on page 142.

The following processes must run on one server only. Running them on two or more servers causes errors:

- `ovcs` (certificate server)
- `ovsdreport` (reporting server)
- `ovsdslm` (SLM server)

The processes that can only function when configured with SLM are:

- `ovsdma`
- `openma`
- `ovsnma`
- `ovpmma`
- `ovisma`

Service Desk Server

The Service Desk server executes multiple tasks. Any of these tasks can lock the system and prevent execution of tasks in parallel.

Symptoms

You expect that a server should be working under a high load but it does not show high CPU load.

Solution

Create more server instances. For more information, see "Increase the Number of Server Instances" on page 66.

To see which process is blocking other processes (or threads), use the HPJmeter tool. For more information about HPjmeter refer to the following URL:

http://www.javaperformancetuning.com/tools/hpjmeter/index.shtml

As described in the HPjmeter documentation, excessive logging can cause delays.

More information on HPJMeter is also available at the following URL:

http://www.hp.com/products1/unix/java/hpjmeter/index.html

Increase the Number of Server Instances

Service Desk management servers do not execute database and UI rules in parallel. They are executed in sequential order, and can trigger each other. In some circumstances this can cause delays on the server.

Symptoms

A large queue of database and UI rules can create a high CPU load on the management server, long response times from clients, and average load on the database. You can display the current queue size on the server by opening the **Server Monitor**, and selecting the **Queues** tab.

Solution

Tune the database and UI rules. For instructions, see “Database and UI Rules” on page 52.

Increase the number of management servers so that database and UI rules can be executed in parallel.

Configure the server computer to run several management servers. This requires multiple processors and a large amount of RAM. For example, if the CPU load on the server computer is relatively low and there is 4 GB of memory available, you can add two additional management servers and allocate 1 GB of memory to each. Reserve 1 GB of memory for the operating system.

For more information, refer to the section titled “Java Heap Size” on page 59 and also the *HP OpenView Service Desk Administrator’s Guide*.

3 Logging

This chapter discusses the following topics:

- User and environment variables, used to indicate the location of log files

- Installation log files
- XPL log files:
  - Type of log file produced
  - Syntax for naming log files
  - Relating log files to specific processes
  - Adjusting log file settings
  - Switching logging on and off
  - Viewing a binary log file

User and Environment Variables Related to Log Files

User and environment variables are used to indicate the location of log files generated by Service Desk. These variables are listed below.

User Variables

The following user variables are used when storing installation log files:

- **Temp** (Windows operating systems)

  For example, `%USERPROFILE%\Local Settings\Temp`

- **tmp** (UNIX-based operating systems)

  For example, `var/tmp`

Environment Variables

The following environment variables are used when storing log files:

- **UserName**

  The login name used to access the machine, for example `Administrator`.

- **OvInstallDir**

  The Service Desk installation directory. After installation, this value is also stored as the operating system environment variable `OvInstallDir`.

  The default installation paths are as follows:

  Windows

  `C:\Program Files\HP OpenView`

  UNIX

  `/opt/OV`

- **OvDataDir**

  The data directory of the Service Desk installation. After installation this value is also stored as the operating system environment variable `OvDataDir`. The default paths for the data directory are as follows:

Windows

```
C:\Program Files\HP OpenView\data
```

UNIX

```
/var/opt/OV
```

Service Desk Installation Log Files

At installation, several log files are generated. This section discusses the following log files:

- A *configuration log file* that stores information on the settings selected by the user during installation of each Service Desk component.
- An *installation log file* that records the progress of the installation for each component.

Configuration Log File

While a component is being installed, a configuration file `ovinstallparams.ini` is stored in:

Windows

```
<%TEMP%>\HPOvInstaller
```

UNIX

```
$tmp/HPOvInstaller
```

Once the component has been successfully installed, the date and time string (`_<date><time>`) is appended to the name of the file, and the file is moved to one of the locations shown below.

Windows

```
%TEMP%\HPOvInstaller\<appName>_<revision>\
```

UNIX

```
$tmp/HPOvInstaller/<appName>_<revision>/
```

`<appName>` refers to the executable (for example, client, server, or agent), and `<revision>` refers to the build identifier (for example `5.00.722`).

Installation Log File

For each installed component, an installation log file named `<appName>_<date>_HPOvInstallerLog.html` is saved to the following location:

```
<tempdir>/HPOvInstaller/<appName>_<revision>/
```

where:

`<tempdir>` is either `%TEMP%` or `$tmp`, for the appropriate operating system.

`<appName>` refers to the component executable (for example, client, server, or agent).

`<revision>` refers to the build identifier (version of the installer), for example `5.00.722`.

The file is produced in both HTML and text formats. The parameters `<date>` and `<time>` identify the date and the time the installation was started.

For example, for a client installation on a Windows operating system, a log file named:

```
client_2005.722_2006.02_11_26_HPOvInstallerLog.html
```

is saved to the following location:

```
C:\<%Temp%>\HPOvinstaller\client_2005.722\
```

Searching Installation Logs for Errors

If an installation fails or aborts, search the installation log file for the `ERROR` keyword. This can be a starting point for locating the source of the problem.

To investigate an installation problem:

1. Search the installation log file for the package that reported the error (for example, `HPOvObsSv`).
2. Examine the package log file to find details of the installation error. For example:

   Windows

   ```
   C:\Documents and Settings\user\Local
   Settings\Temp\HPOvInstaller\server_5.10.020\Package_msi_
   HPOvObsSv_install.log
   ```

   UNIX

   ```
   /var/tmp/HPOvInstaller/server_5.10.020/Package_sd_HPOvOb
   sSv_install.log
   ```

XPL Log Files

This section discusses log files related to XPL tracing:

- The types of log files produced (binary and text).
- The convention used to name log files.
- Relating log files to processes, and tracking log file generation.
- Rotating log files to control their size.
- Setting the defaults for properties such as file size, and the number of log files.
- Switching logging on and off.

About Binary and Text Log Files

Two types of log file are created:

- Binary
- Text

By default only the `ovcd` process (ovc control daemon) creates a binary log file, `system.bin`. The other processes in Service Desk create log files, in plain text only, for the current locale (the language set by the locale environment variable), and in US English (default). Log files are stored in the directory `OvDataDir/log` and in its subdirectories.

Binary files are locale-neutral. For example, if a binary file is generated on a Japanese-language system, the messages are displayed in Japanese, using the `ovlogdump` tool (see “Viewing the Contents of a Binary Log File” on page 89). If you were to take the same log file, install it on an English-language system, then you could view the same text in English, using the `ovlogdump` tool.

Text log files are locale-specific. Messages are written to the text files in the language set by the locale environment variable. Text logging can be switched on or off (see “Switching Logging Off and On” on page 87).

Naming Log Files

For default XPL logging, the syntax for naming a log file is as follows:

```
"logname%u.%g."+locale_id
```

Each part of the file name is described below.

`logname`	Indicates the type of program that is logging. The log file name `System` is the default name and is used by the Object Server component, for example. A large part of Service Desk runs in the Object Server, therefore many Service Desk log messages are stored in the `System` log files. For a list of log file names, see the section “Relating Log File Names to Processes” on page 76.
`%u`	A number that uniquely identifies the process that is logging. See “Tracking Log File Generation” on page 78.
`%g`	A counter used when log files reach their maximum length and the files are rotated (see “Controlling the Size of Log Files” on page 82).
`locale_id`	Refers to the locale code. This allows localized log statements to be written. For a non-English language Service Desk installation, the log files are written in en_US and the other language. For a German language installation, for example, this would produce `system0.1.en_US` and `system0.1.de_DE`

NOTE Language abbreviations are listed in the ISO 639 standard, country abbreviations are listed in the ISO 3166 standard.

Relating Log File Names to Processes

As described in “Naming Log Files”, the initial string of the log file name (System, Rules, and so on) is determined by the process that generates the log file.

The list below shows the names of each log file and the associated process:

`System`	Used by Object Server processes, including the Service Desk agent.
`Rules`	Used by Object Server to log detailed messages about database and UI rules that are executed.
`slm`	Used by OvSdSLM process.
`OvisMA`	Used by OVIS Metric Adapter.
`OvpmMA`	Used by the OVPM Metric Adapter.
`OvsnMA`	Used by OVSN Metric Adapter.
`OvsdMA`	Used by OVSD Metric Adapter.
`OpenMA`	Used by Open Metric Adapter.

NOTE In Service Desk 5.0, there are no specific rules log files, output is sent to one of the system logs.

In Service Desk 5.10, debug output for the rules is redirected to a new file type: `rules0.0.en_US` through `rules9.0.en_US`.

The list below shows the names of the SLM reporting log files and the associated process:

`DimensionExporter`	Used by the Dimension Exporter.
`DataFeeder`	Used by the Reporting Data Feeder.
`DimensionManager`	Used by the Reporting Dimension Manager.
`sd_report_admin.log`	Used by Service Desk reporting.

For more information on process logging, that is, which process is logging to a specific log file, see “Tracking Log File Generation” on page 78.

On the OVPI report server, the following tables are used to store log messages:

`SLM_R_PROCEDURELOG`	For SLM reporting.
`RSDHD_PROCEDURELOG`	For the Service Desk reporting Help desk bundle.
`RSDCM_PROCEDURELOG`	For the Service Desk reporting Change Management bundle.

The log files for Apache are stored in:

Windows

```
OVDATADIR\log\apache\a
```

UNIX

```
/var/opt/OV/log/apache/a
```

Log messages directly related to the GUI are stored in the directories listed below.

Windows

```
C:\Documents and Settings\USERNAME\Application
Data\HP OpenView\log
```

UNIX

```
~/.ov/log
```

Tracking Log File Generation

This section discusses two topics related to log file generation and is divided into the two main operating system types:

- How to determine which process is writing to a log file.
- The sequence of log files generated by processes.

NOTE For information on the `ovc` command, which controls the starting and stopping, event notification, and status reporting of all components registered with the HP OpenView control service, see the HP OpenView Service Desk online help.

Logging in UNIX

`fuser` command

The `fuser` command shows which process is writing to a specified log file. For example, to determine which process writes to the `system1.0.en` log file, enter the following at the command line:

```
# fuser -u system1.0.en
```

This command returns the following:

```
>system1.0.en: 28162o(root)
```

This shows that the process with `ID 28162` has the file open (`o` suffix = open). Then, to identify the process enter the following at the command line:

```
# ps -ef | grep 28162
```

This command returns the following:

```
>root 28162 28154 0 May 31 ? 48:44
/opt/OV/nonOV/jre/1.4/bin/PA_RISC2.0/java
-Did=OvObsServer -XX:
```

This shows that Java process `OvObsServer` (Object Server) writes to the `system1.0.en` log file.

`lsof` command

You can also use the UNIX diagnostic tool `lsof` to view a list of files open by a specified process.

Example 3-1 Java Processes and Log Files on an HP-UX 11 Server

Table 3-1 shows an example of Java processes and the associated log files on an HP-UX 11 server.

Table 3-1 Java Processes and Log Files

Log File	Process	Description
`system0.0.en`	`-DjmcId=ovjm ovloginsv`	Login Server
`system1.0.en`	`-Did=OvObsServer ovobs`	Object Server
`system2.0.en`	`-Did=OvsdMA ovsdma`	Metric Adapter
`system3.0.en`	`-Did=OvSlmServer ovsdslm`	Service Level Management
`system4.0.en`	`-Did=OvObsAg ovobsag`	Object Server Agent
`system5.0.en`	`-Did=OvEntityExporter ovsdreport`	Service Desk Reporting

NOTE You can match the Java process names with entries in the configuration files (`/var/opt/OV/conf/ctrl/<name>.xml`) to find out to which Service Desk process it corresponds.

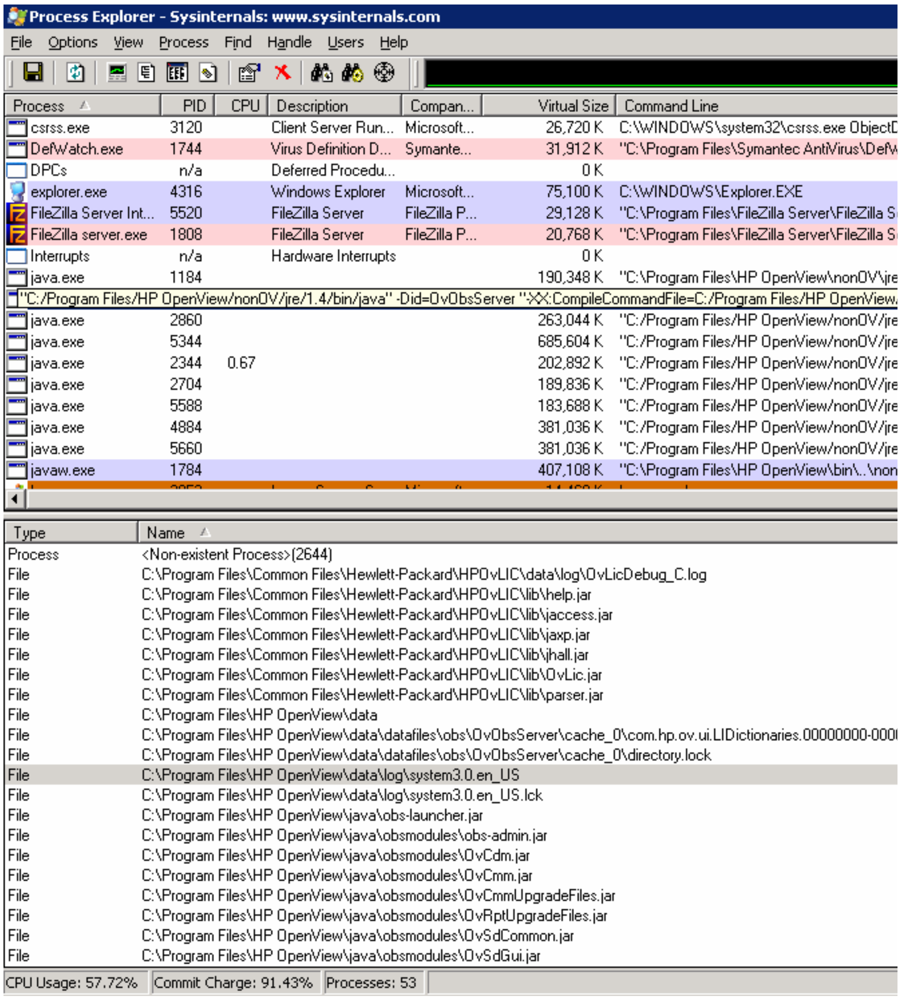
Logging in Windows

Use the Process Explorer from Sysinternals, available at the following URL:

http://www.sysinternals.com/Utilities/ProcessExplorer.html

Figure 3-1 shows that the selected Object Server (ovobs) process is using the `System3.0.en_US` log file. The `-Did=OvObsServer` in the command line indicates that this is the ovobs process:

Figure 3-1 **The Process Explorer**


Process Explorer - Sysinternals: www.sysinternals.com
File Options View Process Find Handle Users Help

Process	PID	CPU	Description	Compan...	Virtual Size	Command Line
csrss.exe	3120		Client Server Run...	Microsoft...	26,720 K	C:\WINDOWS\system32\csrss.exe ObjectD
DefWatch.exe	1744		Virus Definition D...	Symante...	31,912 K	"C:\Program Files\Symantec AntiVirus\DefW
DPCs	n/a		Deferred Procedu...		0 K	
explorer.exe	4316		Windows Explorer	Microsoft...	75,100 K	C:\WINDOWS\Explorer.EXE
FileZilla Server Int...	5520		FileZilla Server	FileZilla P...	29,128 K	"C:\Program Files\FileZilla Server\FileZilla S
FileZilla server.exe	1808		FileZilla Server	FileZilla P...	20,768 K	"C:\Program Files\FileZilla Server\FileZilla S
Interrupts	n/a		Hardware Interrupts		0 K	
java.exe	1184				190,348 K	"C:\Program Files\HP OpenView\nonOV\jre
"C:/Program Files/HP OpenView/nonOV/jre/1.4/bin/java" -Did=OvObsServer "-XX:CompileCommandFile=C:/Program Files/HP OpenView						
java.exe	2860				263,044 K	"C:/Program Files/HP OpenView/nonOV/jre
java.exe	5344				685,604 K	"C:/Program Files/HP OpenView/nonOV/jre
java.exe	2344	0.67			202,892 K	"C:/Program Files/HP OpenView/nonOV/jre
java.exe	2704				189,836 K	"C:/Program Files/HP OpenView/nonOV/jre
java.exe	5588				183,688 K	"C:/Program Files/HP OpenView/nonOV/jre
java.exe	4884				381,036 K	"C:/Program Files/HP OpenView/nonOV/jre
java.exe	5660				381,036 K	"C:/Program Files/HP OpenView/nonOV/jre
javaw.exe	1784				407,108 K	"C:\Program Files\HP OpenView\bin\..\non

Type	Name
Process	<Non-existent Process>(2644)
File	C:\Program Files\Common Files\Hewlett-Packard\HPOvLIC\data\log\OvLicDebug_C.log
File	C:\Program Files\Common Files\Hewlett-Packard\HPOvLIC\lib\help.jar
File	C:\Program Files\Common Files\Hewlett-Packard\HPOvLIC\lib\jaccess.jar
File	C:\Program Files\Common Files\Hewlett-Packard\HPOvLIC\lib\jaxp.jar
File	C:\Program Files\Common Files\Hewlett-Packard\HPOvLIC\lib\jhall.jar
File	C:\Program Files\Common Files\Hewlett-Packard\HPOvLIC\lib\OvLic.jar
File	C:\Program Files\Common Files\Hewlett-Packard\HPOvLIC\lib\parser.jar
File	C:\Program Files\HP OpenView\data
File	C:\Program Files\HP OpenView\data\datafiles\obs\OvObsServer\cache_0\com.hp.ov.ui.LIDictionaries.00000000-000
File	C:\Program Files\HP OpenView\data\datafiles\obs\OvObsServer\cache_0\directory.lock
File	C:\Program Files\HP OpenView\data\log\system3.0.en_US
File	C:\Program Files\HP OpenView\data\log\system3.0.en_US.lck
File	C:\Program Files\HP OpenView\java\obs-launcher.jar
File	C:\Program Files\HP OpenView\java\obsmodules\obs-admin.jar
File	C:\Program Files\HP OpenView\java\obsmodules\OvCdm.jar
File	C:\Program Files\HP OpenView\java\obsmodules\OvCmm.jar
File	C:\Program Files\HP OpenView\java\obsmodules\OvCmmUpgradeFiles.jar
File	C:\Program Files\HP OpenView\java\obsmodules\OvRptUpgradeFiles.jar
File	C:\Program Files\HP OpenView\java\obsmodules\OvSdCommon.jar
File	C:\Program Files\HP OpenView\java\obsmodules\OvSdGui.jar

CPU Usage: 57.72% | Commit Charge: 91.43% | Processes: 53

Configuring the Level of Logging

The log level determines the level of detail for logged messages. The lowest level displays the most detailed information (similar to tracing). You can set the following logging levels:

- SEVERE (highest value)
- WARNING
- INFO
- CONFIG
- FINE
- FINER
- FINEST (lowest value)

To set the logging level:

1. Create the file `logprop.txt` in `<OVDATADIR>` with the following contents:

   - Add the console log handler:

     ```
     handlers=java.util.logging.ConsoleHandler
     ```

   - Logging level for SLM processes (also the default global logging level):

     ```
     level=FINER
     ```

   - Logging level for the console:

     ```
     java.util.logging.ConsoleHandler.level=WARNING
     ```

   - Logging formatter for the console:

     ```
     java.util.logging.ConsoleHandler.formatter=java.util.
     logging.SimpleFormatter
     ```

   - Logging level for the Object Server processes:

     ```
     com.hp.ov.obs.level=FINER
     ```

2. Modify the process Control registration files (see “Control Configuration” on page 142) and add in the start-hook:

   ```
   -Djava.util.logging.config.file="OVDATADIR\logprop.txt"
   ```

For example, modify the `OvObs.xml` file (in `OVDATADIR\conf\obs`) as follows:

```
-Did=OvObsServer
-Djava.util.logging.config.file=
"C:\Program Files\HP OpenView\data\logprop.txt"
```

Modify the `OvSdSlmCtrl.xml` file (in `OVDATADIR\conf\slm`) as follows:

```
-Did=OvSlmServer -Djava.util.logging.config.file=
"C:\Program Files\HP OpenView\data\logprop.txt"
```

3. Register the Control registration files that you changed (see section "Control Configuration" on page 142).
4. Restart the processes.

NOTE Production systems need higher logging levels. Logging levels lower than INFO are not recommended. This is because each log message blocks the execution of the processes. A low logging level can have dramatic effects on performance.

For more information, refer to the following URL:

http://java.sun.com/j2se/1.4.2/docs/guide/util/logging/overview.html

Controlling the Size of Log Files

To prevent log files from growing excessively, log file rolling is used to create multiple smaller files that can be archived or removed. You can specify a maximum number of log files to be created, and the maximum size of each log file (see "Other Log File Settings" on page 83). Whenever a log entry causes a log file to exceed the maximum size, the file is closed and a new file is opened, using the next sequence number as part of its name. This is illustrated in Example 3-2.

Example 3-2 **Rolling Log Files Example**

In this example, the logging process initially writes to a log file named `system1.0.en_US`. When the file is full (its limit is specified in the log file settings), it is renamed to `system1.1.en_US` and a new system1.0.en_US file is created. The logging process continues to write to system1.0.en_US. When the new file is full, system1.1.en_US is renamed to system1.2.en_US, the existing, full system1.0.en_US file is renamed to system1.1.en_US, and a new, empty system1.0.en_US file is created.

Multiple Java Processes and Log Files

For multiple Java processes using XPL, several logfiles are generated. For example, if three Java processes utilizing XPL are started in the order shown below, the log files generated are as follows:

Process1: System0.0.en_US

Process2: System1.0.en_US

Process3: System2.0.en_US

If you stop Process2 and restart it, Service Desk uses the first SystemX file not in use by another process, in this case the System1.0.en_US file.

If you stop all processes and restart them, this time in a different order, the following shows the log files used by the processes in this case:

Process1: System1.0.en_US

Process2: System0.0.en_US

Process3: System2.0.en_US

Other Log File Settings

Use the ovconfchg tool to change the logging properties listed in Table 3-2. For example, to set the maximum number of number of log files to be created by any logging entity, enter the following at the command line:

```
ovconfchg -namespace xpl.log.OvLogFileHandler
-set filecount 20
```

On a Windows operating system, the file

<DataDir>\conf\xpl\config\local_settings.ini is adjusted to reflect the change.

Using the ovconfchg tool, you can change the properties listed below. These are described in detail in Table 3-2.

- Locales

- Handlers, in addition to the default handlers for logging
- Default action for the OV root logger
- Default action for application-specific file handlers
- Maximum number of files to be created for any specific logging entity
- File size limit for any log file

Table 3-2 Configuration Properties for Log Files

Namespace	Property	Default Value	Description
`xpl.log`	`addlocales`	`none`	A whitespace-separated list of additional locales for which log files will be generated. The component will automatically open a handler for the current locale. If not English-based, `en_US` will also be added. The specified property can add any number of locales. Locales will conform to the format specified in“Naming Log Files” on page 75.
`xpl.log`	`handlers`	`none`	A whitespace-separated list of handlers that will be added to the default handlers for logging, in addition to OvLogFileHandler. Potential handlers are those provided by Sun Java logging, OvSupportHandler, and OvConsoleHandler.

Table 3-2 Configuration Properties for Log Files (Continued)

Namespace	Property	Default Value	Description
`xpl.log`	`logparent`	`false`	Sets the default action for the HP OpenView root logger. If set to `true`, log messages are passed to all log handlers in the JRE. Set to false, logs are passed only to handlers in the `com.hp.ov` log namespace. The default for a JRE is a single console handler. Thus, setting this to true causes all logs to be sent to the Java console as well as HP OpenView registered handlers.
`xpl.log`	`apSpecifc UseParent`	`true`	Sets the default action for application specific file handlers. If set to `true`, log messages are written into both the system-wide log and the application-specific log. Set to `false`, logs are written only into the application-specific log. Setting this flag affects all HP OpenView logging. Applications may use the Sun Java 1.4 API method Logger.set UseParentHandlers to control the operation of a single application specific logger.
`xpl.log. OvLogFile Handler`	`filecount`	`10`	The maximum number of log files to be created for any given logging entity.

Table 3-2 Configuration Properties for Log Files (Continued)

Namespace	Property	Default Value	Description
`xpl.log. OvLogFile Handler`	`filesize`	`1`	The file size limit for any log file, in millions of bytes. Whenever a log entry is written that causes the log file to exceed this value, a new log file is created as specified in "Naming Log Files" on page 75.

NOTE If permissions are inadequate for the default locations, log files will be stored in the `<*Data dir*>/log/public` directory.

Location of Configuration and Log Files

The configuration and log files are stored in the following locations:

Windows Configuration and Log Files

- **Configuration file**
  - `C:\Program Files\HP OpenView\Data\conf\xpl\config\local_settings.ini`
  - `C:\Program Files\HP OpenView\Data\conf\xpl\log\log.cfg`
- **Binary log files**
  - `C:\Program Files\HP OpenView\Data\log`
- **Text log files**
  - `C:\Program Files\HP OpenView\Data\log`

UNIX Configuration and Log Files

- **Configuration file**
  - `/var/opt/OV/conf/xpl/log/log.cfg`
  - `/var/opt/OV/conf/xpl/config/`
- **Binary log files**
  - `/var/opt/OV/log`
- **Text log files**
  - `/var/opt/OV/log`

Switching Logging Off and On

You can switch logging off and on by changing the settings in the logging configuration file `log.cfg`.

The file contains a keyword and value pair per line. Blanks, blank lines, or comments are not allowed. The keywords are case sensitive.

The keywords are:

`TextLog=off|on`

Switches locale-specific text logging off and on. Switching this on causes the logging libraries to write a locale-specific log in human-readable format. The default value is `off`.

`OVLog=off|on`

Switches binary logging off and on. The default value is `on`.

CAUTION Switching off this option will prevent any binary logging for any component.

`SysLog=off|on`

Switches logging by `syslog` off and on.

`NTLog=off|on`

Switches logging by the Windows event log off and on. It has no effect on UNIX systems. Default is `off`.

`OVEvents=off|on`

Switches message forwarding to the HP OpenView Operations event system off or on. Default is `on`.

See “Location of Configuration and Log Files” on page 86 for information about the location of the configuration and log files on UNIX and Windows systems.

Viewing the Contents of a Binary Log File

Use the `ovlogdump` command to dump a binary log file as text in the current locale to the console. This allows you to view the contents of the text file and use the information to help you troubleshoot problems.

ovlogdump(1)

NAME

ovlogdump – dumps a specified binary log file as text in the current locale to the console

SYNOPSIS

```
ovlogdump -h|-help

ovlogdump -version

ovlogdump [<binary_logfile_name>]

ovlogdump -merge -tofile <binary_logfile_name>
-fromfiles <binary_logfile1_name>
<binary_logfile2_name>...
```

DESCRIPTION

The `ovlogdump` command dumps a binary log file as text in the current locale to the console. To view the contents of a log file, specify its location and name, otherwise the `system.bin` file is dumped to the console by default.

By default, all the log files are stored in the following location:

On Windows:	`C:\Program Files\HP OpenView\Data\log`
On UNIX:	`/var/opt/OV/log`

If permissions are inadequate for the default locations, the log files are stored in the <*OvDataDir*>`/log/public` directory.

During application logging, if multiple log files are created, you can use the `–merge` option to merge these files into a single binary log file.

Parameters

`ovlogdump` recognizes the following options:

[<*binary_logfile_name*>]

The name and location of the binary log file to be dumped. If the log file name is not specified, `system.bin` file in the <*OVDataDir*>`/log/` directory is displayed on the console by default.

`-merge -tofile <`*`binary_logfile_name`*`> -fromfiles <`*`binary_logfile1_name`*`> <`*`binary_logfile2_name`*`>....`

Merges application log files specified by <*binary_logfile1_name*>.... into a single binary log file specified by <*binary_logfile_name*>. This option is not supported for merging system log files.

`-h|-help`

Displays all available options for the `ovlogdump` command.

`-version`

Displays the version of the `ovlogdump` command.

AUTHOR

`ovlogdump` was developed by Hewlett-Packard Company.

4 HP OpenView Tracing Fundamentals

Troubleshooting and Tracing

HP OpenView Tracing helps you investigate the cause of problems in your HP OpenView product. Using tracing as a troubleshooting procedure provides you with trace log files that can pinpoint when and where a problem has occurred, such as if processes or programs abort, performance is greatly reduced, or unexpected results appear.

You can activate tracing for specific applications, daemons, processes, or services. To simplify the interpretation of the trace log file, you can activate tracing for particular functional areas by specifying one or more functional areas in the trace statement.

HP OpenView Tracing

This section covers the following topics:

- "Tracing Architecture" on page 95.
- "Tracing Tools" on page 96.
- "Trace Configuration Files" on page 97.
- "Trace Server Log Information" on page 103.

Tracing Architecture

HP OpenView Tracing consists of the following major components:

- **Trace-enabled application**: A program (executable, daemon, process, or service) that has incorporated HP OpenView Tracing.

  HP OpenView Tracing is a component included in various HP OpenView products, such as HP OpenView Operations and Network Node Manager. Note that all processes of a product may not use the tracing feature. For more information about tracing in a HP OpenView product, refer to the appropriate product documentation.

  Tracing can be enabled by application. Components and categories can be further added to create a logical or functional subdivision for the trace. Each combination of component and category creates a unique trace object that may be controlled independently of others within the application. For more information about configuring tracing, refer to "Trace Configuration Files" on page 97.

- **Tracing clients**:

  — Configuration client: Configures the trace server with trace configuration information set by the user. A standard text editor (see "Trace Configuration Files" on page 97) or the `ovtrcgui` tool (see "Tracing Tools" on page 96) can be used to create configuration information.

  — Monitor client: Receives trace messages from the trace server and is capable of writing the trace output to a file or directly to standard out.

- **Trace server**: A program that acts as a proxy for sending trace configuration information from a configuration client to a trace-enabled application. The program then sends trace messages from the trace-enabled application to the monitor client. The communication is established using TCP/IP sockets.

  The `ovtrcadm` tool is provided to help you administer the trace server. It enables you to add and remove systems hosting the applications to be traced and view the trace configuration of all the applications. Use this tool to shut down the trace server. Do not kill the trace server process. For more information about ovtrcadm, refer to "Tracing Tools" on page 96.

NOTE The trace server and the configuration client can reside on local or remote systems. However, the trace server and the trace-enabled application must reside on the same system.

Tracing Tools

By default, the tracing tools are installed during the HP OpenView product installation. The default installation and data directory information is shown in Table 4-1.

Table 4-1 **Installation and Data Directory Information**

Operating System	Installation Directory	Data Directory
HP-UX, Solaris, Linux	`/opt/OV/`	`/var/opt/OV/`
Windows	`C:\Program Files\ HP OpenView\`	`C:\Program Files\ HP OpenView\data\`
AIX	`/usr/lpp/OV/`	`/var/opt/OV/`
Tru64	`/usr/opt/OV/`	`/usr/var/opt/OV/`

The tracing tools available for UNIX and Windows operations are listed in Table 4-2.

Table 4-2 Tracing Tools

Tracing Component	Tool Name and Path Information	
	UNIX	Windows

Trace server	`ovtrcd`
	`<`*`install_dir`*`>/lbin/xpl/trc/`
		`ovtrcsvc`
		`<`*`install_dir`*`>\bin\`

Configuration client	`ovtrccfg`
	`<`*`install_dir`*`>/support/`
		`ovtrcgui`
		`<`*`install_dir`*`>\support\`

Monitor client	`ovtrcmon`
	`<`*`install_dir`*`>/support/`
		`ovtrcgui`
		`<`*`install_dir`*`>\support\`

Administration tool for trace server	`ovtrcadm`
	`<`*`install_dir`*`>/support/`
		`ovtrcadm (command-line tool)`
		`<`*`install_dir`*`>\support\`

On Windows, the tracing tools are accessible from the command line and from a graphical user interface. It is recommended that you use the graphical interface tool, `ovtrcgui`, to configure and monitor tracing. The tool provides features, such as saving trace configuration to a file, editing trace configurations, loading trace output from a file, saving trace output to a file, exporting trace output in ASCII format, and analyzing trace output using filters and different views. The supported operating systems are Windows NT®, Windows 2000, Windows 2003, and Windows XP®.

Trace Configuration Files

Trace configuration files are ASCII text files that hold configuration information for tracing. The files can be created using a standard text editor or the `ovtrcgui` tool. The files use the extension `tcf`.

Do not use the name `OVTrace.tcf`, as this is a reserved file name.

The trace server maintains the state of tracing for each trace-enabled application until an updated configuration is sent to the trace-enabled application or until tracing is disabled.

The syntax of a trace configuration file takes the following form:

Syntax

```
TCF Version <version_number>
APP: "<application_name>"
SINK: File "<file_name>" "force=[1];
maxfiles=[1..100];maxsize=[0..1000];"
SINK: Socket "<node>" "node=<node_name>;"
TRACE: "<component_name>" "<category_name>"
<keyword_list>
```

Each line of the syntax is described in detail in the following sections.

TCF Version

The TCF version line specifies that this is a trace configuration file and also specifies the version number of the file. It is case-sensitive and must be specified exactly as shown below:

Syntax:

```
TCF Version <version_number>
```

Example:

```
TCF Version 1.1
```

APP

The application line defines the name of the application to be traced. It must start with `APP` followed by a colon (`:`) and a space. The application name should be specified within double quotes (`"..."`). Multiple applications can be specified for the trace. Repeat this pattern for each application that you want to trace.

Syntax:

```
APP: "<application_name>"
```

Example:

```
APP: "dbmanager"
APP: "opcmsg"
APP: "poller"
```

SINK

The sink line specifies the target to which trace output is directed. The target can be a file or a system running the trace server. The line must begin with `SINK` followed by a colon (`:`) and a space (` `). The arguments on the line should be separated by spaces.

The `SINK` line has three arguments.

The first argument must be one of the two keywords, `File` or `Socket`, without double quotes (`"..."`). This represents the type of Sink.

The second argument is the name of the type of Sink and must be in double quotes (`"..."`). If the sink type is `File`, then this argument is the name of the file. If the sink type is `Socket`, then this argument is the name of the system running the trace server to which you want the application to send the trace messages.

The third argument is additional sink type options. The options must be in double quotes (`"..."`), and each option must be followed by a semi-colon (;).

The options for sink type `File` are:

- `force=n`
- `maxfiles=n`
- `maxsize=n`

force The `force` option is followed by an integer value `n` that is not zero. If the value is 1, trace output is buffered until the buffer is full before flushing the output to disk. Buffering can improve performance, but it also means that if the application crashes, then the last traces may not get written to disk. If the value is greater than 1, then the tracing subsystem forces the trace output to the disk after each trace event is written. The default is `force=1`.

maxfiles The `maxfiles` option is followed by an integer value between 1 and 100. This option allows you to specify the number of historic trace log files to be retained. Each time an application starts to trace to the file, a backup is made of the previous file (if any) by adding ".001" to the name and renaming the file. If there was a ".001" file already, then it is renamed to ".002" and so on. The same backup scheme is in effect if the current log file reaches the maximum size.

maxsize The `maxsize` option is followed by an integer or float value between 0 and 1000 that specifies the maximum amount of disk space in megabytes (MB) to be used for each file.

If the last block of trace output written to the file causes the file to be larger than the specified maximum, then the next output causes the current output file to be closed and backed up and a new output file to be created. A value of 0 is a special case that lets the file grow until you run out of disk space.

NOTE When target is set as file, use the `ovtrcmon` or `ovtrcgui` tool to view the contents of the file. Formatting is not orderly when the file is opened using standard text editors.

Syntax:

```
SINK: File "<file_name>" "force=[1];
maxfiles=[1..100];maxsize=[0..1000];"
```

Example:

```
SINK: File "C:\\TEMP\Output.trc"
"force=1;maxfiles=10;maxsize=100;"
```

For the sink type `Socket`, one option is supported:

- `node=<node_name>`

The value of `node_name` is the communication path to the system on which the trace server is running and to where the trace output must be sent. It can be a DNS name or an IP address. If you want to create a configuration file that sends the output to the local trace server, regardless of the system to which you copy the configuration file, you must set the sink type to `localhost` and remove any `node=` option from the options string.

Syntax:

```
SINK: Socket "<node>" "node=<node_name>;"
```

Example:

```
SINK: Socket "bigfoot" "node=161.114.22.105;"
```

TRACE

The trace line must begin with `TRACE` followed by a colon (`:`) and a space ( ). The arguments on the line must be separated by spaces.

The first argument is the trace component name and it must be in double quotes (`"..."`). Multiple components can be specified.

The second argument is the trace category name and it must also be in double quotes (`"..."`). Multiple components can be specified. If you are using one of the standard categories in the code, it is mapped to the string value which you specify here. For the exact mapping of standard category constants to string values, see the language-appropriate documentation (C++, or Java).

Syntax:

```
TRACE: "<component_name>" "<category_name>"
<keyword_list>
```

Example:

```
TRACE: "database" "Parms" Error Info Warn
TRACE: "xpl.io" "Trace" Info
```

You can use the wildcard "`*`" as the component name, category name, or both. This is useful when using applications in the mode where they read their configuration information directly from a file.

When an application tries to determine the settings for `component A` and `category B`, it first looks to see if the configuration contains an explicit trace definition for this pair. If the trace definition is there, it uses these settings. If it is not, then it looks to see if there is a configuration for `component A` and category `*`. If there is, it uses these settings. If there is not, then if looks to see if there is a configuration for `component *` and `category *`. If there is, it uses those settings. If not, then the trace is not activated.

NOTE Configuration files that use the wildcard(`*`) feature cannot be loaded into `ovtrcgui` or `ovtrccfg` tools.

The remaining parameters are a variable list of keyword options. At least one of the keywords, `Error`, `Info`, or `Warn`, must be in the list. The supported keywords are shown in Table 4-3.

Table 4-3 **Trace Keywords**

Keyword	Description
`Error`	Enable traces marked as errors.
`Warn`	Enable traces marked as warnings.
`Info`	Enable traces marked as information
`Support`	Enables normal tracing. Trace output includes: • Informational messages • Warnings messages • Error messages This option is recommended for troubleshooting problems. You can enable tracing for a long duration as the overhead to capture trace output is minimal with this option.

`Developer`
`Verbose`
`Location`
`Stack`
	These keywords are targeted for trace messages that require source code access to be interpreted. They should be used when source-level detailed trace messages are requested by the HP Software Support Online representative who is helping you investigate the problem.
	Note that Verbose and Developer provide similar output. They are not distinct in nature.

Sample Trace Configuration Files

Sink to File

```
TCF Version 3.2
APP: "dbmanager"
APP: "dbpooler"
SINK: File "C:\\TEMP\\Output.trc"
"force=1;maxfiles=10;maxsize=100;"
```

```
TRACE: "DbManager" "Parms" Error Info Warn Developer
TRACE: "opcmsg" "Init" Info Verbose
TRACE: "bbc.rpc" "Proc" Info, Verbose
```

Sink to Socket

```
TCF Version 3.2
APP: "nodedisc"
SINK: Socket "mgtstation" "node=161.114.22.105;"
TRACE: "Discovery" "Event" Error Info Warn
TRACE: "opcpol" "Operation" Error Info Warn Developer
```

Trace Server Log Information

The trace server log information is captured in `trc.log` file. The path information is as follows:

UNIX	`<`*`data_dir`*`>/log/`
Windows	`<`*`install_dir`*`>\data\log\`

Tracing Methods

HP OpenView Tracing supports static and dynamic tracing.

Static Tracing

Static tracing is enabled during application startup and permits application startup traces to be captured. After the application has started, the tracing configuration cannot be modified. Static tracing can only be disabled by stopping the application. Unless the trace configuration file is renamed or moved, tracing resumes after the next application startup. Static tracing can be configured in two ways:

- Locating the `OVTrace.tcf` file in the application startup directory or in:

  Windows

  `<`*`install_dir`*`>\HP OpenView\data\datafiles\xpl\`

  UNIX

  `<`*`data_dir`*`>/datafiles/xpl/`

  The location of the application start-up directory is application dependent. The location depends on how the application starts. To make it simpler, it is recommended that you place the file in the location mentioned above than in the application start-up directory.

- Defining an environment variable named `TRACE_CONFIG_FILE` to specify the name and location of the trace configuration file. This method enables you to determine the configuration file name and location.

For more information about enabling static tracing, refer to "Enabling Static Tracing" on page 124.

Dynamic Tracing

Dynamic tracing is enabled after the application has started and is dynamically configurable. This method is a common choice for troubleshooting problems. Dynamic tracing can be configured using `ovtrccfg` or `ovtrcgui` together with a trace configuration file. Using `ovtrcgui`, you can also dynamically set trace configuration through Trace Configuration dialogs.

Using the configuration client, you can dynamically configure a trace server that is running on the local system or on a remote system. If you have direct access to the system, the local configuration option can be used. If you do not have direct access to the system, the remote configuration option must be used.

For more information about enabling dynamic tracing, see “Enabling Dynamic Tracing” on page 127.

NOTE If you have access to the `ovtrcgui` tool, it is easier to use the tool to configure tracing configuration remotely.

How to Trace

This section outlines a general methodology that can be applied in most cases where you need to enable tracing within an HP OpenView product. You need to use these steps as a general guideline and may need to make modifications to the procedure as necessary to fit the actual problem. The procedure assumes that a problem has been identified and that there is the need to capture application tracing information.

To install and configure HP OpenView tracing, complete the following steps. These steps are covered in detail in Installing and Configuring HP OpenView Tracing chapter.

1. Determine the applications (process/daemon) for which you need to capture trace information.
2. Determine whether the applications identified are trace-enabled.
3. Determine whether the trace-enabled application requires any pre-configuration steps to enable HP OpenView Tracing.
4. Determine which components and categories within each application need to be traced.
5. Determine which attribute flags associated with each component/category combination need to be set.
6. Determine the most suitable tracing configurations for the problem being experienced. Tracing can be static or dynamic, configured locally or remotely, written directly to a file, or transmitted through the trace server.
7. Create the trace configuration file to match the tracing configuration selected.
8. Depending on the trace configuration selected, enable either static or dynamic tracing.
9. Execute the application-specific commands necessary to duplicate the problem that initiated the requirement for tracing output. When the desired behavior has been duplicated, the tracing can be stopped or disabled.
10. If you are using Windows, the trace messages can be filtered on parameters such as components, categories, and severity.

11. Depending on the trace configuration selected, disable either static or dynamic tracing.
12. Gather the trace configuration file and the trace output files. Evaluate the trace messages or package the files for transfer to HP Software Support Online for evaluation.

5 Installing and Configuring HP OpenView Tracing

Auto Installation of Tracing Tools

The tracing tools are automatically installed during the HP OpenView product installation. To support dynamic trace configuration without additional steps, the trace server is configured to start when the system starts.

On Windows operating systems, this is done by installing the trace server, `ovtrcsvc`, as a Windows service called `HP OpenView Shared Trace Service`. On UNIX, the trace server is automatically started for you by the addition of `ovtrcd` to the startup script of the HP OpenView product.

Trace Server Startup on UNIX-based systems

The trace server is started by adding `ovtrcd` to the start-up script. This is automatically done during the HP OpenView product installation.

The trace server can be started and stopped using the following commands from the root account:

- Start trace server:

  ```
  <install_dir>/lbin/xpl/trc/ovtrcd
  ```

- Stop trace server:

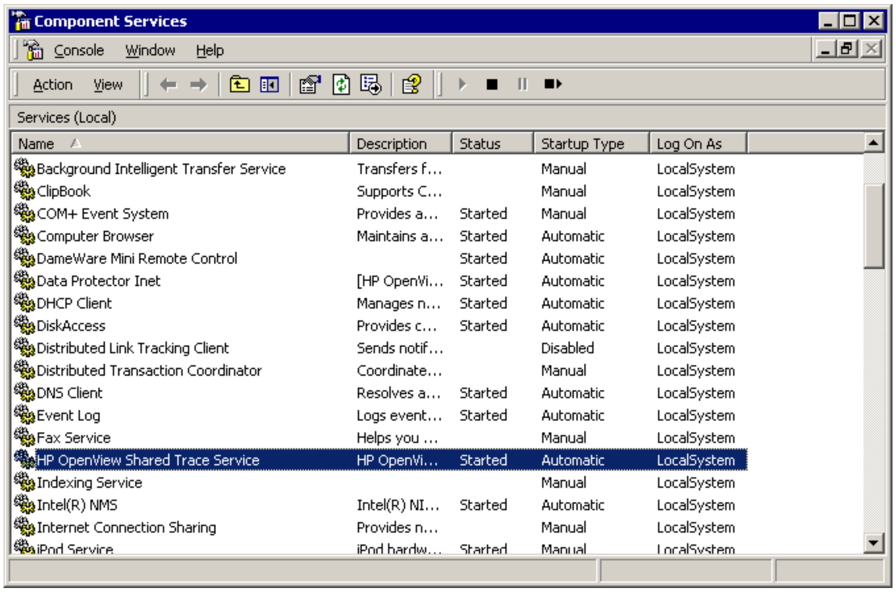
  ```
  <install_dir>/support/ovtrcadm-srvshutdown
  ```

CAUTION If the trace server is stopped and restarted, the tracing state for all applications is lost and cannot be reloaded without restarting all applications to be traced.

Trace Server Startup on Windows Operating Systems

The trace server is automatically installed and started for you by the addition of `ovtrcsvc` as a Windows service called `HP OpenView Shared Trace Service` (see Figure 2-1). This is automatically done during the HP OpenView product installation.

Figure 5-1 **HP OpenView Trace Shared Services in the Component Services Window**



Name	Description	Status	Startup Type	Log On As
Background Intelligent Transfer Service	Transfers f...		Manual	LocalSystem
ClipBook	Supports C...		Manual	LocalSystem
COM+ Event System	Provides a...	Started	Manual	LocalSystem
Computer Browser	Maintains a...	Started	Automatic	LocalSystem
DameWare Mini Remote Control		Started	Automatic	LocalSystem
Data Protector Inet	[HP OpenVi...	Started	Automatic	LocalSystem
DHCP Client	Manages n...	Started	Automatic	LocalSystem
DiskAccess	Provides c...	Started	Automatic	LocalSystem
Distributed Link Tracking Client	Sends notif...		Disabled	LocalSystem
Distributed Transaction Coordinator	Coordinate...		Manual	LocalSystem
DNS Client	Resolves a...	Started	Automatic	LocalSystem
Event Log	Logs event...	Started	Automatic	LocalSystem
Fax Service	Helps you ...		Manual	LocalSystem
HP OpenView Shared Trace Service	HP OpenVi...	Started	Automatic	LocalSystem
Indexing Service			Manual	LocalSystem
Intel(R) NMS	Intel(R) NI...	Started	Automatic	LocalSystem
Internet Connection Sharing	Provides n...		Manual	LocalSystem
iPod Service	iPod hardw...	Started	Manual	LocalSystem

Setting Up Tracing

To set up tracing of a problem, complete the following steps:

1. Determine the applications (process or daemon) for which you need to capture trace information.
2. Determine whether the applications identified as being the root of the problems being experienced are all trace-enabled (applications that incorporate HP OpenView Tracing). This procedure can be used only for trace-enabled applications. The non-trace-enabled applications need to use existing tracing mechanisms to capture trace output. Use the tools described below to find out which applications are trace-enabled.

   UNIX-based Operating Systems

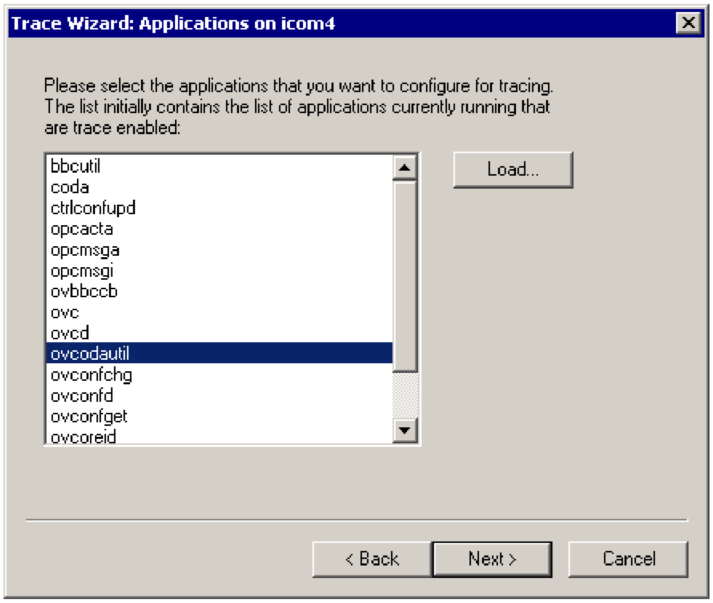
   Use the `ovtrcadm -srvconfig` option to view the names of all trace-enabled applications that have registered with the `ovtrcd` process, and the components and categories defined for each trace-enabled application.

   Windows Operating Systems

   Use the `ovtrcgui` tool to connect to the system running the trace server. The connection can be either local or remote. After a connection is established, you can see the names of the trace-enabled applications running on the system (see Figure 5-2).

CAUTION Avoid selecting all applications. The volume of trace information generated usually becomes too much to analyze efficiently.

Figure 5-2 Applications Dialog


Trace Wizard: Applications on icom4

Please select the applications that you want to configure for tracing. The list initially contains the list of applications currently running that are trace enabled:

```
bbcutil
coda
ctrlconfupd
opcacta
opcmsga
opcmsgi
ovbbccb
ovc
ovcd
ovcodautil
ovconfchg
ovconfd
ovconfget
ovcoreid
```

3. Determine if the trace-enabled application requires any additional pre-configuration steps to enable tracing. In general, this should not be necessary, but the implementation of tracing in some HP OpenView products does require additional pre-configuration steps. For example, the `NNM/ET` processes require that the process `lrf` file be modified to output the trace messages.

NOTE Refer to appropriate HP OpenView product documentation to determine which applications require pre-configuration steps.

4. Determine which components and categories within each application need to be traced.

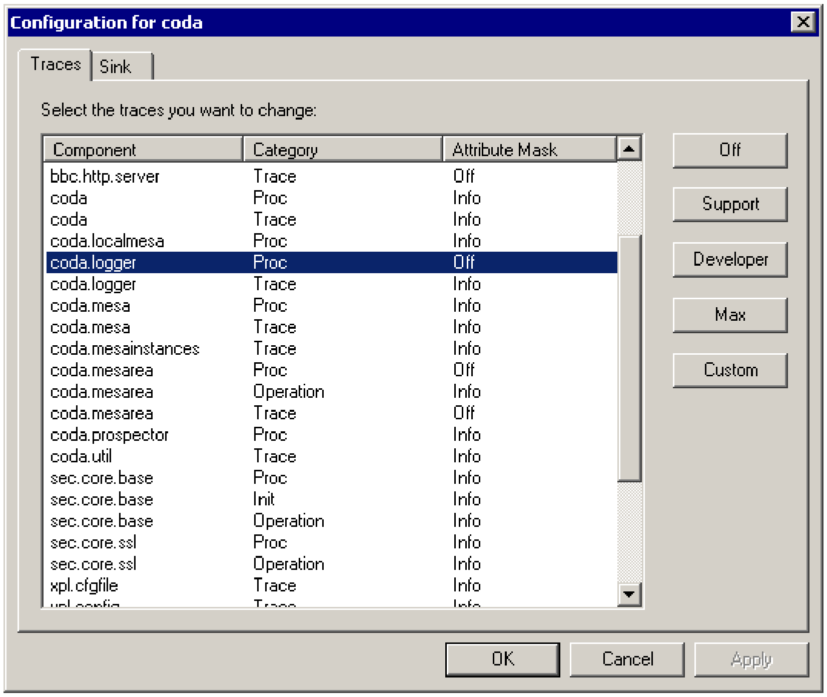
   UNIX-based Systems

   Manually enter the component and category names into the trace configuration file. For more information, refer to "Trace Configuration Files" on page 97.

Windows operating systems

Use the `ovtrcgui` tool to select the components and categories.

Figure 5-3 Trace Configuration Dialog



Select the traces you want to change:

Component	Category	Attribute Mask
bbc.http.server	Trace	Off
coda	Proc	Info
coda	Trace	Info
coda.localmesa	Proc	Info
coda.logger	Proc	Off
coda.logger	Trace	Info
coda.mesa	Proc	Info
coda.mesa	Trace	Info
coda.mesainstances	Trace	Info
coda.mesarea	Proc	Off
coda.mesarea	Operation	Info
coda.mesarea	Trace	Off
coda.prospector	Proc	Info
coda.util	Trace	Info
sec.core.base	Proc	Info
sec.core.base	Init	Info
sec.core.base	Operation	Info
sec.core.ssl	Proc	Info
sec.core.ssl	Operation	Info
xpl.cfgfile	Trace	Info

After you have identified the components and categories that are available, determine which ones relate to the problem for which you want to establish tracing.

CAUTION Avoid selecting all applications. The volume of trace information generated usually becomes too much to analyze efficiently.

5. Determine the attribute to be associated with each component/category combination. The recommended starting points are the usual `Support` attributes: `Info`, `Warn`, and `Error`. Use these

starting points unless you are directed to set the `Verbose` or `Developer` attributes by your HP Software Support Online representative, as shown in Table 5-1.

Table 5-1 Trace Attribute Flags

Attribute Mask	Trace Keyword	Recommendations
`Support`	`Info`	**Normal tracing** Recommended for all troubleshooting purposes. The overhead to capture information is low and tracing can be enabled on the system for long durations.
	`Warnings`	
	`Error`	
`Developer` `Max` `Custom`		**Advanced Tracing** Trace messages require source code access to be interpreted. These attributes should be used when source level detailed trace messages are requested by the HP Software Support Online representative who is helping you investigate the problem. Note that Verbose and Developer provide similar output. They are not distinct in nature.

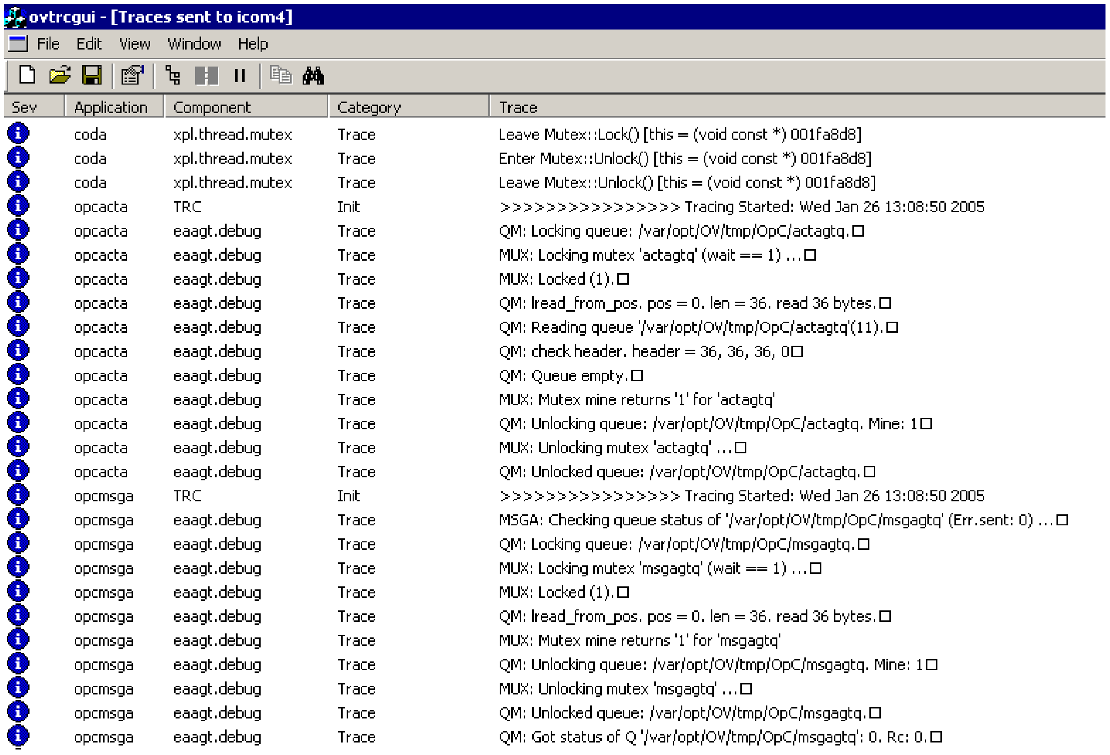
NOTE For UNIX-based operating systems, the attribute is manually entered in the trace configuration file, as described in “Trace Configuration Files” on page 97.

6. Determine which tracing configuration is most suitable for the tracing situation. Tracing can be static or dynamic, configured locally or remotely, written directly to a file, or transmitted from the trace server.

NOTE If working across a firewall, the tracing messages can only cross a firewall if port 5053 (TCP) is open.

The trace output from both static and dynamic tracing can be written directly to a file or transmitted to a local or remote trace server.

Figure 5-4 Trace Output Dialog



Sev	Application	Component	Category	Trace
ℹ	coda	xpl.thread.mutex	Trace	Leave Mutex::Lock() [this = (void const *) 001fa8d8]
ℹ	coda	xpl.thread.mutex	Trace	Enter Mutex::Unlock() [this = (void const *) 001fa8d8]
ℹ	coda	xpl.thread.mutex	Trace	Leave Mutex::Unlock() [this = (void const *) 001fa8d8]
ℹ	opcacta	TRC	Init	>>>>>>>>>>>>>>>> Tracing Started: Wed Jan 26 13:08:50 2005
ℹ	opcacta	eaagt.debug	Trace	QM: Locking queue: /var/opt/OV/tmp/OpC/actagtq.□
ℹ	opcacta	eaagt.debug	Trace	MUX: Locking mutex 'actagtq' (wait == 1) ...□
ℹ	opcacta	eaagt.debug	Trace	MUX: Locked (1).□
ℹ	opcacta	eaagt.debug	Trace	QM: lread_from_pos. pos = 0. len = 36. read 36 bytes.□
ℹ	opcacta	eaagt.debug	Trace	QM: Reading queue '/var/opt/OV/tmp/OpC/actagtq'(11).□
ℹ	opcacta	eaagt.debug	Trace	QM: check header. header = 36, 36, 36, 0□
ℹ	opcacta	eaagt.debug	Trace	QM: Queue empty.□
ℹ	opcacta	eaagt.debug	Trace	MUX: Mutex mine returns '1' for 'actagtq'
ℹ	opcacta	eaagt.debug	Trace	QM: Unlocking queue: /var/opt/OV/tmp/OpC/actagtq. Mine: 1□
ℹ	opcacta	eaagt.debug	Trace	MUX: Unlocking mutex 'actagtq' ...□
ℹ	opcacta	eaagt.debug	Trace	QM: Unlocked queue: /var/opt/OV/tmp/OpC/actagtq.□
ℹ	opcmsga	TRC	Init	>>>>>>>>>>>>>>>> Tracing Started: Wed Jan 26 13:08:50 2005
ℹ	opcmsga	eaagt.debug	Trace	MSGA: Checking queue status of '/var/opt/OV/tmp/OpC/msgagtq' (Err.sent: 0) ...□
ℹ	opcmsga	eaagt.debug	Trace	QM: Locking queue: /var/opt/OV/tmp/OpC/msgagtq.□
ℹ	opcmsga	eaagt.debug	Trace	MUX: Locking mutex 'msgagtq' (wait == 1) ...□
ℹ	opcmsga	eaagt.debug	Trace	MUX: Locked (1).□
ℹ	opcmsga	eaagt.debug	Trace	QM: lread_from_pos. pos = 0. len = 36. read 36 bytes.□
ℹ	opcmsga	eaagt.debug	Trace	MUX: Mutex mine returns '1' for 'msgagtq'
ℹ	opcmsga	eaagt.debug	Trace	QM: Unlocking queue: /var/opt/OV/tmp/OpC/msgagtq. Mine: 1□
ℹ	opcmsga	eaagt.debug	Trace	MUX: Unlocking mutex 'msgagtq' ...□
ℹ	opcmsga	eaagt.debug	Trace	QM: Unlocked queue: /var/opt/OV/tmp/OpC/msgagtq.□
ℹ	opcmsga	eaagt.debug	Trace	QM: Got status of Q '/var/opt/OV/tmp/OpC/msgagtq': 0. Rc: 0.□

7. On Windows operating systems, you can save the trace configuration into a file, if required, to match the tracing configuration selected. Locate the trace configuration file in a directory that can be accessed by either the application or the configuration client. If using the static tracing model, specify the trace configuration file name using

the TRACE_CONFIG_FILE environment variable. If using the dynamic tracing model, the trace configuration file can be located locally or remotely. This is described in the section titled “Sample Trace Configuration File”.

Sample Trace Configuration File

This sample trace configuration file enables tracing on two applications, opcmsgrb and opcmsgm. The sink is configured as a socket with the system supnode1 as the target server. The components selected are opc.msg, opc.act, and opc.int. The associated category selected is Trace. The tracing attributes are set to the Support defaults of Info, Warn, and Error for all, with the Verbose attribute set on the opc.msg components.

```
TCF Version 3.2
APP: "opcmsgrb"
SINK: Socket "supnode1" "node=161.114.22.105;"
TRACE: "opc.actn" "Trace" Info Warn Error
TRACE: "opc.int" "Trace" Info Warn Error
TRACE: "opc.msg" "Trace" Info Warn Error Verbose
APP: "opcmsgm"
SINK: Socket "supnode1" "node=161.114.22.105;"
TRACE: "opc.actn" "Trace" Info Warn Error
TRACE: "opc.msg" "Trace" Info Warn Error Verbose
```

TIP The ovtrcgui tool makes it easier to create a trace configuration file.

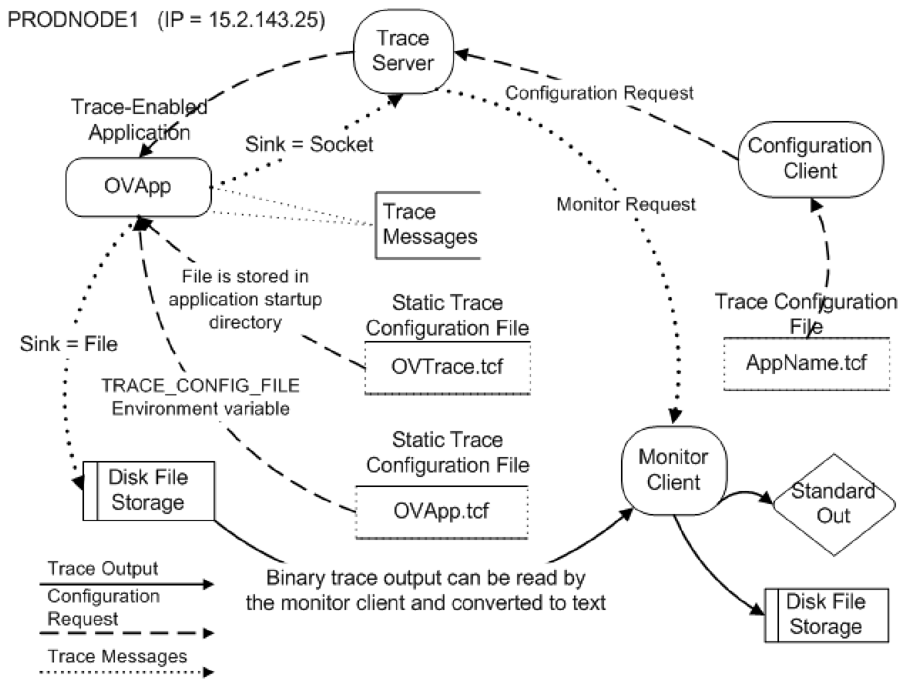
8. Depending on the trace configuration selected, enable either static or dynamic tracing. To find out how to enable tracing, see the following procedures:

   - “Enabling Static Tracing” on page 124.
   - “Enabling Dynamic Tracing” on page 127.

Local Tracing Options

Figure 5-5 shows the static and dynamic tracing configurations available when limiting tracing to one particular node (not using any of the remote tracing capabilities).

Figure 5-5 Local Tracing Options


PRODNODE1 (IP = 15.2.143.25)
Trace Server
Configuration Request
Trace-Enabled Application
OVApp
Sink = Socket
Configuration Client
Trace Messages
Monitor Request
File is stored in application startup directory
Static Trace Configuration File
OVTrace.tcf
Trace Configuration File
AppName.tcf
Sink = File
TRACE_CONFIG_FILE Environment variable
Static Trace Configuration File
OVApp.tcf
Monitor Client
Standard Out
Disk File Storage
Trace Output
Configuration Request
Trace Messages
Binary trace output can be read by the monitor client and converted to text
Disk File Storage

Static Tracing

Figure 5-5 also illustrates two ways static tracing can be configured to start when the `OVApp` starts:

- Create a trace configuration file named `OVTrace.tcf` and place this configuration file in the application startup directory.

Create a trace configuration file named `OVApp.tcf` (or specify an alternative file name) and define the `TRACE_CONFIG_FILE` environment variable to reference this file.

The configuration file is read by the `OVApp` process during startup and will enable tracing within the `OVApp` application.

- The trace messages can be output directly to a file, using the Sink to File configuration option, or it can be output to a trace server, using the Sink to Socket configuration option.
- Use `Ovtrcgui` or `ovtrcmon` and select the `Sink to Socket` option to monitor the trace messages directly from the trace server.

  Alternatively, the client can read the binary trace output file created by the `OVApp` application, if the `Sink to File` option is selected.

Dynamic Tracing

- Figure 5-5 illustrates that tracing can be configured using the configuration client tool `(ovtrcgui` or `ovtrccfg)`, which reads a trace configuration file named `AppName.tcf`.
- This trace configuration request is sent through the local trace server to the trace-enabled application.
- The trace-enabled application sends the trace messages to the local trace server. This assumes that the sink is configured as `Socket` to the local system.
- The monitor client `(ovtrcgui` or `ovtrcmon)`is used to monitor the trace messages, and can be configured to send the trace messages to standard out or to a file on the disk.

Hints and Tips

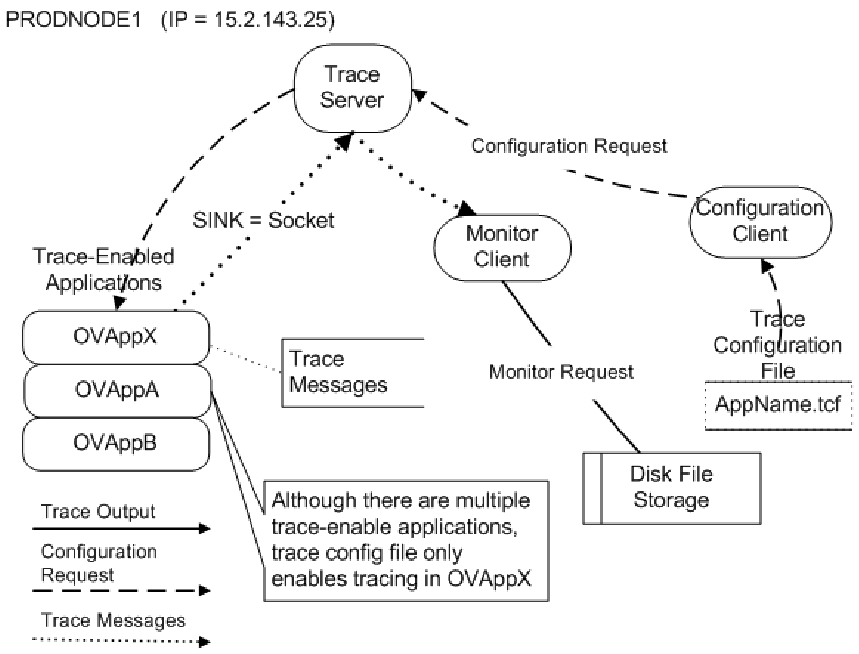
- There may be multiple trace-enabled applications and other non trace-enabled applications running on the same system.
- Many trace-enabled applications can be specified in a trace configuration file.
- There should be only one trace server process running on the system.

Local Dynamic Tracing Options

This section describes the following tracing scenario:

- Dynamic tracing is enabled on `OVAppX` using the local trace server process as the `Sink` and using the trace configuration file named `AppName.tcf`.
- Tracing is monitored using the monitor client (`ovtrcgui` or `ovtrcmon`).
- Trace messages are output in binary format to the file named `OVAppX.trc`.

Figure 5-6 **Local Dynamic Tracing Options**



AppName.tcf File Contents

```
TCF Version 3.2
APP: "OVAppX"
SINK: Socket "PRODNODE1" "node=15.2.143.25;"
TRACE: "Comp1-Name" "Parms" Error Info Warn
TRACE: "Comp2-Name" "Init" Info Verbose
```

Establishing Local Dynamic Tracing

Figure 5-6 illustrates how local dynamic tracing can be configured.

To configure dynamic tracing, follow these steps:

1. Verify that the local trace server process is running.
2. Verify that the `OVAppX` process is running, and that it was started after the trace server.
3. Create the `AppName.tcf` trace configuration file as shown in "AppName.tcf File Contents" on page 120. Substitute the actual `Application`, `Component`, and `Category` names.
4. Make a trace configuration request using the configuration client `(ovtrcgui` or `ovtrccfg).`

NOTE If you are using the `ovtrccfg` tools and the local trace server, the `-server` option is not required.

5. Make a trace monitor request using the monitor client `(ovtrcgui` or `ovtrcmon).`
6. Execute application-specific commands to duplicate the problem or situation for which you need tracing information. When these actions are completed, tracing can be disabled.
7. Disable tracing.

   The application will continue to run, only the tracing will stop.
8. Stop the monitoring of trace messages.
9. Examine the trace output to identify the cause of the problem or send the trace output files to HP Software Support Online for evaluation.

Local Static Tracing Options

This section describes the following tracing scenario:

- Static tracing is enabled on `OVAppX` to capture startup trace messages by placing a trace configuration file named `OVTrace.tcf` in the application startup directory or in:

Windows

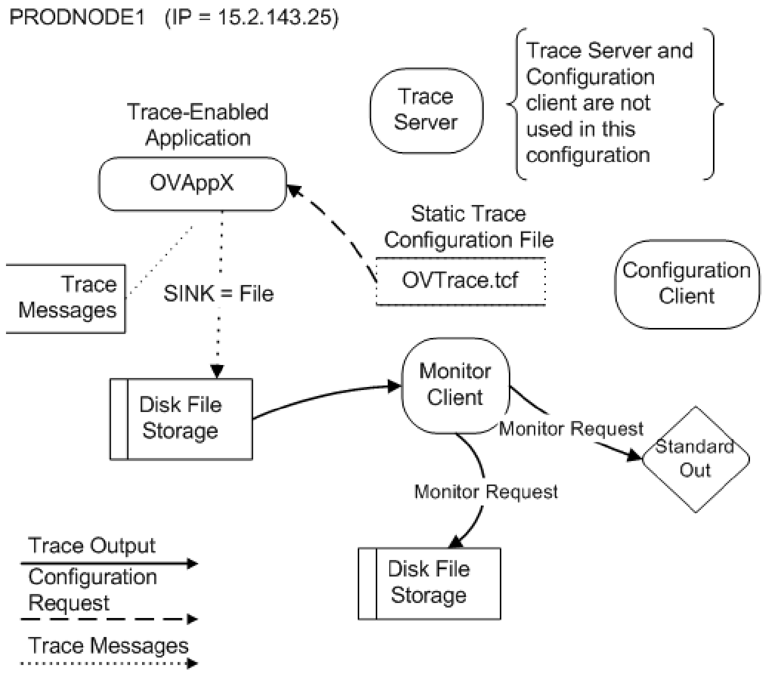
<*install_dir*>\HP OpenView\data\datafiles\xpl\

UNIX

<*data_dir*>/datafiles/xpl/

- Trace messages are output in binary format to the file named `OVAppX.trc`
- Trace output is monitored using the monitor client (`ovtrcgui` or `ovtrcmon`) which reads the binary trace output file created. The trace output is converted to text, written to a text file, and viewed on the standard out device.

Figure 5-7 Local Static Tracing Options



AppName.tcf File Contents

```
TCF Version 3.2
APP: "OVAppX"
SINK: File "/var/opt/OV/share/log/OVAppX.trc"
```

```
"force=1;maxfiles=10;maxsize=100;"
TRACE: "Comp1-Name" "Parms" Error Info Warn
TRACE: "Comp2-Name" "Init" Info Verbose
```

Establishing Local Static Tracing

Figure 5-7 illustrates how local static tracing can be configured. The steps you must execute are as follows:

1. Create the `OVTrace.tcf` trace configuration file as shown above and place the file in application startup directory or in:

   Windows

   `<install_dir>\HP OpenView\data\datafiles\xpl\`

   UNIX

   `<data_dir>/datafiles/xpl/`

   Substitute the actual `Application`, `Component`, and `Category` names.

2. Stop the `OVAppX` process if it is running, using the appropriate application commands.
3. Restart the `OVAppX` process.
4. Execute application-specific commands to duplicate the problem or situation for which you need tracing information. When these actions are completed, tracing can be disabled.
5. Stop the `OVAppX` process.
6. Make a trace monitor request using the monitor client `(ovtrcgui` or `ovtrcmon)` to read from the trace output file, convert the trace messages to text, and redirect the output to a text file or directly to standard output.
7. Examine the trace output to identify the cause of the problem or to send the trace output files to HP Software Support Online for evaluation.

Enabling Static Tracing

To enable static tracing, complete the following steps:

1. Verify that the trace server is running.

   If the sink configuration is defined to a socket, the trace server must be started and running before the application is started. Tracing fails if the application is started before starting the trace server. If the sink configuration is defined to a file, the trace server is not required to be running.

   Windows

   Open the Services dialog and verify that the state of the service named `HP OpenView Shared Trace Service` is `Started`.

   UNIX

   Execute the following command to verify that the trace server process is running:

   ```
   ps -ef | grep ovtrcd
   ```

   The information returned should be similar to the following:

   ```
   root@ supnode1: ps -ef | grep ovtrcd

   root 18750     1  0  Mar  5  ? 0:00
   /opt/OV/lbin/xpl/trc/ovtrcd
   ```

2. If using a configuration file named `OVTrace.tcf`, verify that the file is located in the application startup directory or in:

   Windows

   <*data_dir*>/datafiles/xpl/

   UNIX

   <*install_dir*>\HP OpenView\data\datafiles\xpl\

   Another option is using the `TRACE_CONFIG_FILE` environment variable to specify the name and location of the trace configuration file.

3. Stop the target trace-enabled applications if it is running, using the required application specific commands.

4. Make a trace monitor request using either `ovtrcmon` or `ovtrcgui` if the sink was set to `Socket`.

   Windows

   By default, a Trace Monitor window is displayed after the configuration is completed. To open a new Trace Monitor window, follow these steps:

   a. Start the `ovtrcgui` tool.

   b. From the **File** menu, click **New**, then select **Trace Monitor**.

   c. Select the system for which you want to monitor trace messages and then click **OK**.

   This starts a new trace monitor window.

   UNIX

   There are a number of options available with the `ovtrcmon` command. Refer to the `ovtrcmon` manpage documentation for the complete option details.

   To monitor trace messages, execute one of the following commands or a similar command using additional `ovtrcmon` command options:

   To monitor trace messages from `supnode1` and output traces to a file in binary format:

   ```
   ovtrcmon -server supnode1 -tofile
   /tmp/traceout.trc
   ```

   To monitor trace messages from `supnode1`, display verbose format, output directly to standard out:

   ```
   ovtrcmon -server supnode1 -verbose
   ```

   To monitor trace messages from `supnode1`, display short format, and redirect standard out to a file:

   ```
   ovtrcmon -server supnode1 -short >
   /tmp/traces.trc
   ```

NOTE

The `ovtrcgui` tool cannot be used to dynamically modify the configuration because a static configuration was used.

If the Sink was set to `File`, a monitor client is not required.

5. Start the target trace-enabled applications using the required application specific commands.
6. Execute the application-specific commands necessary to duplicate the problem that you want to trace. When the desired behavior has been duplicated, tracing can be stopped.
7. Stop the target trace-enabled applications using the required application-specific commands.
8. If you are using `ovtrcgui`, the trace messages can be filtered on multiple parameters including components, categories, and severity. You can also save the output to a file.
9. Collect the trace configuration file and the trace output files. Evaluate the trace messages or package the files for transfer to HP Software Support Online for evaluation. If the trace output was written directly to a file, there may be multiple versions of the trace output files. The Sink to File configuration option `Maxfiles` allows for multiple trace output files. These files have an extension of `.001` - `.100` added to the file name.
10. Reconfigure the application to disable tracing before the application is restarted. If using the `OVTrace.tcf` file, the configuration file should be removed or renamed to prevent the application from reading the trace configuration file the next time the application starts. If using the `TRACE_CONFIG_FILE` environment variable, the value should be disabled or the specified configuration file should be removed or renamed.

CAUTION Tracing will restart the next time the application starts unless tracing is disabled.

Enabling Dynamic Tracing

Dynamic tracing has a variety of implementation options including, local or remote configuration and local or remote monitoring. The procedure outlined below covers the general sequence of steps required to enable dynamic tracing. However, they make no attempt to cover all the possible configuration combinations. You must know which system, either local or remote is to be used to execute the commands.

1. Place the trace configuration file in a location to which the client configuration tool has access. The location can be a local or network directory.
   - The trace configuration file should not be named `OVTrace.tcf`, because this is a reserved file name.
   - The `TRACE_CONFIG_FILE` environment variable must not be defined.
2. Verify that the trace server is running.

   The trace server must be started and running before the application is started. If the application is started before the trace server is started, tracing is not possible.

   Windows Open the **Services** dialog and verify that the state of the service named `HP OpenView Shared Trace Service` is set to `Started`.

   UNIX Execute the following command to verify that the trace server process is running:

   ```
   ps -ef | grep ovtrcd
   ```

   The information returned should be similar to the following:

   ```
   root@ supnode1: ps -ef | grep ovtrcd

   root 18750     1  0  Mar  5  ?        0:00
   /opt/OV/lbin/xpl/trc/ovtrcd
   ```
3. Verify the targeted trace-enabled applications are running and that they were started after the trace server was started. If the trace server was started after the trace-enabled applications, tracing will not be possible.

4. Make a trace configuration request using either `ovtrccfg` or `ovtrcgui`.

Windows

Start a new Trace Wizard and select the option to load a configuration file using the following steps.

a. Start the `ovtrcgui` tool.

b. From **File** menu, click **Trace Wizard**, then select **Next**.

c. Select the **Configure local applications by loading a saved configuration** option.

d. From the **Open** dialog, locate and select the trace configuration file.

This starts new tracing windows with the configuration setting from the selected trace configuration file.

UNIX

Execute the following command:

```
ovtrccfg -server <server_name>
<trace_config_file_name>
```

5. Make a trace monitor request using either `ovtrcmon` or `ovtrcgui`.

UNIX

There are a number of options available with the `ovtrcmon` command. Refer to the `ovtrcmon` manpage documentation for the complete option details.

To monitor trace messages, execute one of the following commands or a similar command using additional `ovtrcmon` command options:

To monitor trace messages from `supnode1` and output traces to a file in binary format:

```
ovtrcmon -server supnode1-tofile
/tmp/traceout.trc
```

To monitor trace messages from `supnode1`, display verbose format, output directly to standard out:

```
ovtrcmon -server supnode1 -verbose
```

To monitor trace messages from `supnode1`, display short format, and redirect standard out to a file:

```
ovtrcmon -server supnode1 -short >
/tmp/traces.trc
```

Windows By default, a Trace Monitor window is displayed after the configuration is completed. To open a new Trace Monitor window, use the following steps.

a. Start the `ovtrcgui` tool.

b. From **File** menu, click **New**, then select **Trace Monitor**.

c. Select the system you want to monitor trace messages and then click **OK**.

This starts a new trace monitor window.

6. Execute the application specific commands necessary to duplicate the problem that you want to trace. When the desired behavior has been duplicated, tracing can be stopped.

7. Stop or disable tracing using either `ovtrccfg` or `ovtrcgui`.

UNIX Execute the following command:

```
ovtrccfg -server <server_name> off
```

Windows Stop tracing using the following steps:

- Select the configuration window associated with the tracing.
- From the **File** menu, click **Close**.

8. If you are using `ovtrcgui`, the trace messages can be filtered on multiple parameters including components, categories, and severity. You can also save the output to a file.

9. Collect the trace configuration file and the trace output files. Evaluate the trace messages or package the files for transfer to HP Software Support Online for evaluation. If the trace output was written directly to a file (`Sink to File`), there may be multiple versions of the trace output files. The `Sink to File` configuration option `Maxfiles` allows for multiple trace output files. These files have an extension of `.001` - `.100` added to the file name.

Tracing Guidelines

Keep the following points in mind when configuring and using tracing:

- The effect of HP OpenView Tracing on system performance is dependant on the trace configuration in use. Performance will degrade if the amount of trace information to be collected is large. Therefore, it is important to enable appropriate and optimum number of applications, components, and categories for tracing.
- Always use `ovtrccfg` or `ovtrcgui` tool to disable tracing.
- Use the `ovtrcadm` tool to shut down the trace server.
- Save and reuse a trace configuration file that identifies a specific problem.

Known Limitations

Keep the following points in mind when configuring and using tracing:

- Time synchronization of trace events across systems with different clock settings is not supported.
- Trace server records traces in the order in which the trace messages arrived at the server. It does not synchronize the trace message time nor preserve the time the trace message arrived at the server. This may be a concern if trace messages from multiple remote systems are monitored. Use the Universal Coordinated Time (UTC) option in these cases. Both `ovtrcgui` and `ovtrcmon` tools provide the UTC option.

6 Tracing Processes

This chapter gives a basic guide to tracing for Service Desk processes.

Tracing Processes in Service Desk

XPL tracing provides Service Desk users with a tool to record individual processes and commands. This information enables you to determine the root cause of any errors that Service Desk generates.

See also Chapter 4, “HP OpenView Tracing Fundamentals,” on page 93.

Tracing on UNIX

The following section explains tracing on an HP-UX system. To perform remote logging or use a `.tcf` file for a UNIX system, you must use a Service Desk client installed on a Windows system. HP Support can supply you with a `.tcf` file or you can create one. Use this file to collect XPL data. The following example uses the `OBSStartup.tcf` file. You can also use the steps provided in Tracing on Windows to connect to a UNIX server:

1. Allow the client access to the trace server:

   ```
   ovtrcadm -a <clienthostname>
   ```

   Location: `/opt/OV/support`

   Where *`<clienthostname>`* specifies a localhost or an IP address

   This allows the GUI to connect to the trace server.

   If you are using a Windows GUI, follow Windows instructions and skip the following steps.

2. Transfer the `.tcf` file to the UNIX server using FTP or the Windows GUI to connect remotely to the trace server for real time tracing.

   Make sure that the `.tcf` file is transferred in binary mode, otherwise the file is corrupted.

3. If the following directory does not exist, create it:

   ```
   /var/opt/OV/log/xpl
   ```

4. Start tracing:

   ```
   ovtrccfg -server <servername> OBSStartup.tcf
   ```

   Location: `/opt/OV/support`

5. Start the CORE:

   ```
   ovc -start CORE
   ```

   Location: `/opt/OV/bin`

6. Start the management server:

   ```
   ovc -start OvSlmServer
   ```

   Location: `/opt/OV/bin`

7. Wait for abort or startup then shutdown `ovc`:

   ```
   ovc -stop OvSlmServer
   ```

   ```
   ovc -kill
   ```

   Location: `/opt/OV/bin`

8. Disable tracing:

   ```
   ovtrccfg -server <servername> off
   ```

   Location: `/opt/OV/support`

`OBSStartup.trc` is created in the `/var/opt/OV/log/xpl` directory. Open this file using the trace GUI on windows.

Tracing on Windows

On a Windows server, the trace service is started when the server is started. This service allows the server components to capture trace data. The windows GUI component `ovtrcgui` allows you to configure the `trc` file to capture data on processes that are causing problems.

Perform the following to provide a trace for Service Desk on a Windows system:

1. Allow the client to access trace data. Enter the following:

   ```
   ovtrcadm -a <clienthostname>
   ```

   Location: `C:\Program Files\HP Openview\support\`

   Where *clienthostname* is the localhost or an IP address

   This enables the GUI to connect to the trace server.

2. Run `ovtrcgui` at the command line on the Service Desk client, and select the **Trace Wizard**.

Location: `C:\Program Files\HP Openview\support\`

This GUI can run only on Windows but you can use this program to configure trace files for UNIX systems.

a. In the **Mode** dropdown list, select **New setup for real-time tracing**, click **Next**.

   This step creates a trace file template that you can save and reuse, and use on other systems. You can use local and or remote systems to receive real time tracing data.

b. From **Networking**, select **Local setup only** (if Service Desk trace server and GUI are on the same machine), click **Next**.

c. From **Applications on localhost**, select **OvSlmServer**, and click **Next**.

d. From **Traces for OvSlmServer**, select all components in the list and click **Max** to set the maximum tracing level. Click **Next**. This sets maximum tracing for all components.

e. Click **Finish** to start the logging process.

3. Start the Slm module:

   ```
   ovc -start OvSlmServer
   ```

4. Allow the process to start. If the process you are tracing aborts or encounters a problem, the Tracing Window is displayed. Save the contents of the Tracing Window and send the file to Support. To do this:

   - Close the Tracing Window

     This creates a `.trc` trace file. You can open this file in the windows GUI.

5. Save the trace configurations by doing the following:

   - Close the Configuration Window.

     This creates the `<tracefile>.tcf` file which you can use on other Windows or UNIX systems.

   - When you close the configuration window an information window warns you that there are traces still running in the background. Choose **Yes** to close the active traces.

With trace configuration files you can:

- FTP the `<tracefile>.tcf` to another system and modify it to capture data when the GUI is not available.
- Create `.tcf` files and send them to the customer to use for tracing (for Support Engineers).
- Open `<tracefile>.tcf` in the GUI at any time and start the trace again. You can do this if the GUI is still allowed access to the trace server (see step 1).

7 Configuring Server Processes

This chapter explains how to configure Service Desk management server processes using HP OpenView's control component tools.

HP OpenView Process Control Component

The HP OpenView Control component implements Java Mirror Control using MX4J that implements Java (TM) Management Extensions (JMX) and JMX Remote API (JSR 160). For more information, refer to the following URL:

`http://mx4j.sourceforge.net/`

The ovc Control Tool

`ovc` controls starting and stopping, event notification, and status reporting of all components, including the Service Desk server, registered with the Control service, `ovcd`. A component can be a server process, an agent (for example, the Performance agent or the Discovery agent), an event interceptor, or an application delivered by an integrator.

`ovc` is used from the command line. It has no GUI functionality.

For details of how to use this tool, refer to the ovc(1) man page.

The ovcd Control Daemon

The Control daemon (`ovcd`) is installed as a service/daemon (`HP OpenView Ctrl Service`) and allows the Service Desk server to start automatically at machine startup, provided that the service is set for automatic startup). The Control daemon must be running to use the Control utility.

To start the Control daemon, enter the following command:

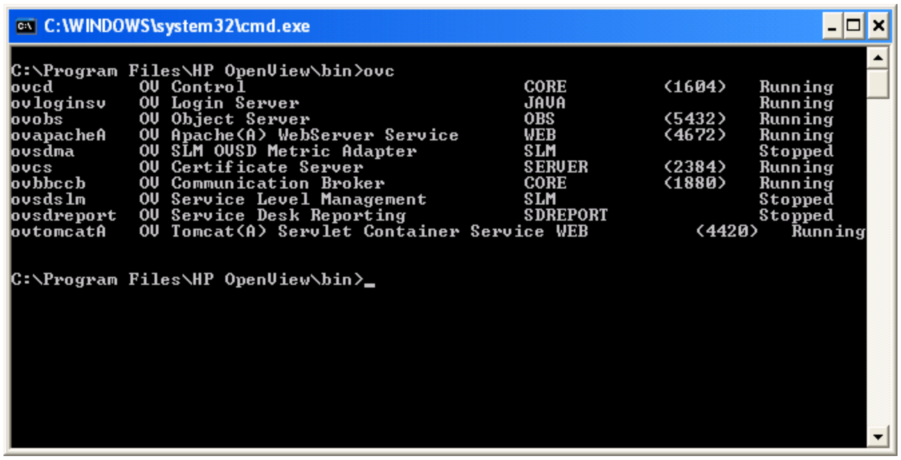
```
ovc -start ovcd
```

If the Control daemon is not running, the following error message appears:

```
(ctrl-111) Ovcd is not yet started
```

Figure 7-1 shows the status output from the `ovc` command when used without parameters.

Figure 7-1 Ovc Command Output



```
C:\Program Files\HP OpenView\bin>ovc
ovcd         OV Control                          CORE        (1604)   Running
ovloginsv    OV Login Server                     JAVA                 Running
ovobs        OV Object Server                    OBS         (5432)   Running
ovapacheA    OV Apache(A) WebServer Service      WEB         (4672)   Running
ovsdma       OV SLM OVSD Metric Adapter          SLM                  Stopped
ovcs         OV Certificate Server               SERVER      (2384)   Running
ovbbccb      OV Communication Broker             CORE        (1880)   Running
ovsdslm      OV Service Level Management         SLM                  Stopped
ovsdreport   OV Service Desk Reporting           SDREPORT             Stopped
ovtomcatA    OV Tomcat(A) Servlet Container Service WEB         (4420)   Running

C:\Program Files\HP OpenView\bin>_
```

From left to right, the columns display the process name, description, group, ID, and status of all registered processes.

The process name is the abbreviation used by the Control component to identify the process. The process group is an abbreviation for processes that are related to each other. The process ID is the process number given by the operating system (PID in the Windows Task Manager). The process status can be:

`Stopped`	Process is not running.
`Initializing`	Process is starting.
`Running`	Process is running.
`Aborted`	Process was started but startup failed, or process stopped after encountering an error.

Use the `ovc` command to start, stop, or restart a process. You can do this by process name or process group. For example, to start the Object Server process `ovobs`, you can use any of the following commands:

`ovc -start OBS`

`ovc -start ovobsag`

`ovc -start ovobs`

Control Configuration

The Control component uses an internal repository, the XML-files in the directory `%OvDataDir%conf/obs/ctrl`, to store information about processes, their dependencies and how to start, stop, and query processes. You can manipulate this repository using the `ovcreg` utility.

NOTE Do not manually edit the files in:

`%OvDataDir%conf/obs/ctrl`

Use the `ovcreg` utility.

To change the configuration for a process:

1. Stop the process by entering the appropriate command at the command line.

   For example, for `ovobs`, enter the following command:

   ```
   ovc -stop ovobs
   ```

2. Remove the current configuration of the process from the repository. For example, for `ovobs` enter the following command:

   ```
   ovcreg -del ovobs
   ```

3. In the `%OvDataDir%conf` directory and its subdirectories, search for every XML file that contains the abbreviation of the component you want to change, for example `obs`, `slm`, or `report`. Here, `%OvDataDir%` refers to `C:\Program Files\HP OpenView\data`.

4. Edit the `<ovc:Start>` tag in each file found in Step 3. Then register the process using `ovcreg`. For example, for `ovobs` enter the following at the command line:

   ```
   ovcreg -add %OvDataDir%conf/obs/OvObs.xml
   ```

5. After you register the changed configuration, restart the process using the `ovc` command. For example, for `ovobs`, enter the following at the command line.

   ```
   ovc -start ovobs
   ```

Troubleshooting Control Error Messages

This section covers Control error messages and how to troubleshoot them.

(ctrl-7) Error in the Target Component

This error indicates that the Control has failed to start the requested component.

Cause

Failure to start the component because of the following reasons:

- Incorrect path for executable in the `<CommandLine>` tag of the `START` or `INITIALIZE` hooks.
- Timeout while executing any of the actions. Check actions defined in `INITIALIZE` hook.
- Problem in the component itself. For example, the component cannot be started as a stand-alone component, that is, it cannot be started without the Control component.
- Problem specific to HP-UX. The `<CommandLine>` tag value specified after `<ProcessDescription>` is not found or is not present within the first 60 characters of the command string used to start the process.
- Problem specific to Windows. Exceedingly long PATH value in the system-wide PATH environment variable.

Solution

- Check whether the requested process, in the `ABORTED` or `STOPPED` state, is running:

Windows	Task Manager
UNIX	**`ps -ef`**

  If the process is running, check whether the registration file contains the `<CommandLine>` tag after `<ProcessDescription>`. If this tag is present, ensure that the value of `<CommandLine>` is a substring of the actual command string used to start the component.

- Run the command string `<CommandLine>` of the `START` hook manually and check the process entry in the process table. The `<ProcessDescription>` specified in the registration file should match the one in the process table. If the entries do not match, change the `<ProcessDescription>` value, re-register the component, and restart it.
- If the process name is not in the process table, there is a problem with the component itself. If the process name is present, check the `<CommandLine>` after `<ProcessDescription>` in the same way.
- If the component has an `INITIALIZE` hook, ensure that the `START` hook is correct and execute the command string specified in the `INITIALIZE` hook. This command should return 0 on exit.
- By default, Control allows 60 seconds for the [execute] actions specified in the `INITIALIZE` hook to complete. If the action does not complete within 60 seconds, Control forces the component to stop. You can change this interval using the following command:

  ```
  ovconfchg -ns ctrl.ovcd -set ACTION_TIMEOUT 120
  ```

  This problem can occur when tracing is enabled.
- On HP-UX, make sure that `<CommandLine>` appears within the first 60 characters of the command string used to start the process. This is required for Java processes, where the process description is common for all Java applications. To do this, pass a unique name as a system property to JVM (using the `-D` option). Use the unique name in the `<CommandLine>` tag.
- In windows, if the environment variable PATH contains many entries, Control can fail to start the components correctly. This occurs when the number of characters exceeds 651 in Control 01.50.141. This limit is increased to 1239 characters in Control 01.50.150 and above.

Security Error when Executing the <Start|Stop|Status|Notify> Method

This error indicates a failure in authentication or authorization.

Cause

Problems in SSL handshaking:

- CERT_INSTALLED configuration setting is set to TRUE when certificates are not installed.
- Certificates are installed but they are not valid.
- The node does not have the right to perform the operation, `ovrc` only.

Solution

- Ensure that CERT_INSTALLED is set to FALSE when certificates are not installed.
- Ensure that valid certificates are installed.

Communication Error when Executing the <Start|Stop Notify|Status> method

Cause

SoapException, RpcException, and IOException (failed to create the socket, connection establishment failure).

Solution

This error does not usually occur with `ovc`. If it does, execute again to resolve the error.

(ctrl-4) No Component Matches Target

Cause

The requested target is not registered with Control.

Solution

Register the requested target and run the command again (see “Control Configuration” on page 142).

No Target Component Subscribed for this Event

Cause

When using `ovc -notify <EVENT_NAME> <target>`, none of the registered components `[<target>]` are subscribed for the specified event.

Solution

Make sure that the target is registered for the EVENT_NAME in its registration file, (OnEvent hook).

Error in One of the Target Prerequisites

Cause

Control has failed to start the component because one or more prerequisites or dependencies for that component have failed.

Solution

Check the registration file and identify the dependent component or components. Try to start these dependent components independently using the command:

```
ovc -start <Dependent Component>
```

If this results in an error in the target component, follow the steps specified for “(ctrl-7) Error in the Target Component” on page 143.

Control can also give this error if the dependent component is in the RUNNING state. `ovc -start` can also generate this error when all registered components are RUNNING. This can happen when more than one component has the same dependency and the dependent component has an INITIALIZE hook that fails. See the following examples.

Example 1:

Components A, B and C. A and B are dependant on C. Control randomly selects a component to start; for example, component A. Control checks that A is dependent on C, and tries to start C. C fails because of an INITIALIZE hook timeout. Control fails to start A because it cannot start the target prerequisite C. Control tries to start B, checks that the dependent component C is not RUNNING, and attempts to start C again. C starts this time, which results in B starting. Because C is already RUNNING, it cannot start C again and because Control has failed to start component A it gives the error:

```
Error in one of the target prerequisites.
```

Even though the dependent component C is running.

Example 2:

Component A is dependent on B and B is dependent on C.

Control attempts to start C first but fails because of an INITIALIZE hook timeout. Control stores the error message as:

```
Error in the target component
```

Control attempts to start B and finds that B is dependent on C. Control attempts to start C again, but fails for the same reason. Control stores the error as:

Error in one of the target prerequisites.

ovc -start results in multiple errors but Control can generate one error only. Control attempts to start the third component A, notes that it depends on B which in turn depends on C. Control again attempts to start C, and this time INITIALIZE hook succeeds. Control starts C, B and A but reports the last saved error message:

```
Error in one of the target prerequisites.
```

Not Authorized

See “Security Error when Executing the <Start|Stop|Status|Notify> Method” on page 144.

Not Authenticated

See “Security Error when Executing the <Start|Stop|Status|Notify> Method” on page 144.

Control Daemon is Running but Cannot be Accessed Remotely

Cause

This message indicates that Control is running, but its service cannot be accessed from a remote system. This is reported when `ovbbccb` is in ABORTED state or when other processes like Control are not able to connect to `ovbbccb` (it is in RUNNING state). In this case, communication cannot be established between the agent and other machines.

Solution

Restart Control.

Control Daemon is Being Initialized

Cause

This error is displayed when Control fails to create its RpcServer or hangs when creating the RpcServer.

Solution

Kill and restart the `ovcd` process. On UNIX, remove the file `<$DataDir>/tmp/ovcd.sock` before restarting Control.

Error. Check Log File for More Information

Cause

The error information is logged in the log file.

Solution

See both System.txt and any component specific log files to analyze the problem.

Using the Config Component

The Config component has an internal repository for component settings. The Config component manages port numbers, intervals, retry settings for components that are used by Service Desk. You can view the contents of the Config repository by running:

```
ovconfget
```

Use the `-h` parameter to view the help for this utility.

To edit the repository, enter:

```
ovconfchg -edit
```

This opens `hp.XplConfig.ovconfchg` for editing.

To stop the affected processes, use `ovc`.

If you changed settings for Apache or Tomcat, you can verify the changed settings by running the following commands from the installation `bin` directory:

Apache on Windows

```
cscript ovapachectl.vbs -getconf
```

Apache on UNIX

```
ovapachectl -getconf
```

Tomcat on Windows

```
cscript ovtomcatctl.vbs -getconf
```

Tomcat on UNIX

```
ovtomcatctl -getconf
```

Start the changed process using the `ovc` command.

Using Certificates

Service Desk uses certificates to authenticate anything that wants to communicate with the server (also for server to server). Certificates are managed by one certificate server only. The certificate server is installed with the primary server installation. Using `ovc` this is displayed as follows:

```
ovcs    OV Certificate Server SERVER    (4780)    Running
```

Service Desk clients are installed with a trusted certificate by default. There are situations where a trusted certificate must be added manually to another Service Desk environment. For examples, see the HP OpenView Service Desk online help.

The following applies when changing certificates:

- `ovcs`: The Certificate Server. This process can run on one server only (the primary server). This process validates certificates.
- `ovcoreid`: A command-line utility that returns the Core ID of the Service Desk installation. Each Service Desk installation has a unique Core ID. You can use the Core ID to specify the unique ID of a certificate to be used with `ovcm`.
- `ovcm`: A command-line utility to manage certificates on the certificate (primary) server.
- `ovcert`: A command-line utility to manipulate registered certificates. For example, on a client or a secondary server.

Tracing

To start tracing, follow these steps:

1. Check that `HP OpenView Shared Trace Service` is running (see Services in Windows).
2. Go to the *<OVINSTALLDIR>*`\support` directory and run the following command:

   ```
   ovtrcadm -a localhost
   ```

3. Launch `ovtrcgui.exe` and follow the steps to configure components for tracing. See also “Tracing Processes in Service Desk” on page 134.

8 Testing and Troubleshooting the Service Desk Agent

This chapter describes the following topics related to the Service Desk agent:

- Parameters used with the agent.
- Monitoring the agent.
- Viewing the log file produced by the agent.
- Testing the agent.
- Troubleshooting the agent in the following situations:
    - When the agent does not execute the required actions.
    - When the agent has executed a process but no GUI is displayed.

By default, the agent is managed by `OvCtrl`. See `%OvDataDir%conf/obs/OvObsAg.xml`, the registration file for the agent to be submitted to `OvCtrl` (`ovc -start ovobsag`). When passing parameters to the Service Desk agent, you should add them at the end of the `<CommandLine>` of the `START` hook in the XML-file.

By default the agent uses port 50998.

Parameters:

- **`OvObsAgent`**

  (no parameters used)

  By default, the Time To Live (`TTL`) property is set to `1`. A TTL of 1 means that a server at another side of a router will not hear this agent's multicast alive message.

- **`OvObsAgent 5`**

  TTL set to 5

- **`OvObsAgent`** ***`<servername.domain.com>`***

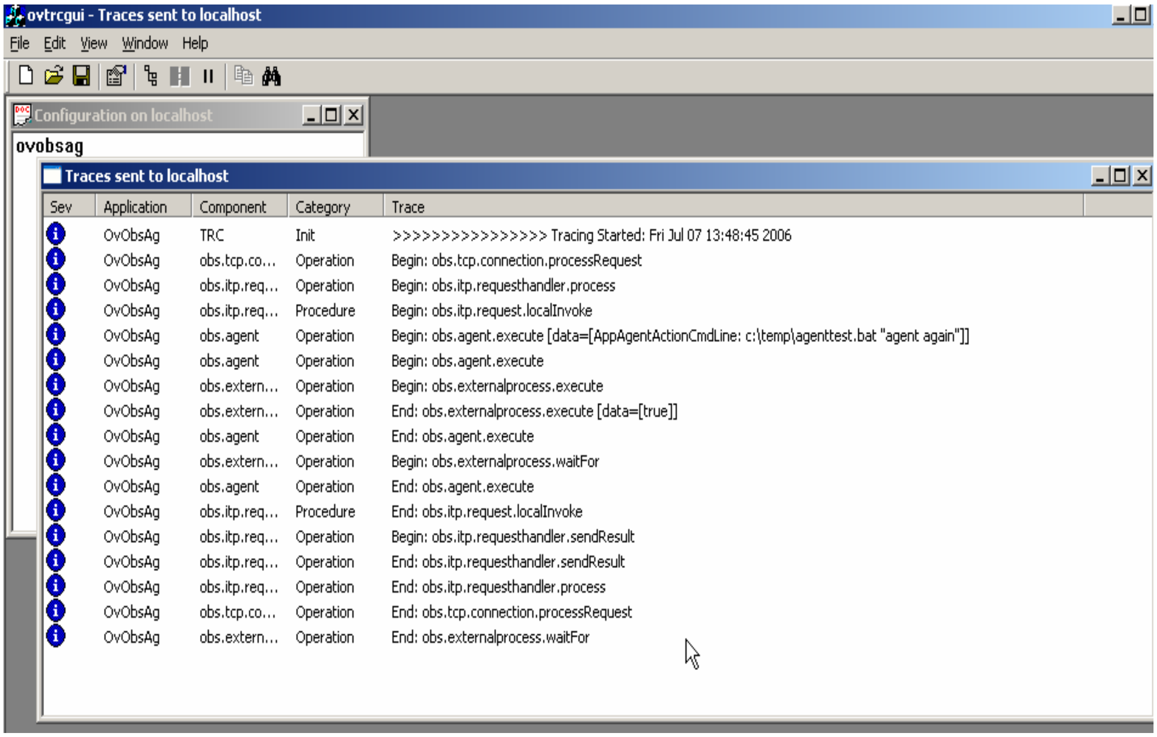
  The agent communicates with the server specified by the hostname.

- **`OvObsAgent`** ***`<10.0.0.1>`***

  The agent communicates with the server specified by the IP address.

To monitor the agent, use the XPL tools as illustrated in Figure 8-1:

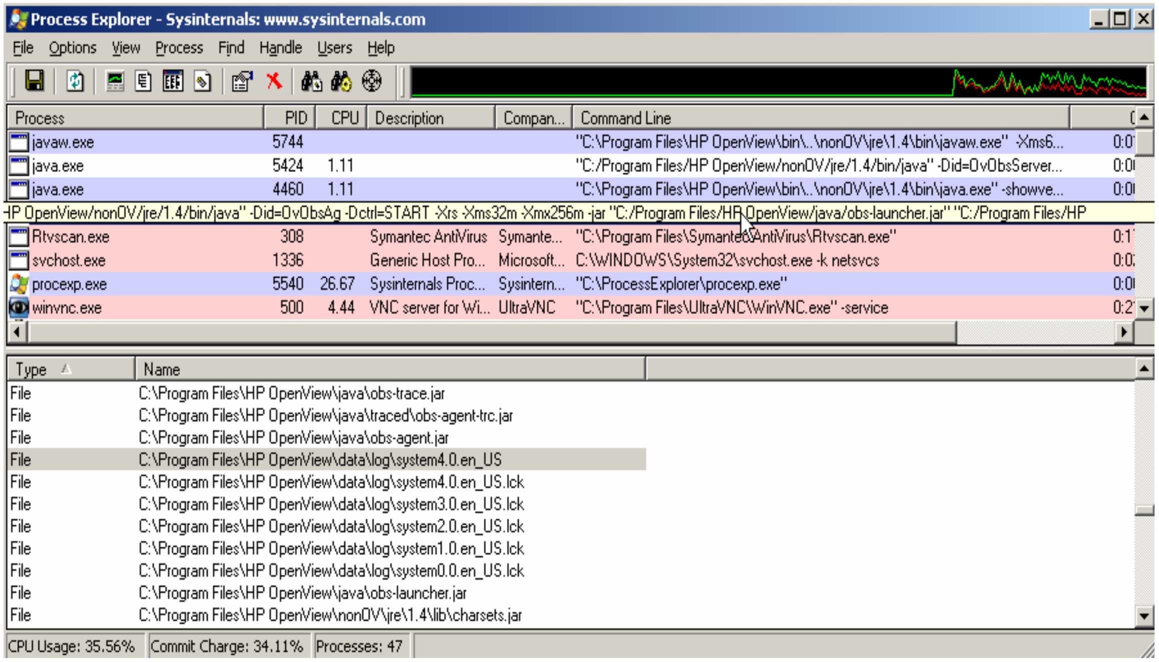
Figure 8-1 Using the XPL Tools to Monitor the Service Desk Agent


ovtrcgui - Traces sent to localhost
File Edit View Window Help
Configuration on localhost
ovobsag
Traces sent to localhost

```
Sev  Application  Component       Category   Trace
     OvObsAg      TRC             Init       >>>>>>>>>>>>>>> Tracing Started: Fri Jul 07 13:48:45 2006
     OvObsAg      obs.tcp.co...   Operation  Begin: obs.tcp.connection.processRequest
     OvObsAg      obs.itp.req...  Operation  Begin: obs.itp.requesthandler.process
     OvObsAg      obs.itp.req...  Procedure  Begin: obs.itp.request.localInvoke
     OvObsAg      obs.agent       Operation  Begin: obs.agent.execute [data=[AppAgentActionCmdLine: c:\temp\agenttest.bat "agent again"]]
     OvObsAg      obs.agent       Operation  Begin: obs.agent.execute
     OvObsAg      obs.extern...   Operation  Begin: obs.externalprocess.execute
     OvObsAg      obs.extern...   Operation  End: obs.externalprocess.execute [data=[true]]
     OvObsAg      obs.agent       Operation  End: obs.agent.execute
     OvObsAg      obs.extern...   Operation  Begin: obs.externalprocess.waitFor
     OvObsAg      obs.agent       Operation  End: obs.agent.execute
     OvObsAg      obs.itp.req...  Procedure  End: obs.itp.request.localInvoke
     OvObsAg      obs.itp.req...  Operation  Begin: obs.itp.requesthandler.sendResult
     OvObsAg      obs.itp.req...  Operation  End: obs.itp.requesthandler.sendResult
     OvObsAg      obs.itp.req...  Operation  End: obs.itp.requesthandler.process
     OvObsAg      obs.tcp.co...   Operation  End: obs.tcp.connection.processRequest
     OvObsAg      obs.extern...   Operation  End: obs.externalprocess.waitFor
```

You can use the Sysinternals Process Explorer (see Chapter 6, “Tracing Processes,” on page 133) to find the log file the agent is using. Figure 8-2 illustrates the log file `system4.0.en_US`.

Figure 8-2 Sysinternals Process Explorer


Process Explorer - Sysinternals: www.sysinternals.com
File Options View Process Find Handle Users Help

Process	PID	CPU	Description	Compan...	Command Line
javaw.exe	5744				"C:\Program Files\HP OpenView\bin\..\nonOV\jre\1.4\bin\javaw.exe" -Xms6...
java.exe	5424	1.11			"C:/Program Files/HP OpenView/nonOV/jre/1.4/bin/java" -Did=OvObsServer...
java.exe	4460	1.11			"C:\Program Files\HP OpenView\bin\..\nonOV\jre\1.4\bin\java.exe" -showve...
Rtvscan.exe	308		Symantec AntiVirus	Symante...	"C:\Program Files\Symantec AntiVirus\Rtvscan.exe"
svchost.exe	1336		Generic Host Pro...	Microsoft...	C:\WINDOWS\System32\svchost.exe -k netsvcs
procexp.exe	5540	26.67	Sysinternals Proc...	Sysintern...	"C:\ProcessExplorer\procexp.exe"
winvnc.exe	500	4.44	VNC server for Wi...	UltraVNC	"C:\Program Files\UltraVNC\WinVNC.exe" -service

```
HP OpenView/nonOV/jre/1.4/bin/java" -Did=OvObsAg -Dctrl=START -Xrs -Xms32m -Xmx256m -jar "C:/Program Files/HP OpenView/java/obs-launcher.jar" "C:/Program Files/HP
```

Type	Name
File	C:\Program Files\HP OpenView\java\obs-trace.jar
File	C:\Program Files\HP OpenView\java\traced\obs-agent-trc.jar
File	C:\Program Files\HP OpenView\java\obs-agent.jar
File	C:\Program Files\HP OpenView\data\log\system4.0.en_US
File	C:\Program Files\HP OpenView\data\log\system4.0.en_US.lck
File	C:\Program Files\HP OpenView\data\log\system3.0.en_US.lck
File	C:\Program Files\HP OpenView\data\log\system2.0.en_US.lck
File	C:\Program Files\HP OpenView\data\log\system1.0.en_US.lck
File	C:\Program Files\HP OpenView\data\log\system0.0.en_US.lck
File	C:\Program Files\HP OpenView\java\obs-launcher.jar
File	C:\Program Files\HP OpenView\nonOV\jre\1.4\lib\charsets.jar

CPU Usage: 35.56% Commit Charge: 34.11% Processes: 47

Testing the SD Agent

To test the Service Desk agent, follow these steps:

1. Install the SD agent on the target machine (in this example named `target.hp.com`).
2. Create a batch-file `agenttest.bat` in `c:\temp` with the following contents:

   ```
   echo agenttest %1 > c:\temp\agenttest.txt
   ```
3. Start the server.
4. Start the agent.
5. Start the client.
6. Create an agent test database rule as follows:

   a. When configuration item is created or modified

      where Name 2 (*) contains agent

      agenttest (Command Exec Action), null, Host: target.hp.com, Command line: `c:\temp\agenttest.bat, Parameters: "[Name 2] "`

      Before saving this rule, select the debug option.

   b. Create or update a configuration item with `Name 2` set to “agent test”. Save the configuration item.

      In the rules log, you should find confirmation that the server is dispatching an action to the agent. For example, in a `rules0.0-log` file:

      ```
      <date>; FINER; Dispatching action. Rule: agenttest,
      Action: agenttest (Command Exec Action) host:
      target.hp.com command: c:\temp\agenttest.bat "agent
      test"
      ```

   c. On the target machine, in the directory `c:\temp` you should find a file called `agenttest.txt` with the following contents:

      ```
      agenttest "agent test"
      ```

SD Agent Does Not Execute Required Actions

The following errors are returned by the Service Desk agent when it fails to execute actions:

```
Error: <date>;com.hp.ov.obs.rules;SEVERE;Rule: agenttest
Action: agenttest (Command Exec Action) Agent: pc.hp.com:
not responding (no agent running?).
```

This error is logged in the server log files:

Rules log files	`rules0.0.en_US`
System log files	`system0.0.en_US`

```
Server Monitor> Queues> "Agent failed job queue" increases
```

Check the agent log file for errors. For example, the following error is logged because of insufficient write access:

```
<date>;com.hp.ov.obs;WARNING;External program returned error
code 1.
Command: "c:\temp\agenttest.bat" "agent testje"
Output: C:\Program Files\HP OpenView\data\tmp>echo agenttest
"agent testje"  1>c:\temp\agenttest.txt
Access is denied.
```

If the agent fails to launch an interactive program see "Agent Does Not Launch a Program with a GUI" on page 160.

Solution

Check the following:

- Is the agent running on the target machine? Use `ovc -status` to check.
- Are there firewall settings that could prevent communication with the agent?
- Does the agent have the correct TTL to access the server?
- Can you execute the action manually?

NOTE The server reports errors only if the agent is unable to execute an action. The server does not report errors when the agent is able to execute an action but the results are not as expected.

Agent Does Not Launch a Program with a GUI

The agent executes an action and the associated process is running, but the GUI does not appear. For example, the agent launches the `Notepad` application and you can see that the process is running, but the `Notepad` application GUI is not displaying.

Solution

An agent, started as a background process, using `ovc -start ovobsag`, cannot interact with the desktop. This is the case for all processes launched in the background, for example, Windows Services and UNIX daemons. Use UI Rules with actions instead.

If you cannot use UI Rules, start the agent at the command line with:

```
OvObsAgent
```

This process is started by the user and not by the system as background process. It can interact with the desktop and therefore display the program GUI.

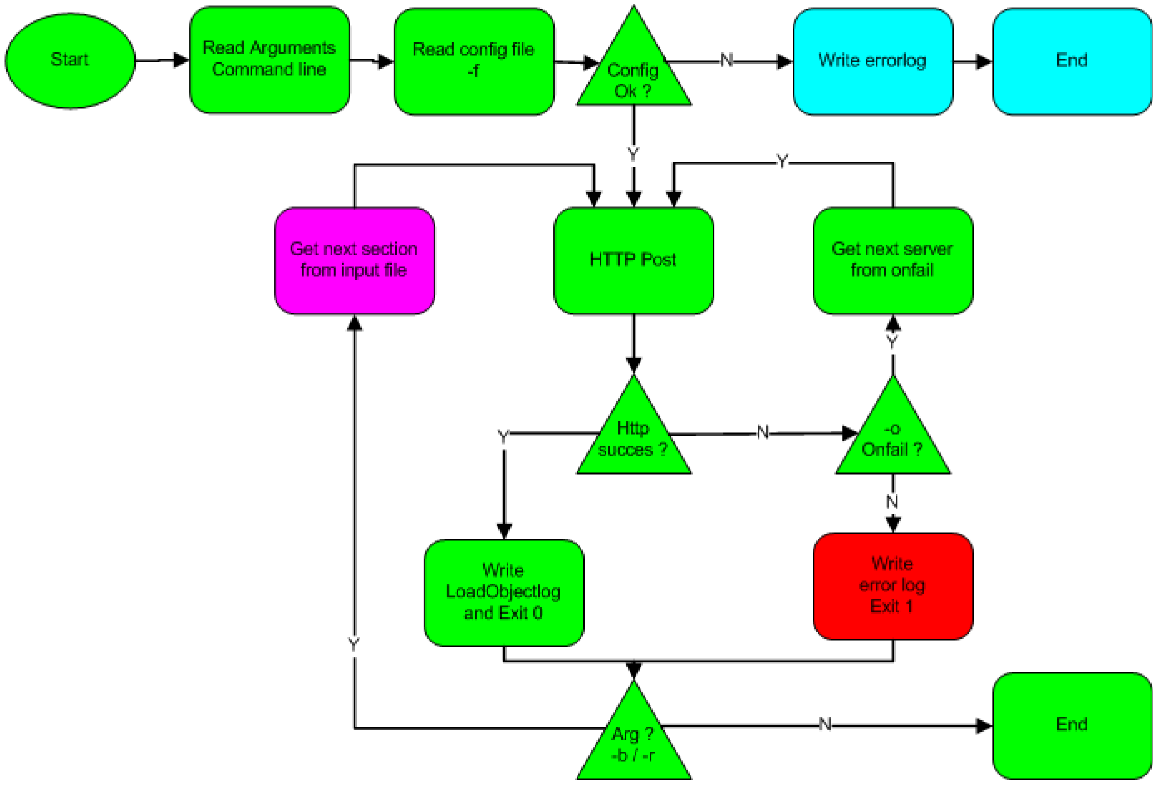
9 Object Loader

This chapter provides support information for Object Loader.

Object Loader Internals

Object Loader is used to create, update, and delete objects in Object Server. In essence, it is a custom HTTP client that connects to the HTTP listener of an Object Server application server. Figure 9-1 shows the program flow of Object Loader on the client side.

Figure 9-1 Object Loader Process Flow



The program flow steps of Object Loader on the client side are describe in the following sections:

- “HTTP Post” on page 163.

- "Import Mappings" on page 164.
- "Value Lists" on page 164.
- "HTTP Response" on page 165.
- "Command-Line Arguments and Configuration File Parameters" on page 165.
- "The ONFAIL Option" on page 166.

HTTP Post

`OvObsLoadObject` sends an `HTTP POST` command to the `evpost` URI on an Object Server application server. The body of this request contains data in the form of *`<key>=<value>`* pairs. A number of parameters are always required in order to make a valid request, as shown in Table 9-1.

Table 9-1 **HTTP Post Parameters**

Parameter	Description
`MAPPING`	Name of an existing import mapping on the application server.
`CLASSNAME`	Name of an object class as defined within the import mapping.
`MODUS`	Type of operation to perform on the object: `INSERT`, `UPDATE` or `DELETE`.
`USERNAME`	The user name of an Object Server user.
`PASSWORD`	Password for the user name specified under `USERNAME`.

Although the `MODUS` parameter has three values listed, the `INSERT` and `UPDATE` settings actually work the same way: they create a new object when the referenced object is not found, and they update an existing object if it is found.

The parameters described in the above table are metadata. In other words, they are not the actual data that you want to send with `OvObsLoadObject`, but rather the data that describes how you intend to use the data.

Metadata alone does not provide Object Loader with any information that allows it to refer to an object uniquely. This makes it impossible for the application server to recognize to which object you are actually referring. Therefore, you also need to specify at least one parameter that identifies the object - its primary key, and at least a minimal set of parameters required for the operation you are requesting. This combination of parameters that is to be translated into an object manipulation is called a value list in Object Loader terms.

The primary key and the set of related required parameters is a result of the import mapping you are using and the object class therein that you are referring to. Therefore, understanding import mappings is key to understanding Object Loader.

Import Mappings

An import mapping is basically a table that translates an external naming convention to the attributes of objects in the Object Server database. It is defined in the HP OpenView Configuration workspaces in the HP OpenView console. Often, attributes have a default value. Attributes can be associated with a template to define these defaults for a specific import mapping. A template is a set of pre-defined values for the attributes of an object.

More than one object type can be defined in an import mapping. However, import mappings are not commonly used that way in combination with Object Loader. The object types in an import mapping correspond to the possible values for `CLASSNAME` in Object Loader.

Value Lists

After you choose an import mapping, the only thing you need to do to manipulate an object in Object Server is specify the primary key of the object, and the values with which you want to update the attribute.

A limiting factor in selecting the keys and values for the value list is the state of the object. For example, when you update the status of a configuration item to set it to `Production`, required field settings defined by the Service Desk administrator in the HP OpenView console may cause the `Located at` attribute of the configuration item object to become mandatory. The workflow layer of Object Server is responsible for this behavior.

HP OpenView Configuration Workspaces → Data → Required Fields

When working from the HP OpenView console, you see feedback on the screen, indicating the activity of the Workflow layer. In our example, you would see a red asterisk appear in the label of the `Located at` field after setting the status to `Production`. Because Object Loader is designed to be used in automated environments, the person that sets it up needs to know how the rules defined on the Workflow layer affect the required attributes of an object.

HTTP Response

As specified in the HTTP 1.0 protocol (RFC 1945), any HTTP request is answered with a response that indicates whether the command was successful (the HTTP listener of Object Server accepts HTTP 1.1 requests as well, but Object Loader uses 1.0).

As the HTTP 1.0 standard prescribes, a message starting with the line `HTTP/1.0 200 OK` indicates the command was successfully processed. Success is loosely defined in this context: the message body may still contain a warning that in fact nothing happened (for example, because there were no changes to save). This means that you have attempted an update of an object with values that were no different from the ones already known to Object Server.

A message starting with `HTTP/1.0 500 Internal Server Error` indicates that the request failed to be processed. In this case, the body of the message contains a more descriptive error message that explains what went wrong.

Command-Line Arguments and Configuration File Parameters

The previous sections describe the data that needs to be sent with an `evpost POST` request to the HTTP listener of the Object Server application server. Because Object Loader is essentially no more than a utility to form such a request, the arguments for `OvObsLoadObject` correlate to the required parameters for the request. `OvObsLoadObject` does not use the names of the request parameters as its arguments. Arguments are specified as is customary on UNIX systems: a hyphen followed by a letter or keyword (or the long option form: two hyphens followed by a long option name). The subsequent argument contains the actual value you want to set, unless the previous argument was a flag. A flag is a boolean argument that is false by default and becomes true whenever it is used at the command line.

The argument -v or --keyval (VALUE_LIST) is handled in a special way. All arguments following -v or --keyval are regarded as the value for this argument, and should contain key/value pairs separated by the equal sign. These key/value pairs make up the actual object data you need to send.

Instead of reading arguments at the command line, you may also specify them in a configuration file. This configuration file needs to be formatted in the same way as Windows .INI files.

In Windows .INI files, the key/value pairs are divided into sections. A section header is a line that contains the section name between square brackets. The section [OVOBSLOADOBJECT] provides defaults for the corresponding command-line arguments.

Other valid section headers take the form of [EVENT_identifier] where identifier is a string that provides a unique key for the set of parameters in that section. This form is used in Object Loader's log files (see "Errors and Logging" on page 172). It allows the log files to be used for the re-sending of events. There are two ways to re-send events from a configuration file: by re-sending all events in the log file (-b option), or by re-sending only those events that have a SEND parameter that is set to false (-r --resend option). The --resend option changes the SEND parameter to true in the configuration file for each event it now succeeds in sending.

When a value list is specified in a configuration file, the various key/value pairs therein are separated by the pound sign (#). Furthermore, special characters, such as new lines and tabs are substituted by escape characters (a backslash followed by a letter) that follow the UNIX convention. The escape characters are a potential source of trouble because character sequences that resemble escape characters are actually interpreted as escape characters unless the text is pre-processed. For example, the character sequence \n in the string DOMAIN1\name1 is unintentionally interpreted as a new line escape sequence unless the text string is pre-processed by adding an additional backslash: DOMAIN1\\name1.

The ONFAIL Option

The ONFAIL parameter and its equivalent, the -o or --onfail at the command line, are often misunderstood. The ONFAIL option is not a load balancing mechanism, nor does it negotiate with application servers to decide which one should handle the request.

ONFAIL is an optional parameter that specifies a list of Object Server application servers to try in case the one specified with the SERVER parameter does not respond. Not responding means not accepting a TCP connection in this context. As soon as a TCP connection is accepted by an application server, Object Loader assumes it can post data even though it might not have connected to a HTTP server but to some other TCP-based server instead. After posting a HTTP request to another type of TCP server, the server usually responds that it does not understand the request and closes the connection.

The order in which the servers appear in the ONFAIL parameter is also the order in which they are contacted by Object Loader until one responds or all have been tried.

Another common misconception is that ONFAIL has to do with retrying events that could not be posted earlier. This is not the case. When an application server responds to a HTTP request with any HTTP response other than an HTTP 200 message, the request is written to the error log of Object Loader without soliciting any other application servers.

Queuing

Queuing tools are available to ensure that events arrive at the application server in the same order in which they occur.

The order of events need to be controlled in situations where many Object Loader calls are made that refer to the same object. The Object Server application server handles requests that are posted with Object Loader asynchronously. Not every event is processed with the same speed.

Situations can arise in which one thread of the application server is still processing an insert request, while another processes an update for the object that is still being inserted. Consequently, the update fails because it looks for an object that is not yet created. The queuing tools make sure that one event is sent at a time in the order in which they were created, thus preventing this sort of problem from arising.

The queuing tools implement a FIFO command execution queue, in which each command is processed as a blocking operation. You can put any shell command in the queue. Its use is not limited to Object Loader commands.

If a command in the queue hangs, it causes the entire queue to hang. This limitation is inherent to its design. The queuing tools have no way to determine what is a reasonable amount of time for a command to spend running, nor any mechanism to detect that a program is no longer responding.

Architecture of the Queuing Tools

The queuing tools use a directory as a queue. An entry in the queue is added by creating a new file starting with the prefix `job`. The rest of the file name is used to indicate the order in which they are to be processed when the files are retrieved in alphabetical order. Each line in each file contains a command to be executed by the queue. The job queue directory should be examined to see if it contains a file called `BLOCKED` before adding a job file to it. If the `BLOCKED` file exists, no jobs may be added until it is removed.

Before the files are de-queued (and consequently, the commands specified therein executed), a file called `RUNNING` is created in the queue directory. The presence of this file indicates that a process is busy executing the commands in the queue. This prevents another process from trying to operate on the same set of files as the current de-queue process and conflicts between the two arising as a consequence. Once de-queuing is complete, the `RUNNING` file is removed. During the de-queue process, each line from each file is executed. When all lines from a file have been executed, it is deleted. This results in the queue directory being emptied completely of all files with prefix `job` when de-queuing is finished.

Implementation of the Queuing Tools

The queuing tools are written in Perl, having the common functions between the scripts in the `LoadObject::OvObsQueue` module and its dependencies.

On Windows and UNIX, the Perl scripts are distributed with all the necessary modules and libraries. The scripts are installed in the `<`*`installation_directory`*`>\lbin\obs\loadobject` folder.

In addition to `OvObsEnqueue` and `OvObsDequeue`, a third program is delivered with the queuing tools: `OvObsCtlQueue`. This program allows the user to block the queue (that is, create the `BLOCKED` file), examine the queue status, and empty the queue without executing the files.

The order of the job files is maintained by adding a timestamp to the file name, starting with the year, followed by the month, day, and time up to the second. On UNIX, the process ID is added to make the job truly unique when two jobs are scheduled for precisely the same time (each time `OvObsEnqueue` runs, it will have a higher process ID). On Windows, process IDs are recycled so they cannot be used to create a unique file name. Instead, the time in milliseconds is added. It is unlikely that two `OvObsEnqueue` processes start during the same millisecond. The time in milliseconds is not used in UNIX because it cannot be retrieved in Perl without installing third party modules that link into runtime libraries on the host system.

The queuing tools are shipped with scripts (batch files on Windows). The scripts function as wrappers around the queue commands. The name of a script file refers to its corresponding queue command (in the case of Windows, the name of the queue command followed by `.bat`). The scripts queue the command supplied as argument, and immediately de-queue it, unless another `OvObsDequeue` process is already active. The scripts are used in scenarios where the command inserted in the queue must be executed as soon as possible. In such a scenario, it could happen that the last few jobs put in the queue are not seen by the `OvObsDequeue` process and therefore, the `OvObsDequeue` process will have to be started again after it finishes the first time around to make sure that the jobs are in fact executed as soon as possible.

Implementing Event Forwarding with Object Loader

Introduction

Object Loader is one way to send data to Object Server without interacting with the HP OpenView console. There are other ways to do this as well. You can use inbound email messages or write a Java program that utilizes the Object Server Web API. You would use Object Loader rather than email when you need direct feedback on the success of the operation (which email cannot provide). Object Loader is a good alternative to Java when you want to manipulate a great number of objects in Object Server that are loosely related to each other and these manipulations are triggered one at a time. In that situation, starting up the JVM for each request will waste a lot of time compared to using Object Loader.

In other words, Object Loader is an integration tool, a program to be called by other programs for simple interaction with Object Server without user intervention.

Installing Object Loader

For instructions on installing Object Loader, see the *HP OpenView Service Desk Installation Guide*. The installation process installs the `HPOvObsLo` package. This package is dependent on the `HPOvPerlA` package. Object Loader is compliant with localization (L10N) and internationalization standards. Messages generated by Object Loader will be in the locale of the user. Messages generated by Object Server in response to Object Loader events will be send to the client (Object Loader) in the language of the ACCOUNT.

To simplify maintenance, you will probably also want a configuration file with defaults. This configuration file is called `OvObsLoadObject.conf`, and can be found in the `<`*`data_directory`*`>\conf\obs\loadobject` directory. The operating system account running `OvObsLoadObject` must have access to the network.

Exit Codes

When the `OvObsLoadObject` program stops, it sets an exit code that the calling program may use to verify what Object Loader has done. Table 9-2 describes the exit code values.

Table 9-2 Exit Code Values

Return Code	Description
2	`OvObsLoadObject` invoked with incorrect usage.
3	Not used.
4	Not used.
5	Result of a re-send event request.
6	Unable to rename the named file.
7	Unable to open named file.
8	Not used.
9	Unable to connect to an application server.
0	Good return code.
99	Perl not found.

Errors and Logging

Object Loader keeps two log files in the `<*data_directory*>\log\` directory: one for events that were successfully sent to the application server and one for events that could not be processed. When an event fails, the reason for the failure is stored in either the `CLIENT_RESPONSE` or `SERVER_RESPONSE` field.

Nothing is entered in the log files if you start `OvObsLoadObject` when the command-line parameters cannot be processed successfully. In that situation, exit code 1 is returned and the error is written to standard output.

Some Object Loader server errors are not interpreted as errors and are logged in the normal log file, sometimes with the prefix `WARNING`. In other cases, the full error message from the server is not logged in either log file, but is delivered as a generic error message. For troubleshooting purposes, it is useful to turn on DEBUG-level logging on the server. This should result in further error information being logged to the Object Server log.

CLIENT_RESPONSE Field in Log File

Whenever a `CLIENT_RESPONSE` field is logged, it means that the communication with the application server failed, most likely because the wrong connection parameters were specified. When a client error is reported, review the values used for server, port, account and onfail.

SERVER_RESPONSE Field in Log File

The `SERVER_RESPONSE` field is filled both for successfully sent events as well as those that failed. For a successful event request, the `SERVER_RESPONSE` field contains `OK`. When the server successfully processed the request but the result of the operation is doubtful, the `SERVER_RESPONSE` field contains a string starting with `Warning:` A `SERVER_RESPONSE` starting with `Error:` is indicative of a request that was received by the server but could not be processed. Whatever is stated after `Error` should give you an indication of what is wrong. If the problem does not become apparent immediately from looking at the error message, review the values used for account, import mapping, class name, and value list.

Client Error Messages

The following list specifies the error messages that are specific to Object Loader.

QUEUE[-ing] related messages:

- `open_queue     := Opening the queue %s`
- `close_queue    := Closing the queue ...`
- `add_entry      := Adding entry to the queue ...`
- `done           := Done.`
- `queue_blocked  := Queue %s is blocked.`
- `race_condition := Race condition for queue entry detected.\n`

- `proc_failure   := Execution of %s failed, exit code: %s`
- `last_os_error  := Last known OS error was: %s (but may or may not be associated with the current problem)`
- `flush_queue    := Flushing the queue %s`
- `blocking       := Blocking the queue %s`
- `unblock        := Unblocking the queue %s`
- `status_queue   := The Queue status = %s`
- `start_dequeue  := Starting the queue processor for queue %s`
- `end_dequeue    := Finished processing`
- `invalid_job    := Error opening job %s.`
- `invalid_queue  := The Queue %s does not exist.`
- `job_del_err    := Could not remove job %s.`
- `err_not_blocked:= The Queue %s was not blocked.`
- `err_blocked    := The Queue %s is blocked.`
- `err_unblock    := Could not unblock queue %s.`
- `err_log        := Can not open log %s for writing.\n$!`
- `already_running:= The Queue %s is already running.`
- `status         := The Queue status = %s`
- `entries        := The Queue contains : %s entries.`
- `interrupted    := Interrupt received!`
- `err_fork       := Error forking process: %s`
- `dequeued       := Dequeued %s jobs.`

ERROR CODES

- `OvObsLo-1010E  := The following required parameter(s) are missing from the command line or have not been specified in the configuration file: %s`
- `OvObsLo-1020E  := Can not open the file: %s`
- `OvObsLo-1030E  := No sections found in the configuration file %s`

- `OvObsLo-1040E  := Can not rename the file: %s`
- `OvObsLo-1050E  := Can not connect to HTTP daemon on %s`
- `OvObsLo-1060E  := Or any of %s`
- `OvObsLo-1070E  := The ValueList is empty!`
- `OvObsLo-1080E  := Invalid modus: %s`
- `OvObsLo-1090E  := The expected LoadObject messages file is missing`
- `OvObsLo-1100E  := No usage statement available!`
- `OvObsLo-1110E  := The file %s is not readable by the effective uid`

WARNING CODES

- `OvObsLo-5010W  := Could not open %s`
- `OvObsLo-5020W  := Output from the current command is : %s`

10 Troubleshooting Data Exchange

This chapter explains troubleshooting for Data Exchange.

Troubleshooting Export

```
Error: Unable to retrieve database connection to:
'jdbc:odbc:org_excel', user ''
```

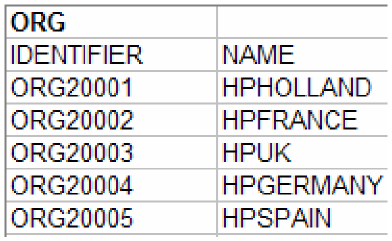
The data source is open by another user. This example shows that an Excel sheet is open by another user.

```
Error: Section [ORGANIZATION]: Error in processing SQL
statement [Microsoft][ODBC Excel Driver] Too few parameters.
Expected 1.
```

There is a mismatch between the `ATT` and `COLUMNS` information in the `INI` file or one of the columns does not exist in the data source. Check for typographical errors in the `INI` file.

Figure 10-1 provides an example of how to troubleshoot this error

Figure 10-1 **Data Source Example**



ORG	
IDENTIFIER	NAME
ORG20001	HPHOLLAND
ORG20002	HPFRANCE
ORG20003	HPUK
ORG20004	HPGERMANY
ORG20005	HPSPAIN

Example `INI` file contents:

```
[ORGANIZATION]
SOURCE=[ORG]
ATT=[IDENTIFIER], [NAME]
COLUMNS=DISTINCT [SEARCHCODE], [NAME]
LOADTABLE=TRUE
```

Change the `COLUMNS` information in the `INI` file as follows:

```
[ORGANIZATION]
SOURCE=[ORG]
ATT=[IDENTIFIER], [NAME]
COLUMNS=DISTINCT [IDENTIFIER] as [SEARCHCODE], [NAME]
LOADTABLE=TRUE
```

Troubleshooting Import

```
Error: The file provided is not a valid CIM file for Data
Exchange. Unexpected root element '' was found
```

This error can indicate the following:

- The structure of the XML file is invalid or damaged, for example, the file is empty.
- The required DTD file is not present in the XML file directory. During the export process, the `cim_dtd_v20.dtd` file is created. This file needed because it describes the structure of the XML file. Each XML file that is created with the Data Exchange export process contains a reference to the DTD file. This reference is as follows:

  ```
  <!DOCTYPE CIM SYSTEM "cim_dtd_v20.dtd"
  ```

```
Error: Parser error: For input string: "ORG20001"
```

This error can be generated because the ID is being used as the column name. An example of how to troubleshoot this error is shown in the following XML file:

```
...
<INSTANCE CLASSNAME="ORGANIZATION">

<PROPERTY NAME="ID" TYPE="string">
<VALUE>1</VALUE>
</PROPERTY>

<PROPERTY NAME="ID" TYPE="string">
<VALUE>ORG20001</VALUE>
</PROPERTY>
...
```

The first property is the internal ID used by Data exchange to keep track of records. The second property is the ID value of the organization, ORG2001.

It would be incorrect to change the export configuration so that ID is not used as a column name, for example:

```
COLUMNS=DISTINCT [ID] as [SEARCHCODE]
```

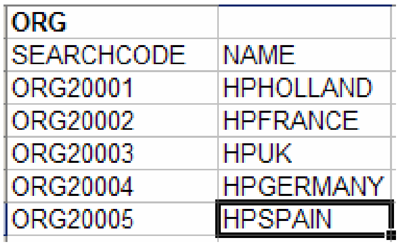
This is incorrect because the resulting XML file would be as follows:

```
...
<INSTANCE CLASSNAME="ORGANIZATION">

<PROPERTY NAME="ID" TYPE="string">
<VALUE>1</VALUE>
</PROPERTY>

<PROPERTY NAME="NAME" TYPE="string">
<VALUE>HPHOLLAND</VALUE>
</PROPERTY>
...
```

The organization ID value is not stored in the XML file because Data Exchange confuses the organization ID value with the record number ID during the export process. In this case, the import fails because ID is used as the primary key and Data Exchange cannot find an ID value in the XML file. However for non-primary keys, the import succeeds without error without importing the organization ID value.

The correct solution is to change the name of the column in the data source as shown in Figure 10-2.

Figure 10-2 **Data Source**



ORG	
SEARCHCODE	NAME
ORG20001	HPHOLLAND
ORG20002	HPFRANCE
ORG20003	HPUK
ORG20004	HPGERMANY
ORG20005	HPSPAIN

The `INI` file must be updated as follows:

```
ATT=[SEARCHCODE], [NAME]
COLUMNS=DISTINCT [SEARCHCODE], [NAME]
```

The XML is then as follows:

```
...
<INSTANCE CLASSNAME="ORGANIZATION">
<PROPERTY NAME="ID" TYPE="string">
<VALUE>1</VALUE>
</PROPERTY>
<PROPERTY NAME="SEARCHCODE" TYPE="string">
<VALUE>ORG20001</VALUE>
</PROPERTY>

<PROPERTY NAME="NAME" TYPE="string">
<VALUE>HPHOLLAND</VALUE
</PROPERTY>
...
```

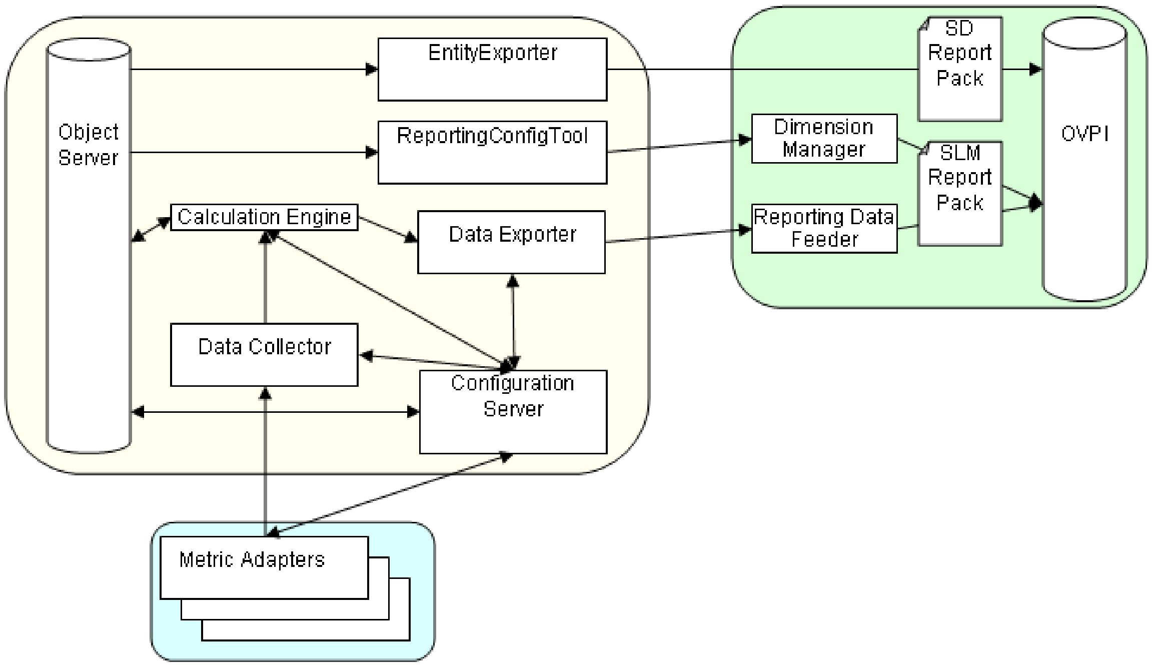
11 Troubleshooting the SLM Module

This chapter provides troubleshooting information for the Service Desk SLM module.

SLM Module Components

Figure 11-1 illustrates the components of the SLM module explained in this chapter.

Figure 11-1 **SLM Module Components**



Runtime Information: the Dump Command

In addition to tracing, you can troubleshoot problems by creating a dump of all variables in an application using the `dump` command.

You can use the following kinds of dump:

- Synchronized Dump: Safe to run at any time.
- Non-synchronized Dump: Not safe to run when the application is active. Useful if everything is blocked in the application because of synchronization.

You can perform a dump using a web browser (Internet Explorer, Netscape Navigator). Type the dump URL in the Address field. The dumped data is displayed in the browser.

Because web browsers use a cache, check the dump time to find out if it was redone in the application when trying to redo it from the web browser.

The available dump commands are:

- SLM core (Configuration broker, Calculation Engine, and Licensing only):

  `http://<SLM-host>:383/com.hp.ov.sd.slm/CCMServer/Command?Dump`

  `http://<SLM-host>:383/com.hp.ov.sd.slm/CCMServer/Command?DumpUnsafe`

- OVIS Metric Adapter:

  `http://<MA-host>:383/com.hp.ov.sd.slm/CCM/OvisMA/Command?Dump`

  `http://<MA-host>:383/com.hp.ov.sd.slm/CCM/OvisMA/Command?DumpUnsafe`

- OVPM Metric Adapter:

  `http://<MA-host>:383/com.hp.ov.sd.slm/CCM/OvpmMA/Command?Dump`

  `http://<MA-host>:383/com.hp.ov.sd.slm/CCM/OvpmMA/Command?DumpUnsafe`

- OVSN Metric Adapter

  ```
  http://<MA-host>:383/com.hp.ov.sd.slm/CCM/OvsnMA/
  Command?DumpUnsafe
  ```

- OVSD Metric Adapter:

  ```
  http://<MA-host>:383/com.hp.ov.sd.slm/CCM/OvsdMA/
  Command?Dump
  ```

  ```
  http://<MA-host>:383/com.hp.ov.sd.slm/CCM/OvsdMA/
  Command?DumpUnsafe
  ```

- Open Metric Adapter:

  ```
  http://<MA-host>:383/com.hp.ov.sd.slm/CCM/OpenMA/
  Command?Dump
  ```

  ```
  http://<MA-host>:383/com.hp.ov.sd.slm/CCM/OpenMA/
  Command?DumpUnsafe
  ```

Propagation of Changes (Attribute Change Control Management)

This section explains the control applied by the SLM business logic to attributes that play a specific role in the SLM Calculation Engine and or SLM reporting. It also covers the effects of changing these attribute values.

The attributes can be grouped in five categories, as shown in Table 11-1.

Table 11-1 Attribute Categories

Cat	Control Performed in Ovconsole	Effect on Calculation if Value Changes	Effect on Reporting	
			Before Reporting ConfigTool	After Reporting ConfigTool
1	Attribute can be edited at all times.	Modifications are applied dynamically.	Old value applied on reporting summaries. Data after change is incorrect.	New value is applied on reporting summaries. Data before change is incorrect.
2	Attribute can be edited at all times	Modifications are applied in SLA when Management Status changes: Managed → Not Managed or when SLM Core restarts.	Not applied.	Not applied Data is not lost.

Table 11-1 Attribute Categories (Continued)

Cat	Control Performed in Ovconsole	Effect on Calculation if Value Changes	Effect on Reporting	
			Before Reporting ConfigTool	After Reporting ConfigTool
3	If related SLAs are Managed, Error Message is displayed when attribute is edited.	Modifications are applied by SLA when SLA Management Status changes: Not Managed → Managed. All Statuses of all related SLAs, Services, CIs deleted.	Old value is applied on reporting summaries. Data after change is incorrect.	New value is applied Old value not accessible.
4	Attribute cannot be edited unless linked to SLA.	N/A	N/A	N/A
5	Cannot be edited unless associated Status is linked (managed once). Delete/add Service metric allows you to again add the operator.	N/A	N/A	N/A

Table 11-2 associates each attribute with one of the five attribute categories. All attributes that are not listed here are not controlled specifically by SLM. Changing these attribute values has no impact.

Table 11-2 Attribute Associated with Category

Attributes	Category
SLA-Applied Timezone	3
SLA-Evaluation Period	3
SLA-Hierarchy Filter	4
SLA-Schedules of type Service Hours	3
SLA-Service Definition	4
SLA-Service Level	4
SLA-Services	3
SLA-Actual Start	3
SLA-Actual Finish	3
SLA-Service Provided by Organization	3
SLA-Timezone Used For Service Hours	3
Service-Schedules of type Service Hours	3
Service-Service Metric SLO (Edit/Add/Delete)	3
Service-Service Metric CSLO (Violation Change of Value)	Not editable
Service-Service Metric CSLO (Jeopardy Change of Value)	1
Service-Service Metric CSLO (Jeopardy Add/Delete)	1
Service-Service Metric CSLO ((All other) Change of Operator)	5

Table 11-2 Attribute Associated with Category (Continued)

Attributes	Category
Service-CI Metric SLO (Change of Value)	1
Service-CI Metric SLO (Change of Operator)	5
Service-CI Metric SLO (Add/Delete of Metric)	3
Service-Sub CI of any relation type (Uses CI, Managed Supported CI)	2
Service-Sub Services of any type (Uses Services, Parent Child)	2
Service-Availability Propagation rule	2
Service Level-Schedules of type Service Hours	3
Configuration Item-Availability Propagation rule	2
Configuration Item-Metric Calculation Rule	2
Configuration Item-CI Metric	2
Configuration Item-Sub CI of any relation type (Related CI, Managed Supported CI)	2
Configuration Item-Schedules of type Planned Downtime	2

SLM Module Component-Specific Troubleshooting

The following section covers troubleshooting for components in the SLM module.

Metric Adapters

Common Metric Adapters

Table 11-3 Trace Components (Common Metric Adapters)

Trace Components
`com.hp.ov.sd.slm.sa.common.CCM` BBC command server. Useful for trace initialization of BBC command server and all handlers.
`com.hp.ov.sd.slm.sa.common.Config` Extension of `com.hp.ov.sd.slm.sa.common.Configurations`.
`com.hp.ov.sd.slm.sa.common.Configurations` Management of Metric Adapter configuration (MA parameters, connector parameters, discovery filters, data collection tasks, MRPs/metrics, MRP/metric definitions, locations). Use to trace loading and saving for configuration.
`com.hp.ov.sd.slm.sa.common.CSClient` Connection and communication from the Metric Adapter to the SLM server Configuration Broker. Use to trace messages sent to the SLM server Configuration Broker.
`com.hp.ov.sd.slm.sa.common.Discovery` Common code for any Metric Adapter to manage any discovery (MRP/metric, MRP/metric definition, location). Use to trace targets discovered new or deleted, and activation of MRP/metric (only) discovery.

Table 11-3 Trace Components (Common Metric Adapters) (Continued)

Trace Components
`com.hp.ov.sd.slm.sa.common.InitiatorTask` Common code for internally scheduled tasks used for data collection (manage data collection specific parameters). Use to trace reading of parameters, and management of list of unreachable Datapoints.
`com.hp.ov.sd.slm.sa.common.LocationDiscovery` Common code for any Metric Adapter to manage location discovery definition. Use to trace the activation of location discovery.
`com.hp.ov.sd.slm.sa.common.Monitoring` Common code for any Metric Adapter to manage monitoring statuses of connectors, tasks, and publisher. The monitoring status is included in the heartbeat message sent periodically to the SLM server Data Collector.
`com.hp.ov.sd.slm.sa.common.MrpDefDiscovery` Common code for any Metric Adapter to manage MRP/metric definition discovery. Use to trace activation of MRP/metric definition discovery.
`com.hp.ov.sd.slm.sa.common.Publisher` Messages publishing to SLM server Data Collector. Use to trace Datapoints to be sent.
`com.hp.ov.sd.slm.sa.common.Scheduler` Schedules all internally scheduled tasks. Controls execution of each task at each period.
`com.hp.ov.sd.slm.sa.common.SchedulerTask` Common code for any internally scheduled task (data collection, discovery, heartbeat).
`com.hp.ov.sd.slm.sa.common.SuperVisor` Common code for any Metric Adapter for core (relationship management of all Metric Adapter components: config, task, discovery).

Table 11-3 Trace Components (Common Metric Adapters) (Continued)

Trace Components
`com.hp.ov.sd.slm.sa.common.UniversalConnector` Common code for any Metric Adapter to manage the Connector of monitored application.
`com.hp.ov.sd.slm.sa.common.URIHandlerChangeConfig` BBC handler of config change (changing Metric Adapter config parameters: DefaultTaskPollingPeriod, DataPointSynchronizationDelay) from SLM server Config Broker.
`com.hp.ov.sd.slm.sa.common.URIHandlerCommand` BBC handler of command messages (request config, refresh Datapoints, start MRP/metric discovery, start MRP/metric definition discovery, start location discovery, dump) from SLM server Config Broker.
`com.hp.ov.sd.slm.sa.common.URIHandlerNotification` BBC handler of notification messages (MRP/metric discovery confirmation, MRP/metric definition discovery confirmation, location discovery confirmation) from SLM server Config Broker.
`com.hp.ov.sd.slm.sa.common.URIHandlerUpdate` BBC handler of update messages (MRP/metric, MRP/metric definition filter, location filter) from SLM server Config Broker.
`com.hp.ov.sd.slm.sa.hb` Deprecated for `com.hp.ov.sd.slm.sa.common.HeartBeatTask` in MR. Common code for any internally scheduled task used for heartbeat. Use to trace heartbeat message sent to SLM server Data Collector.

OVIS Metric Adapter

Table 11-4 OVIS Metric Adapter

Trace Application Name	Log File
OvisMA	OvisMAx.y.en_US

Table 11-5 Trace Components (OVIS Metric Adapter)

Trace Components
com.hp.ov.sd.slm.sa.ovis.*   HP OpenView Internet Services Metric Adapter traces (to be enabled with com.hp.ov.sd.slm.sa.common.* to keep all traces).
com.hp.ov.sd.slm.sa.ovis.Connector   Connector to OVIS. Use to trace SQL queries executed to OVIS database and internal information related to OVIS product.
com.hp.ov.sd.slm.sa.ovis.Discovery   Any discovery (MRP/metric, MRP/metric definition, location) management for OVIS Metric Adapter. Use to trace internal steps of OVIS discovery.
com.hp.ov.sd.slm.sa.ovis.InitiatorTask   Data collection task of OVIS Metric Adapter. Use to trace activation of thread, arguments used by the OVIS Connector to retrieve data, and collected Datapoints.
com.hp.ov.sd.slm.sa.ovis.OvisDataPoint   Datapoint for OVIS Metric Adapter. Use to trace problems when creating an OVIS Datapoint.

OVPM Metric Adapter

Table 11-6 OVPM Metric Adapter

Trace Application Name	Log File
`OvpmMA`	`OvpmMAx.y.en_US`

Table 11-7 Trace components (OVPM Metric Adapter)

Trace components
`com.hp.ov.sd.slm.sa.perf.*` HP OpenView Performance Manager Metric Adapter traces (to be enabled with `com.hp.ov.sd.slm.sa.common.*` to keep all traces).
`com.hp.ov.sd.slm.sa.perf.Connector` Connector to OVPM. Useful to trace the HTTP queries executed to the OVPM HTTP server and internal information related to OVPM product.
`com.hp.ov.sd.slm.sa.perf.Discovery` Any discovery (MRP/metric, MRP/metric definition, location) management for OVPM Metric Adapter. Use to trace internal steps of OVPM discovery.
`com.hp.ov.sd.slm.sa.perf.InitiatorTask` Data collection task of OVPM Metric Adapter. Useful to trace the activation of the thread, the arguments (HTTP request) used by the OVPM Connector to get data, and the collected Datapoints.
`com.hp.ov.sd.slm.sa.perf.OvpmDataPoint` Datapoint for OVPM Metric Adapter. Useful to trace problems when creating an OVPM Datapoint.

OVSN Metric Adapter

Table 11-8 OVSN Metric Adapter

Trace Application Name	Log File
OvsnMA	OvsnMAx.y.en_US

Table 11-9 Trace Components (OVSN Metric Adapter)

Trace Components
com.hp.ov.sd.slm.sa.sn.*   HP OpenView Service Navigator Metric Adapter traces (to be enabled with com.hp.ov.sd.slm.sa.common.* to keep all traces).
com.hp.ov.sd.slm.sa.sn.Connector   Connector to OVSN. Use to trace the XML command to OVSN socket, and XML parsing from OVSN socket.
com.hp.ov.sd.slm.sa.sn.Discovery   Any discovery (MRP/metric, MRP/metric definition, location) management for OVSN Metric Adapter. Use to trace internal steps of OVSN discovery.
com.hp.ov.sd.slm.sa.sn.InitiatorTask   Data collection task of OVSN Metric Adapter. Use to trace initial activation of thread and connection.
com.hp.ov.sd.slm.sa.sn.OvsnDataPoint   Deprecated for com.hp.ov.sd.slm.sa.sn.SocketListener in MR). The SocketListener is used by the OVSN Metric Adapter Connector to listen to the status change notification messages sent by OVSN. Use to trace incoming XML message from OVSN.
com.hp.ov.sd.slm.sa.sn.SpiRules   SPI rule management for OVSN Metric Adapter. Use to trace the SPI rule parsed.

Table 11-9 Trace Components (OVSN Metric Adapter) (Continued)

Trace Components
`com.hp.ov.sd.slm.sa.sn.StatusChangesParser` Used by the OVSN Metric Adapter Connector to parse the status change messages from OVSN socket.

OVSD Metric Adapter

Table 11-10 OVSD Metric Adapter

Trace components
`com.hp.ov.sd.slm.sa.ovsd5.*` HP OpenView Service Desk internal Metric Adapter traces (to be enabled with `com.hp.ov.sd.slm.sa.common.*` to keep all traces).
`com.hp.ov.sd.slm.sa.ovsd5.Connector` Connector to Service Desk (through Object Server). Use to trace notification (value change, creation, deletion), OVSD-Metric entity listing.

Table 11-11 Trace Components (OVSD Metric Adapter)

Trace Application Name	Log File
`OvsdMA`	`OvsdMAx.y.en_US`

Open Metric Adapter

Table 11-12 Open Metric Adapter

Trace Application Name	Log File
`OpenMA`	`OpenMAx.y.en_US`

Table 11-13 Trace Components (Open Metric Adapter)

Trace Components
`com.hp.ov.sd.slm.sa.openma.*` Open Metric Adapter traces (to be enabled with `com.hp.ov.sd.slm.sa.common.*` to keep all traces).
`com.hp.ov.sd.slm.sa.openma.DataPool` Access to data sent from openadaptor. Use to trace when data is received from openadaptor.
`com.hp.ov.sd.slm.sa.openma.OvSLMSink` Data container to send any data from openadaptor to Open Metric Adaptor. Use to trace metric values received from openadaptor.
`com.hp.ov.sd.slm.sa.openma.OpenConnector` Connector to retrieve openadaptor data in Open Metric Adapter. Use to trace activity of the Connection.
`com.hp.ov.sd.slm.sa.openma.OpenInitiatorTask` Data collection task of Open Metric Adapter. Use to trace activation of the thread and collected Datapoints.

MA Simulator

Table 11-14 Trace Components (MA Simulator)

Trace Components
`com.hp.ov.sd.slm.sa.simulator.*` Metric Adapter Simulator traces (to be enabled with `com.hp.ov.sd.slm.sa.common.*` to keep all traces).
`com.hp.ov.sd.slm.sa.simulator.Discovery` Any MRP/metric discovery management for Metric Adapter Simulator. Use to trace activation and the MRPs/metrics discovered.

Table 11-14 Trace Components (MA Simulator) (Continued)

Trace Components
`com.hp.ov.sd.slm.sa.simulator.InitiatorTask` Data collection task of Metric Adapter Simulator. Use to trace the value sent for each Datapoint.

SLM Reporting

The following section covers SLM Reporting trace components.

Dimension Exporter

Table 11-15 Dimension Exporter

Trace Application Name	Log File
`Dimension Exporter`	`DimensionExporterx.y.en_US`

Table 11-16 Trace Components (Dimension Exporter)

Trace Components
`com.hp.ov.sd.slm.dimensionExporter` Used to check the SLM model and monitor the connection with the dimension manager.

Reporting Data Feeder

Table 11-17 Reporting Data Feeder

Trace Application Name	Log File
`com.hp.ov.sd.slm.reporting Server.dataFeeder.server. ServerMain`	`DataFeederx.y.en_US`

Table 11-18 Trace Components (Reporting Data Feeder)

Trace components
`com.hp.ov.sd.slm.reportingServer.dataFeeder.server` Display information about the connected client and transactions.
`com.hp.ov.sd.slm.reportingServer.dataFeeder` Display information on Datapoints and facts received.

Reporting Dimension Manager

Table 11-19 Reporting Dimension Manager

Trace Application Name	Log File
`com.hp.ov.sd.slm.reportingServer.dimensionManager.server.ServerMain`	`DimensionManagerx.y.en_US`

Table 11-20 Trace Components (Reporting Dimension Manager)

Trace component
`com.hp.ov.sd.slm.reportingServer.dimensionManager.server` Display information about transactions and their status.
`com.hp.ov.sd.slm.reportingServer.common` Display information about dimensions.

SLM Report Pack

The SLM Report Pack component is installed on the OVPI server. The PL/SQL procedures of this component aggregate the data for the reports and store the traces in the table `SLM_R_PROCEDURELOG`.

Service Desk Reporting

This section covers tracing components for Service Desk Reporting.

Entity Exporter

Table 11-21 Entity Exporter

Trace Application Name	Log File
`OvSDReporting`	`sd_report_admin.logx.y.en_US`

Table 11-22 Trace Components (Entity Exporter)

Trace Components
`sd.report.entityExporter` Use traces to understand all tasks performed by this process: • Export parameters (Full or Incremental, module name exported: Helpdesk or ChangeManagement, start date). • Exporter details: OVPI host, OVPI user. • Detail of the module to export: entities and attributes. • Detail of the entities exported: OID. • Result of the export (number of entities exported).

Report Pack

The Report Pack component is installed on the OVPI server. The PL/SQL procedures of this component aggregate the data for the reports and store the traces in the tables:

- `RSDHD_PROCEDURELOG` (HelpDesk)
- `RSDCM_PROCEDURELOG` (ChangeManagement)

Configuration Server

Table 11-23 Configuration Server

Trace Application Name	Log File
`OvSlm`	`slmx.y.en_US`

Table 11-24 Trace Components (Configuration Server)

Trace components
`com.hp.ov.sd.slm.ohs` The OHS library component. Involved in loading of Services Hierarchies by the Hierarchy Manager. Setting traces on this component lets you see how the hierarchy filter is applied to load the hierarchy and which hierarchies are loaded.
`com.hp.ov.sd.slm.HierarchyManager` The hierarchy manager controls loading of SLA hierarchy. Setting this trace flag gives you information on the `dates of last reload` and `dates of last changes`. For each SLA loaded, it shows deleted statuses (if SLA contract attributes are changed).
`com.hp.ov.sd.slm.common` This component is used by SLM components from the Configuration Server to the Calculation Engine and others. It gives a set of various services. However, activating the traces on this component activates traces on the creation of DataPoints used by the Calculation engine.
`com.hp.ov.sd.slm.cs` The configuration Server component. Set traces on this component to track activity at startup. Specifically for loading SLA Hierarchies.
`com.hp.ov.sd.slm.cs.CCM` CCM is the Command and Configuration Manager. Set the trace on this component to retrieve information on messages exchanged between the Configuration Server and the Metric Adapters, including: Adapter Configuration, Metric Discovery, and Metric Definition Discovery. Activate this trace flag when troubleshooting problems between SLM Server and Metric Adapters.

Table 11-24 Trace Components (Configuration Server) (Continued)

Trace components
`com.hp.ov.sd.slm.cs.sa` Involved in communication between SLM server and Metric Adapter. Used to instantiate objects that are Adapter-specific (OvisMa, OvpmMA, OvsdMa, OvSnMA, and OvOpenMA), such as Metrics or MetricAdapter instances.
`com.hp.ov.sd.slm.heartbeat` Manages the SLM server heartbeat.
`com.hp.ov.sd.slm.main` Handles Start and Stop phases of the SLM Server.
`com.hp.ov.sd.slm.obs` **`com.hp.ov.sd.slm.obs.model`** These components are an interface layer between the SLM server and the Object Server. Activate the trace on these components to retrieve information on Object Server access (reading and writing).

Calculation Engine / Data Collector / Data Exporter

Table 11-25 Calculation Engine / Data Collector / Data Exporter

Trace Application Name	Log File
`OvSlm`	**`slmx.y.en_US`**

Table 11-26 Trace Components (Calculation Engine / Data Collector / Data Exporter)

Trace Components
`com.hp.ov.sd.slm.network` Involved in Datapoint exchange. Designed to receive Datapoints from the network and send them to an external component (for example, reporting). Activate the trace on this component to trace the Datapoints flow between the different processes.

Table 11-26 Trace Components (Calculation Engine / Data Collector / Data Exporter) (Continued)

Trace Components
`com.hp.ov.sd.slm.MetricsAggregationMgr` Involved with Data Collector. Designed to receive Datapoints from the network, sort them and aggregate them according to the Duplicated Datapoint Policy. Activate the trace on this component to display which Datapoints are aggregated and which are sent to the Calculation Engine.
`com.hp.ov.sd.slm.ce` Calculation Engine component. Designed to receive Datapoints from the data collector component and to use their values to compute Compliances and Predictive Compliances of active SLAs. The calculation results are sent to the data exporter and written into Object Server. Activate the trace on this component to display Datapoints used for calculation.
`com.hp.ov.sd.slm.dataexporter` **`com.hp.ov.sd.slm.DataExporter`** Data Exporter component. Designed to export Datapoints and calculation results to Reporting. Activate the trace on this component to display information about the Data Exporter thread.
`com.hp.ov.sd.slm.broker` Used by the Data Exporter. Sends all received Datapoints to Reporting. Activate the trace on this component to display information about Datapoints export.

Alarm Engine

Table 11-27 Alarm Engine

Trace Application Name	Log File
`OvSlm`	`slmx.y.en_US`

Table 11-28 Trace Components (Alarm Engine)

Trace Components
`com.hp.ov.sd.slm.ae` Alarm Engine component. This software component of SLM is responsible for creating Incidents when the Calculation Engine computes that particular states are changed and thresholds are reached. Enable this trace component to trace the start and stop of the Alarm Engine software component, and the creation of Incidents by the Alarm Engine.
`com.hp.ov.sd.slm.ce` This trace component also displays Alarm Engine-related information. In particular, it contains information when the Calculation Engine decides to send Alarms (that is, create an Incident).

Business Logic

Table 11-29 Business Logic

Trace Application Name	Log File
`Ovconsole`	Same as console

Table 11-30 Trace Components (Business Logic)

Trace Components
`com.hp.ov.sd.slm.wf.biz` This part of the Business Logic is responsible for handling all SLM entities (for example, SLA, Services, SLOs, Configuration Items) using the HP OpenView console or the SLM Engine. Enable this trace component to display all Object Server entity operations for the HP OpenView console or to display SLM Engine operations on the SLM entities, for example CRUD operations (Create, Read, Update, Delete), and state changes.
`com.hp.ov.sd.slm` This trace component is used by the SLM Business Logic to log important events related to SLM Entity operations.
`com.hp.ov.sd.slm.wf.biz.StatusCleanupManager` This Business Logic component is responsible for status cleanup. Enable this trace component to display when status cleanup occurs and details of the operation.
`com.hp.ov.sd.slm.wf.biz.ObjectivesCleanupManager` This Business Logic component is responsible for objective status cleanup. Enable this trace component to display when objective status cleanup occurs and details of the operation.
`com.hp.ov.metric.wf.biz` This part of the Business Logic is responsible for the handling of Metrics. Enable these trace components to display the internal handling of metrics and the validation of metric values and types.

Service Designer

Table 11-31 Service Designer

Trace Application Name	Log File
`Ovconsole`	Same as console

Table 11-32 Trace Components (Service Designer)

Trace Components
`com.hp.ov.sd.slm.ui.util` **`com.hp.ov.sd.slm.ui.designer.*`** The Designer UI software component is responsible for editing and displaying Service Hierarchy graphs and the entities that build graphs (both UI objects and SLM entities). Activating the associated trace components gives tracing for operations performed using the designer: operations performed by the user, Designer response to user operations, and the outcome of operations.

Licensing

Table 11-33 Trace Components (Licensing)

Trace Components
`com.hp.ov.sd.slm.module` Code used by Object Server for SLM module. Use to trace licensed entities (to be enabled in the console).
`com.hp.ov.sd.slm.license.LicenseGuard` Licensing management in the SLM server. Use to trace license information, such as expiration date.

SLM Module Error Messages

This section describes the SLM error messages and explanations.

`You can not change this %1 as it is linked to a %2 and the %3 type is Aggregated`

For Aggregated Metrics, the SLO is automatically defined as the reverse of the CSLO. If the CSLO is already defined for an Aggregated Metric, the user cannot change it.

`You cannot delete this hierarchy filter because one or more services relate to it.`

You tried to delete a Hierarchy Filter that has at least one related service. Verify that the Hierarchy Filter is not used by a service before deleting it.

`You cannot delete this hierarchy filter because one or more SLAs relate to it.`

You tried to delete a Hierarchy Filter that is referenced by a service or an SLA.

`Can't relate this %1 to the %2 because the %1 contains invalid %3s.`

You tried to associate an Invalid Hierarchy filter to a Service or an SLA. For SLM, a valid hierarchy filter must have a service as the root object type.

`Unable to Update the table, Form returned null reference`

This error reports an internal software error in the service designer.

`Cannot change %1 as it is linked to a managed ServiceLevelAgreement`

Some SLM objects or attributes cannot be changed if they are related to a managed SLA. This error is generated if you try to change an SLM object or attribute when its corresponding SLA has Managed status. To prevent this, set the SLAs `Activity Status` to **Editable**, modify the SLM attribute, and reset the SLA to **Managed**.

`Cannot change %1 as it is linked to managed %2 %3`

Some SLM objects or attributes cannot be changed if they are related to a managed SLA. This error is generated if you try to change an SLM object or attribute when its corresponding SLA has Managed status. To prevent this, set the SLAs `Activity Status` to **Editable**, modify the SLM attribute, and reset the SLA to **Managed**.

`Cannot delete %1 as it is linked to a managed ServiceLevelAgreement`

Some SLM Objects cannot be deleted when they are part of a service hierarchy that is monitored by a managed SLA. This error is generated if you try to delete an SLM object when its related SLA has Managed status.

`You may not delete this %e: it is required for module %1`

This message is displayed if you try to delete an incident category kernel code.

`It is not possible that a Hierarchy Filter and a Service Definition is set`

This error is displayed if you try to associate a service definition to a service that is not compatible.

`Please create SLO in the Table above before trying to open form`

This message is displayed if you try to edit a Compliance Threshold directly. Edit the Compliance Threshold from the Compliance Threshold Table.

`This Compliance SLO cannot be deleted because it is referenced by the Service %1 which is linked to the SLA %2`

This error is generated when you try to delete a Compliance SLO that is referenced by a service that is in turn linked to an SLA.

`A Compliance SLO must have exactly one Compliance Threshold for Violation.`

This error is generated if a Violation Compliance Threshold is not set or if more than one is set.

`Thresholds have the same value or the same severity.`

This error is generated if the created Compliance Threshold is not valid.

`There must be less than 4 jeopardies Thresholds.`

This error is generated if more than four Thresholds are created for a CSLO.

`<html>%1 "%2"<p>Metric Definition: <b>%3</b></html>`

Tooltip text.

`This SLA Life Cycle Status (%1) is already mapped to a Management Status.`

This error is generated when the SLA Life Cycle Status is already mapped to a Management Status.

`Empty Field Error`

A required field is empty, enter the required information.

`Encounter exception during sending command '%1' to metric adapter '%2' on host '%3'.`

The requested action did not complete because of a communication problem with the Metric Adapter.

`Please enter an operator in the operator column first`

Specify an operator in the Operator field.

`Error`

Generic error reporting an internal software problem.

`Rule cannot be add. A least one parent rule must match From Object Type and From Category.`

When you add new rule to a Hierarchy filter the From Object Type (the type of object from which the filter rule starts to search) you can only specify an object type that an existing rule can retrieve. For example, before you add a rule that retrieves configuration items with a particular relation type, you need to add a rule that retrieves configuration items that are used by services.

`Error Creating Metric Details Link`

This message is displayed when an error occurred setting metric details for ServiceMetric or ConfigurationItemMetric.

`Error Creating Service Level Objective`

An error occurred while creating an SLO.

`Error deleting %1.`

Object Server problem when unrelating two service resource definitions.

`Error Modifying Service Level Objective Condition`

Error when modifying Service Level Objective Condition.

`Error Modifying Threshold`

Error when modifying Service Level Objective Threshold.

`Error Opening Form`

Error when opening a Compliance SLO Form.

`Error Opening Service Level Objective`

Problem in the modification of a compliance SLO.

```
Rule cannot be added. The rule is duplicated for the
hierarchy filter.
```

You tried to add an existing rule.

`Error saving %1.`

Generic error reported by the SLM Service Designer when it fails to save created or modified entities to the Object Server.

```
Failed to send command '%1' to Metric Adapter '%2' on host
'%3'. Please check logs for details.
```

The requested action did not complete because of a communication problem with the Metric Adapter.

```
From Object Type for first rule of a filter must match Root
Object Type of filter.
```

For the first rule you add to a hierarchy filter the From Object Type must be equal to the root object type of the hierarchy filter.

`This node is already a leaf node.`

You tried to make a Leaf a Node that is already a Leaf.

```
The Metric Adapter Definition value must be set to the Metric
Adapter Definition value of the Discovery Filter.
```

Created Metric Adapter does not have MetricAdapter definitions.

`Hierarchy filter cannot be changed if a related SLA is not in editable state.`

You tried to modify a Hierarchy Filter that has at least one associated SLA that is not in the `Editable` state. Ensure that all the associated SLAs are in the state `Editable` before editing the filter.

`The hierarchy graph cannot be show because the internal system filter for Service Definition '%1' could not be found!`

Internal error: a service definition that is used by a service does not have an underlying (hidden) internal system hierarchy filter. As a result, the hierarchy cannot be displayed.

`No unconfigured nodes found.`

Informational message. When you select **next unconfigured** or **previous unconfigured** in the `Service Hierarchy Graph` (**Service Hierarchy** tab in the **Service** form), unconfigured nodes in the hierarchy graph cannot be found, therefore all nodes are configured.

`Only one threshold with operator '=' or '!=' is allowed`

The Threshold Value validity depends on the ObjectiveCondition:

- `LessThan` or `LessThanOrEqual`: The violation threshold must have the lowest value.
- `GreaterThan` or `GreateThanOrEqual`: The violation threshold must have the highest value.
- `Equals` or `NotEqual`: Only one threshold is valid and the threshold value is fixed.

`To continue it is necessary to save the current %1. Do you want to save it now?`

The current entity must be saved to continue. If you do not save, the operation is not performed.

`Service Definition for Service Metric not set, unable to create Service Level Objective`

Internal error: A Service Metric Definition needs to be available when adding a Metric to a Service as a Service Metric. This error is generated when the Service Metric Definition is not created. An SLO cannot be created when this definition does not exist.

`You must define the Service Level Objective before defining any Compliance Threshold`

You must define a Service Level Objective (SLO) before you can create Compliance Service Level Objectives. This is because the CSLOs are based on the compliance of the SLOs. To solve this, define an SLO for the metric.

`The Hierarchy filter related to the service must match the hierarchy filter related to SLA`

You tried to relate a service to an SLA that have different hierarchy filters. Services associated with SLAs must have the same hierarchy filter. The incorrect service is associated with the SLA.

`Service Level Agreement status do not allow to modify linked Schedules`

You tried to modify the set of schedules linked to an active or managed SLA. Set the SLA to Editable status to change the schedule set. Be aware that doing this resets the SLA statuses and all the statuses linked to this SLA Evaluation Period.

`Service Level Agreement status do not allow to modify linked Services`

You tried to modify the set of services linked to an active or managed SLA. Set the SLA to Editable status to change the Services (add or remove). Be aware that doing this resets the SLA statuses and all the statuses linked to this SLA Evaluation Period.

`You cannot have a Hierarchy Filter related when relating a Service Definition. Hierarchy Filter gets removed for the SLA.`

It is not possible to have both a Service Definition and a Hierarchy Filter set on an SLA at the same time. You tried to associate a Service Definition to an SLA that already has a hierarchy filter set. To solve this, unrelate the hierarchy filter from the SLA, if the SLA state and associated entities allow this.

`Unable to modify Threshold`

Update of SLO threshold value failed. This message is accompanied with some contextual information to identify the cause of the failure.

`Only one instance of SLM Administration can be created`

It is not possible to create SLM Administration instances. The SLM Administration instance is unique. You should not try to create new instances of the SLM Administration.

`Incorrect value '%1' on entity %e: value must be a decimal number when %2 is '%3'`

You entered an incompatible data type value for `SLO` or for `CSLO Threshold`. The contextual information gives expected data type. Enter the correct data type value.

`The Threshold Value of the Violation Threshold must be less than all other Compliance Thresholds.`

When CSLO thresholds are set (standard metrics, infrastructure availability metric, aggregated metrics with an SLO operator Greater Than or Greater Than Or Equal), the Violation Threshold holds the lowest value of all the CSLO thresholds. You tried to create Jeopardy Thresholds that have lower values than the Violation Threshold or you tried to change the Violation Threshold to a value greater than the existing Jeopardy Threshold. Ensure that the Violation Threshold has the lowest value.

`The Threshold Value of the Violation Threshold must be greater than all other Compliance Thresholds.`

The Threshold Value of the Violation Threshold is not greater than all other Compliance Thresholds values.

Example Case - Troubleshooting SLM and Metric Adapters

Summary

In the case history described below, HP OpenView Service Navigator metric adapters (OvsnMA) cannot discover services in HP OpenView Service Navigator (OVSN). The section below describes how the customer experienced the problem, initial investigations, tracing, a proposed solution, the results of the proposal, and conclusions.

Customer's Description of the Case

"Some services are not being discovered by the Service metric adapter in Service Desk."

Also: "The OVSN services that have been previously discovered are present, but new services cannot be discovered using the HP OpenView Service Navigator metric adapter."

Case Details

Service Desk uses metric adapters to connect to HP OpenView Service Navigator (OVSN), and should collect the services in OVSN. The customer reports that some services are not being discovered by the Service Navigator metric adapter (OvsnMA). Restarting OVSN and manually running OvsnMA does not help. In addition, the customer cannot start the Service Level Manager server.

Customer Environment

Operating System: Windows 2003 SP1.

Service Desk Version: SD 5.0 SP1 is installed on the system.

Additional Information

Information relevant to the case is listed below.

- **Metric Adapter Location**

  The OvsnMA is also installed on the machine where SD 5.0 SP1 is running.

- **OVO Agent**

  An OVO agent was installed and uninstalled on the W2003 SP1 machine hosting the Service Desk management server / Service Level Management server.

- **OvsnMA.xml File**

  The `OvsnMA.xml` file (configured using MAconfig GUI, the tool for configuring metric adapters) specifies the OVSN connector settings. In this file, the OVSN host is specified using the Connector tag (the host is also specified in the Host, ConnectorRef, and OvsnServer tags). After startup, the OVSN Metric Adapter connects to the host machine using the default port 7278.

- **Server Identification**

  The host for the Service Desk Management server is `OVSD.Zinternet.pl.com`. The host for OVSN is OVOUsun.Zinternet.pl.com.

Initial Checks and Investigations

Some initial enquiries and checks were made to troubleshoot and find the source of the problem. These are listed below.

- **OVCS component**

  The `ovcs` service was checked and found to be running. The `system.txt` file reported that this component was restarted automatically by `ovc`.

- **Licensing**

  The services discovered so far were all managed, but there were insufficient licenses available (see log file `Slm0.0.en`). The SLM server was eventually started by first “unmanaging” the services.

However, the OVSN services were still not discovered, the error messages indicating connection problems to the server (see `OvsnMA0.0.en` in the "Log Files" section below).

The batch file `startmaconfig.bat`, used to start the OvsnMA, cannot execute successfully, and displays an error message on execution (see below). This occurs every 30 seconds. See also `OvsnMA0.0.en` in the "Log Files" section.

```
The Adapter discovery can't connect to ovsn for connector
OVOUsun.Zinternet.pl.com
```

- **Connectivity**

  Could the problem be a network connectivity issue or is the OVSN daemon not running? This is not the case - a Telnet session to the host machine was successful. No firewall blocking was found.

- **DNS**

  Is the OVSN host machine registered in DNS?

- **ovc -status**

  Executing the `ovc -status` command reveals the following:

  ```
  SLM ovsdslm is aborted
  ```

  ```
  MA ovsnma is aborted;
  ```

  (All other processes are running.)

Log Files

The section below displays relevant extracts from the log files generated during the failed OvsnMA discovery process.

(For an overview of logging in Service Desk, see "Logging" on page 67).

- **OvsnMA0.#.en**

  ```
  <DATE+TIME>com.hp.ov.sd.slm.util.SlmLogger;log;com.hp.ov
  .sd.slm.sa.sn. Discovery;WARNING; The Adapter discovery
  can't connect to OVSN, for connector:
  OVOUsun.Zinternet.pl.com.
  ```

  ```
  <DATE+TIME>com.hp.ov.sd.slm.util.SlmLogger;log;com.hp.ov
  .sd.slm.sa.common.Discovery;INFO;Metric discovery
  finished.
  ```

- **OvsdMA0.0.en**

  ```
  <DATE+TIME>com.hp.ov.bbc.impl.StatusHandlerRequest;negot
  iate;com.hp.ov.bbc.http;WARNING;Request for
  "http://localhost:383/com.hp.ov.sd.slm/DC/" denied by
  server: Unexpected HTTP status "404 Not Found".
  ```

- **Slm0.0.en**

  ```
  <DATE+TIME>com.hp.ov.sd.slm.util.SlmLogger;log;com.hp.ov
  .sd.slm.license.LicenseGuard;SEVERE;Not enough SLM
  service licenses to load SLA Xyz123
  ```

  ```
  <DATE+TIME>com.hp.ov.sd.slm.util.SlmLogger;log;com.hp.ov
  .sd.slm.HierarchyManager;SEVERE; Not enough service
  licenses to load SLA Xyz123
  ```

Attempting to Resolve the Problem

Two variations of the tracing process can be used to investigate the problem, and these are described below. The objective is to discover why the OvsnMA fails to connect to OVSN.

(For an overview of HP OpenView tracing, see “HP OpenView Tracing Fundamentals” on page 93).

Tracing Process 1

1. Stop all `ovc` processes by entering **`ovc -kill`** at the command line.
2. Delete the files `OvsnMA_discovered.props` and `OvsnMA_discovery.xml`. If these files already exist, wait for the interval specified in the `OvsnMA.xml` file (`<DiscoveryInterval>` tag) before deleting.
3. Open the trace, and add the following Application OvsnMA components:
   - `com.hp.ov.sd.slm.sa.sn.Connector`
   - `com.hp.ov.sd.slm.sa.sn.Discovery`

   Set the value `Attribute Mask` to `Max`.
4. Restart the SLM Server, the Object Server, and the metric adapters by entering the following at the command line:

   ```
   ovc -start ovobs ovsdslm ovisma ovsnma
   ```

5. Check if any `ovc` process has aborted by entering **`ovc -status`** at the command line.
6. After approximately 10 minutes, you should see some information in the trace, after the OvsnMA has started its discovery process.
7. If no information becomes available, wait for the interval specified by `<DiscoveryInterval>` until the OvsnMA starts its discovery process. Alternatively, try to trigger discovery by entering the following command:

   ```
   http://localhost:383/com.hp.ov.sd.slm/CCM/OvsnMA/Command
   ?StartDiscovery
   ```
8. Save the contents of the trace to disk.
9. When the trace process has finished (it will show the connect problem recurring), save the following files to disk:
   - trace file (`*.trc`),
   - slm logfile (as `slm0.0.en`)
   - OvisMA and OvsnMA files (as `OvisMA0.0` and `OvsnMA0.0`, for example).

Results of Trace 1

The OvsnMA operates correctly, but warnings concerning the OVSN server connection process are generated. These occur when the OvsnMA sends a discovery request to the OVSN server though a socket and does not receive a reply. The expected reply is a predefined XML message that contains tags for `<results>`, `<services>`, `<service>`, and so on.

The relevant XML file can also be generated manually as follows:

1. Telnet or login to the OVSN server.
2. Execute the command:

   `opcservice -list -service -xml`

   This displays the reply from the OVSN server to a discovery request from the OvsnMA.
3. Check that the resulting XML file contains </services> tags.

Tracing Process 2

This process is identical to tracing process 1 except that a debug version of `OvSlmMaOvsn.jar` was used.

Results of Trace 2

The tracing process shows that the root cause of the problem is an `out of memory` exception that occurs when the OvsnMA accepts the OVSN response during discovery.

The default memory allocated to OvsnMA is 64 MB, which is probably insufficient for the OVSN.

The solution (see below) is to increase the memory allocation value, to 128 MB, for example. In the case history that forms the basis for this example, the customer increased memory to a value in the range 256 - 512 MB when even 128 MB appeared to be insufficient.

Proposed Solution and Test Results

Based on the tracing procedures and results outlined above, a solution is offered below. The proposal is based on the following assumptions: the number of OVSN metrics is very large. As a result, the default batch interval (60s) is insufficient for the MA discovery process to complete.

The proposal is to increase some intervals, and to increase the memory allocated to the OvsnMA process. Then check to see if this resolves the problem.

To increase some intervals and increase the memory allocated to the OvsnMA process, follow these steps:

1. In the `local_settings.ini` file (`C:\Program Files\HP OpenView\data\conf\xpl\config\local_settings.ini`), change the `MetricDiscoveryBatchInterval` value in the `[slm.ma]` setting to 3600:

   ```
   [slm.ma]
   MetricDiscoveryBatchInterval=3600
   ```

2. To effect this change, type **`ovconfchg`** at the command line. To check the new settings, type **`ovconfget`** at the command line.
3. In the console, display the OvsnMA form in the Metric Adapter workspace, and enter **`86,400`** in the Discovery Interval and Metric Definition Discovery Interval fields. See Figure 11-2 on page 222.

4. Increase the memory allocated to both the MA and the SLM server process to a value in the range 256 - 768 MB. Follow these steps:

   a. Stop the `ovsdslm` process by typing **`ovcreg -del ovsdslm`** at the command line.

   b. In the file `C:\Program Files\HP OpenView\data\conf\slm\OvSdSlmCtrl.xml`, change the settings for XYZ and XYZ. These are shown in the sections marked with an X below.

   ```
   <ovc:Start>
   <ovc:CommandLine>
   "$InstallDir\nonOV\jre\1.4\bin\java" -Xrs
   -Did=OvSlmServer -Dcom.hp.ov.sd.slm.main.Slm -Xms256m
   -Xmx768m -jar "$InstallDir\java\obs-launcher.jar"
   "$InstallDir\java\OvSlmServer.jar"
   </ovc:CommandLine>
   </ovc:Start>
   ```

   and

   ```
   <ovc:Execute>
   <ovc:CommandLine>"$InstallDir\nonOV\jre\1.4\bin\java"
   -Xrs -Did=OvSlmServer -Dcom.hp.ov.sd.slm.main.Slm
   -Xms256m -Xmx768m -jar
   "$InstallDir\java\obs-launcher.jar"
   "$InstallDir\java\OvSlmServer.jar" -stop
   </ovc:CommandLine>
   </ovc:Execute>
   ```

   c. Register the file with OvCtrl daemon by entering the following at the command line:

   ```
   ovcreg -add C:\Program Files\HP OpenView\data\conf\
   slm\OvSdSlmCtrl.xml
   ```

5. Use a discovery filter to discover the most important metrics (you do not need to discover all 20,000 metrics). Renew the filter in the workspace as follows:

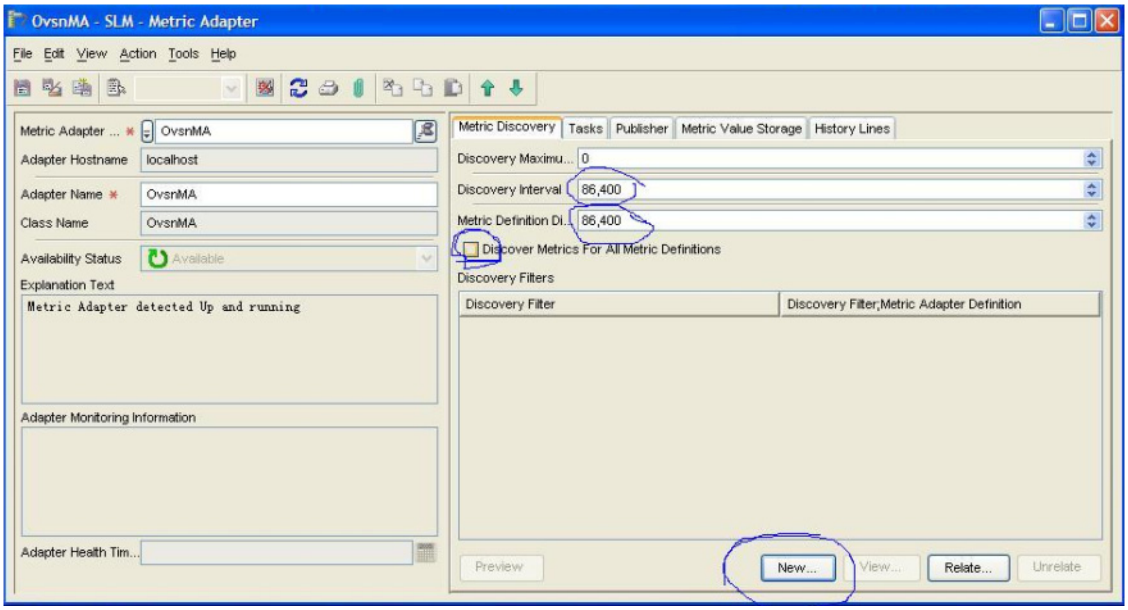
   a. Uncheck the Discover Metrics for All Metric Definitions checkbox.

   b. Press the New button in the bottom to add some filters (see Figure 11-2, “Metric Adapter Workspace”).

Only those metrics whose metric definition is covered in the Discovery Filter will be discovered by the OvsnMA and maintained by SLM.

6. Repeat the tracing procedures described in “Attempting to Resolve the Problem” on page 218.

Figure 11-2 Metric Adapter Workspace



OvsnMA - SLM - Metric Adapter

Metric Adapter ... * OvsnMA
Adapter Hostname localhost
Adapter Name * OvsnMA
Class Name OvsnMA
Availability Status Available
Explanation Text
Metric Adapter detected Up and running
Adapter Monitoring Information
Adapter Health Tim...

Discovery Maximu... 0
Discovery Interval 86,400
Metric Definition Di... 86,400
☐ Discover Metrics For All Metric Definitions
Discovery Filters

Discovery Filter	Discovery Filter;Metric Adapter Definition

Results and Conclusion

The results of the proposed solution outlined above were as follows:

- Memory allocation to the Java VM had to be increased considerably (from 64 MB to 512 MB) to ensure correct functioning.
- The tracing debug version resulted in `out of memory` errors in the OvsnMA log file, as shown below.

```
<DATE+TIME>com.hp.ov.sd.slm.util.SlmLogger;log;com.hp.ov.sd.
slm.sa.sn.Discovery;WARNING;>< The Adapter discovery can't
connect to OVSN, for connector: OVOUsun.Zinternet.pl.com
```

```
<DATE+TIME>com.hp.ov.sd.slm.util.SlmLogger;log;com.hp.ov.sd.
slm.sa.sn.Discovery;SEVERE;Error stack:
java.lang.OutOfMemoryError
```

Conclusion

The case was resolved by combining several of the strategies outlined in preceding sections:

- Increasing the timeout values.
- Increasing memory allocation.
- Reducing the amount of metric data values gathered simultaneously.

12 Troubleshooting Service Desk OVO for UNIX Integration

HP OpenView Operations for UNIX and Service Desk use components that can conflict with each other. This can result in applications aborting or connections failing. The workarounds in this section allow you to work with these problems.

Management Server Does Not Start After Deploying OVO for UNIX Agent

After deploying the OVO for UNIX agent on a Service Desk management server the Service Desk management server does not start. If you run `ovc` the following messages are returned:

```
(ctrl-113) Control daemon is running...But cannot be
accessed remotely. Please wait some more time or restart the
daemon.

(ctrl-21) Communication error when executing 'Status'
method.

(xpl-118) recv() on '127.0.0.1:1733' failed.

(WIN-10054) An existing connection was forcibly closed by
the remote host.
```

Explanation

The CoreID of the managed node is overwritten by its previous value as stored in OVO for UNIX. When the Service Desk management server is installed, it generates a unique CoreID. If you re-install the management server on the same machine (for example, after re-imaging the new environment, or after a HDD failure and replacement), a new CoreID is generated by the installation. When the OVO for UNIX agent is redeployed on this managed node, the new CoreID is overwritten by its predecessor.

Workaround

On the OVO for UNIX management server, delete the managed node (the Service Desk management server) and add it again as a new node before redeploying the OVO for UNIX agent.

OVO for UNIX Agent does not Communicate with the OVO Management Server

After deploying the OVO for UNIX agent on the Service Desk managed node, if you run `opcagt`, it returns that it is buffering messages for the OVO for UNIX management server. If the `opcmsg` template was deployed on the managed node, this does not run.

Explanation

Certificate conflict or missing certificates. To check this, run:

```
opcragt <managed node name>
```

If there is a certificate problem the following messages are returned:

```
Node <managed node name>:

CTRL - CommunicationException:

(ctrl-21) Communication error when executing 'Status'
method.

(sec.core-113) SSL certificate verification error (The
presented peer certificate is not trusted. The certificate
verification chain could not be built.). (OpC40-2130)

Cannot get status information from node <managed node name>.
(OpC40-428) Network communication problems occurred.
(OpC40-427) Failed.
```

Workaround

- Run **`ovcert -list`** to display an overview of your certificates.
- If the list is empty, run **`ovcert -certreq`** to request new certificates and grant them access rights on the OVO for UNIX management server.
- If a list of certificates is returned, remove the top two certificates using **`ovcert -remove`** ***`<certificate number>`***. It is useful to make a backup of these certificates before removing them. Do not remove the last two certificates, ov resource group (OVRG).

- Run `ovcert -certreq` to request new certificates and grant them access rights on the OVO for UNIX management server.

IMPORTANT If you are using the OVO for UNIX - SD Integration in environments that use LDAP or Active Directory authentication for Service Desk clients, this workaround can result in authentication problems because of specific Certificate Server configurations. The functional integration between OVO for UNIX and Service Desk in the above scenario is not tested.

Tomcat Service Conflicts

After installing the Service Desk management server on a UNIX machine that runs an NNM or OVO management server, the `ovtomcatA` (OV Tomcat(A) Servlet Container) service aborts after startup. Service Pages and Web UI sessions cannot connect to the Service Desk management server.

Explanation

NNM and OVO run a separate session of the Tomcat service. The Tomcat service is used by the `ovas` (HP OpenView Application Server) service that allows use of the web interface. The Service Desk management server also uses Tomcat. This results in a conflict on port 8005, causing the service to abort. To check if there is a LISTEN process running indicating the process is in use, enter:

```
netstat -a | grep 8005
```

Workaround

Port 8005 is used by the Service Desk management server to remotely shutdown the Tomcat Server.

To change this, go to the installation `bin` folder and do the following:

1. Run **`./ovconfchg -edit`** (windows: **`ovconfchg.exe -edit`**)
2. Find the entry: `ShutdownPort=8005` and change the port number to any unused port, for example, **`8050`**.
3. To verify that the setting has changed run:

   `./ovtomcatctl -getconf` (windows: **`cscript.exe ovtomcatctl.vbs -getconf`**).
4. Start the `ovtomcatA` service.
5. Verify that you can connect with Service Pages or the Web UI.

NOTE This problem depends on the order in which the applications were started. If Service Desk was started first, then a similar problem can occur for NNM/OVO with the `ovas` service. However, you can solve this using this workaround first and then start `ovas` manually.

13 Service Desk Coexistence Issues

This chapter contains information on coexistence of Service Desk 5.0, Service Desk 5.0 SP1, and Service Desk 5.10 with other HP OpenView products on the same machine.

The following topics are discussed:

- **Coexistence with HP OpenView Operations**

  Provides a list of Service Desk components (5.0 and 5.0 SP1) that can coexist on a machine where HP OpenView Operations (OVO) components are installed.

- **Workarounds**

  For a specific instance of coexistence (Service Desk 5.0 SP1 and OVO for UNIX 8.1 on Windows 2000 AS), two workarounds are described.

IMPORTANT The information provided in this section is subject to frequent changes.

For the most recent information, refer to the coexistence section of the Service Desk 5.0 supported platforms list available from the following location:

`http://openview.hp.com/ecare/getsupportdoc?docid=OV-EN018535`

For the most recent information, refer to the coexistence section of the Service Desk 5.10 supported platforms list available from the following location:

`http://openview.hp.com/ecare/getsupportdoc?docid=OV-EN020917`

You must register before you can access these sites.

Coexistence – Service Desk and HP OpenView Operations

Service Desk 5.0

HP OpenView Operations for Windows 7.5 Management Server on Windows 2000 AS and Windows 2003 ES

This configuration supports the following Service Desk components:

- Service Desk 5.0 - OVO for Windows 7.5 integration.
- Service Desk 5.0 Agent (included in SD-OVO for Windows 7.5 Integration).
- Service Desk 5.0 LoadObject (included in SD-OVO for Windows 7.5 Integration).
- Service Desk 5.0 stand-alone client.

NOTE The Service Desk 5.0 management server is not supported on the same machine as the OVO for Windows 7.5 management server.

OVO for UNIX 8.x Management Server on HP-UX 11.11 PA-RISC

This configuration supports the following Service Desk components:

- Service Desk 5.0 - OVO for UNIX 8.x integration.
- Service Desk 5.0 Agent (included in SD - OVO for UNIX 8.x Integration).
- Service Desk 5.0 LoadObject (included in SD - OVO for UNIX 8.x Integration).
- Service Desk 5.0 stand-alone client.

NOTE The Service Desk 5.0 management server is not supported on the same machine as the OVO for UNIX 8.x management server.

Service Desk 5.0 Service Pack 1

Service Desk 5.0 SP1 Management Server / Stand-alone Client and OVO for Windows 7.5 Agent

The above combination is supported on the platforms listed below.

- Windows 2000 AS Server and Client.
- Windows 2003 ES Server.
- Windows XP Client.

Service Desk 5.0 SP1 Management Server / Stand-alone Client and OVO for UNIX 8.1 Agent

The above combination is supported on the platform listed below.

- Windows 2000 AS Server and Client.

For the combination SD 5.0 SP1 management server/SP1 client and OVO for UNIX 8.1 agent, two workarounds are available. For details, see the following sections:

- "Service Desk 5.0 SP1 and OVO for UNIX 8.1 on Windows 2000 AS – Unable to Start SD Management Server" on page 237.
- "Service Desk 5.0 SP1 and OVO for UNIX 8.1 on Windows 2000 AS – Conflicting or Missing Certificates" on page 238.

Service Desk 5.0 SP1 and OVO for UNIX 8.1 on Windows 2000 AS – Unable to Start SD Management Server

Problem Description

With this configuration, entering the **ovc** command returns the following messages:

```
ctrl-113) Control daemon is running...But cannot be accessed
remotely.

Please wait some more time or restart the daemon..

(ctrl-21) Communication error when executing 'Status'
method.

(xpl-118) recv() on '127.0.0.1:1733' failed.

(WIN-10054) An existing connection was forcibly closed by
the remote host.
```

Explanation

The CoreID of the managed node is overwritten by its previous value as stored in OVO for UNIX. When the Service Desk management server is installed, it generates a unique CoreID. If a new installation of the Service Desk management server is performed on the same system — for example after re-imaging the environment, or after a HDD failure and replacement — a new CoreID will be generated by this installation. When the OVO for UNIX agent is redeployed on this managed node the new CoreID is overwritten by its predecessor.

Workaround

On the OVO for UNIX management server, delete the managed node (the Service Desk management server) and add it again as a new node before redeploying the OVO for UNIX agent.

Service Desk 5.0 SP1 and OVO for UNIX 8.1 on Windows 2000 AS – Conflicting or Missing Certificates

Problem Description

When the OVO for UNIX agent is deployed on the Service Desk managed node, the `opcagt` command indicates that it is buffering its messages for the OVO for UNIX management server. If the `opcmsg` template has been deployed on the managed node, the template does not work either.

Explanation

This is caused by a conflict of certificates, or by missing certificates.

Workaround

1. Run the command **`ovcert -list`** to display an overview of your certificates.
2. If the list is empty, run the command **`ovcert -certreq`** to request new certificates, then 'grant' these certificates on the OVO for UNIX management server.

   If a list with certificates is returned, remove the top two certificates using the command **`ovcert -remove <certificate number>`**. For security purposes, first make a backup of these certificates. You do not need to remove the bottom two **`ov resource group`** (OVRG) certificates (and perhaps this may even be dangerous). Run **`ovcert -certreq`** to request new certificates and grant these certificates on the OVO for UNIX management server.

NOTE The only Service Desk functionality that uses these certificates is the Java Web Start client. As a result of the workaround described above, the Java Web Start clients connecting to the Service Desk management server will now receive their certificates from the OVO management server instead.

14 Troubleshooting Tools

This chapter contains information on the following topics:

- Support tool supplied with Service Desk.
- Descriptions of other troubleshooting tools available for use with Service Desk.

Service Desk Support Tool

The Service Desk Support Tool can be used in the event of a problem arising with Service Desk that results in a call to the Service Desk support line.

The tool collects information about the operating system and Service Desk and puts it into a Support Log file. This log file is intended to be used by support engineers to reproduce problems and to prevent miscommunication between the support team and the customer.

The support tool gathers information about the following areas:

- Operating system.
- Service Desk installation.
- JRE and its settings.
- Service Desk host system.

The tool wraps all log files (including the Support Log file) and cache files together in a WinZip file that can be used by a support engineer or sent to Service Desk support as an attachment to an email message or as a service call for analysis.

Support Log File

The tool reads settings from both the operating system and the Service Desk (client or server) installation and writes them to the Support Log file, `SupportInfo.log`.

Table 14-1 shows the settings that are included in the log file.

Table 14-1

Variable	Description
`os.name`	Operating system
`os.arch`	Processor type
`Java.vendor`	Java vendor
`Java.version`	Java version

Table 14-1 **(Continued)**

Variable	Description
`User.region`	Region of Service Desk machine
`User.language`	User language
`http.agent`	Web browser
`Java.class.path`	Java classpath
`User.timezone`	User time zone
`AppSystem.getVersionInfo`	Service Desk version
`AppSystem.getApplicationVersion`	Service Desk build version
`AppSystem.getUserFolder`	Service Desk user folder
`AppSystem.getCacheFolder`	Service Desk cache folder
`AppSystem.getTempFolder`	Service Desk temporary folder
`AppSystem.getRootFolder`	Service Desk root folder
`AppSystem.getBinaryFolder`	Service Desk bin folder
`AppSystem.getClassesFolder`	Service Desk class folder
`AppSystem.getImagesFolder`	Service Desk images folder
`AppSystem.getRepositoryFolder`	Service Desk repository folder
`AppSystem.getSoundsFolder`	Service Desk sounds folder
`file.encoding`	File encoding
`Java.vm.info`	VM mode information
`Java.home`	Java home directory
`Java.io.tempdir`	Java temp directory
`User.country`	Country where user is located

Table 14-1 (Continued)

Variable	Description
`User.home`	User’s home directory
`Dir.path(DirKey.Install)`	Service Desk install folder
`Dir.path(DirKey.DataTmp)`	Service Desk temporary data folder
`Dir.path(DirKey.Packages)`	Service Desk packages folder
`Dir.path(DirKey.DataWww)`	Service Desk internet folder
`Dir.path(DirKey.NonOV)`	Non-HP Openview folder
`Dir.path(DirKey.Resources)`	Service Desk resources folder

Support Zip File

The files which are retrieved from Service Desk are stored in a zip file in the support folder. The support folder is located in the installation directory.

The support zip file is:

`<client/server>_support_information_`*`yyyyMMdd_HHmmss`*`.zip`

The support zip file contains the support log file, and also the contents of the following directories.

Table 14-2

Log folder (client and server)	*`<Support folder>`*`/log`
Configuration folder (client and server)	*`<Support folder>`*`/config`
User data folder	*`<Support folder>`*`/userdata`
Cache folder (client and server)	*`<Support folder>`*`/cache`

The contents of these directories are copied to the subdirectories in the support folder. After the zip file has been successfully created, the files are removed from the support folder, leaving only the zip file.

Run the Support Tool

The Support Tool is packaged as a jar file, `OvSdSupporttool.jar`, and delivered with Service Desk.

- **Windows**

  `bin\OvSdSupportTool.bat`

- **UNIX**

  `bin/OvSdSupportTool.sh`

Other Service Desk Support Tools

OVII Inventory Tool

This tool shows details of all HP OpenView components installed on a specified system.

The following versions of the tool are available:

- A lightweight command-line version.
- A Java version with its own user interface.

The OVII inventory tool can be downloaded from the following location:

http://ovii.cnd.hp.com/packageViewer/ovinv_tool.html

Integrity Check Tool

`OvObsIntegrityCheck` performs an integrity check on a Service Desk management server. This tool runs automatically during the server installation process. Its use under other circumstances is not supported. If you run this tool manually, the validity of the information displayed may be compromised. For example, the information displayed may include warning messages even on a healthy system.

`OvObsIntegrityCheck` is available in the following directory on a Service Desk management server:

- **UNIX**

  `/opt/OV/bin`

- **Windows**

  `<install-dir>\bin`

Service Desk Version Tool

`ovsdversion` displays the precise version number of each Service Desk component installed on a particular machine. For example, the following information may be displayed on a system on which a Service Desk management server, metric adapters, and object loader are installed:

```
***********************************************************
The following products are installed on this system
***********************************************************
HP OpenView Metric Adapters (v5.10.028)
HP OpenView Load Object (v5.10.028)
HP OpenView Service Desk Management Server (v5.10.028)
```

`ovsdversion` is available in the following directory on a Service Desk management server:

- **UNIX**

  `/opt/OV/bin`

- **Windows**

  `<install-dir>\bin`

Component Library Version Tool

`ovsdcompversion` displays the precise version number of all Service Desk component libraries installed on a particular machine. A section of the output from this command is shown below.

```
Title  : "HP OpenView Service Desk" Common Library
Vendor : "Hewlett-Packard Development Company"
File   : sd-gui.jar (05.10.042)
Title  : "HP OpenView Service Desk" UI FrameWork extensions
Vendor : "Hewlett-Packard Development Company"
File   : sd-sdcmodel.jar (05.10.042)
Title  : "HP OpenView Service Desk" SD Config Model
Vendor : "Hewlett-Packard Development Company"
File   : sd-server.jar (05.10.042)
Title  : "HP OpenView Service Desk" Server Library
Vendor : "Hewlett-Packard Development Company"
File   : sd-upgradefiles.jar
```

```
Title  :
Vendor :
File   : ui-client.jar (02.10.036)
Title  : ui-client
Vendor :
----------------------------------------------------------
Manifest entries for folder:
C:\Program Files\HP OpenView\bin\..\java\OvShSae
----------------------------------------------------------
File   : collector-framework-2.0.jar (2.0.1)
Title  : HP OpenView Self-Healing Services Collector Framework
Vendor : Hewlett-Packard Development Company, L.P.
File   : commons-codec-1.3.jar (1.3)
Title  : Jakarta Commons Codec
Vendor : Apache Software Foundation
File   : isee-utils-2.0.jar (2.0.0)
Title  : HP OpenView Self-Healing Services Collector Framework
Vendor : Hewlett-Packard Development Company, L.P.
Vendor :
```

`ovsdcompversion` is available in the following directory on a Service Desk management server:

- **UNIX**

  `/opt/OV/bin`

- **Windows**

  `<install-dir>\bin`

A

architecture, 95

C

client
  cache folder, 242
  support log information, 240
commands
  OvSdSupporttool, 243
configuration diagrams
  local dynamic tracing options, 120
  local static tracing options, 121
configuring tracing, 112

D

dynamic tracing
  establishing local, 121
  local options, 120

I

installation, 110
  support information, 240
  trace server on UNIX, 110
  trace server on Windows, 111

J

Java
  class path, 241
  home, 241
  VM mode information, 241

L

language
  user, 241
local tracing options
  dynamic, 120
  static, 121
log files
  support tool, 240
logging
  about, 67

O

OvSdSupporttool, 243
ovtrcadm, 96
ovtrccfg, 97
ovtrcgui, 95
ovtrcmon, 97

S

server
  cache folder, 242
  configuration folder, 242
  log folder, 242
setting up tracing, 112
static tracing
  enabling, 124
  establishing local, 123
  local options, 121
support tool
  executing, 243
  log file, 240
  OvSDSupporttool command, 243
  zip file, 242

T

tools
  support, 239
trace
  configuration files, 97
    application, 98
    sink, 99
    syntax version, 98
    trace, 101
trace server, 96
  installation on UNIX, 110
  installation on Windows, 111
  log file, 103
trace-enabled application, 95
tracing
  clients, 95
  dynamic, 104
    enabling, 127
    establishing local, 121
    local options, 120
  guidelines, 130
  how to, 106
  setting up, 112
  static, 104
    enabling, 124
    establishing local, 123
    local options, 121
tracing tool
  installation, 110
  installation on UNIX, 110
  installation on Windows, 111
  overview, 96
troubleshooting

logging, 67

Z

zip file, support tool, 242

invent

Java™
COMPATIBLE