OpenView
Service Quality Manager

Action Executor for Simple Network Management Protocol V2

Installation, Configuration and User's Guide

Edition: 1.4

December 2006

Legal Notices

Warranty

Hewlett-Packard makes no warranty of any kind with regard to this manual, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Hewlett-Packard shall not be held liable for errors contained herein or direct, indirect, special, incidental or consequential damages in connection with the furnishing, performance, or use of this material.

A copy of the specific warranty terms applicable to your Hewlett-Packard product can be obtained from your local Sales and Service Office.

Restricted Rights Legend

Use, duplication or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause in DFARS 252.227-7013.

Hewlett-Packard Company

United States of America

Rights for non-DOD U.S. Government Departments and Agencies are as set forth in FAR 52.227-19(c)(1,2).

Copyright Notices

Trademark Notices

Adobe® and Acrobat® are trademarks of Adobe Systems Incorporated.

HP-UX Release 10.20 and later and HP-UX Release 11.00 and later (in both 32 and 64-bit configurations) on all HP 9000 computers are Open Group UNIX 95 branded products.

Java™ and all Java based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries.

Microsoft® is a U.S. registered trademark of Microsoft Corporation.

Netscape is a U.S. trademark of Netscape Communications Corporation.

NMOS™ is a trademark of RiverSoft Technologies Limited.

Oracle® is a registered U.S. trademark of Oracle Corporation, Redwood City, California.

Oracle7™ and Oracle7 Server™ are trademarks of Oracle Corporation, Redwood City, California.

PostScript® is a trademark of Adobe Systems Incorporated.

Riversoft™ is a trademark of RiverSoft Technologies Limited.

UNIX® is a registered trademark of The Open Group.

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

X/Open® is a registered trademark, and the X device is a trademark of X/Open Company Ltd. in the UK and other countries.

All other product names are the property of their respective trademark or service mark holders and are hereby acknowledged.

Contents

Preface

This document describes how to install and configure the HP OpenView Service Quality Manager (SQM) SNMP Action Executor. It provides an overview of the SNMP Action Executor and describes how to:

- Install the SNMP Action Executor
- Set up the SNMP Action Executor
- Start and stop the SNMP Action Executor.

Intended Audience

This document is intended for OpenView SQM administrators and operators.

Required Knowledge

It is assumed that the reader is familiar with the functionality of OpenView SQM and has previous experience of the following:

- System administration and operations
- Service level management.

It is assumed that the reader is familiar with the concepts described in the following books:

- *HP OpenView Service Quality Manager Overview.*
- *HP OpenView Service Quality Manager Administration Guide.*
- *HP OpenView Service Quality Manager Information Modeling Reference Guide.*

Software Versions

The software versions referred to in this document are specified in Section 3.1. Software and Hardware Requirements.

Typographical Conventions

The following typographical conventions are used in this book:

Bold	New terms and to emphasize important words.
Italic	File names, programs, and parameters. The names of other documents referenced in this manual.
`Monospace`	Source code and examples of file contents. Commands that you enter on the screen. Pathnames. Keyboard key names.

Associated Documents

For a full list of OpenView SQM user documentation, refer to the *HP OpenView Service Quality Manager Product Family Introduction*.

Support

Please visit our HP OpenView web site at:

http://openview.hp.com/

There you will find contact information as well as details about the products, services, and support HP OpenView has to offer.

The HP OpenView support area of the HP OpenView web site includes:

- Downloadable documentation
- Troubleshooting information
- Patches and updates
- Problem reporting
- Training information
- Support program information.

Chapter 1

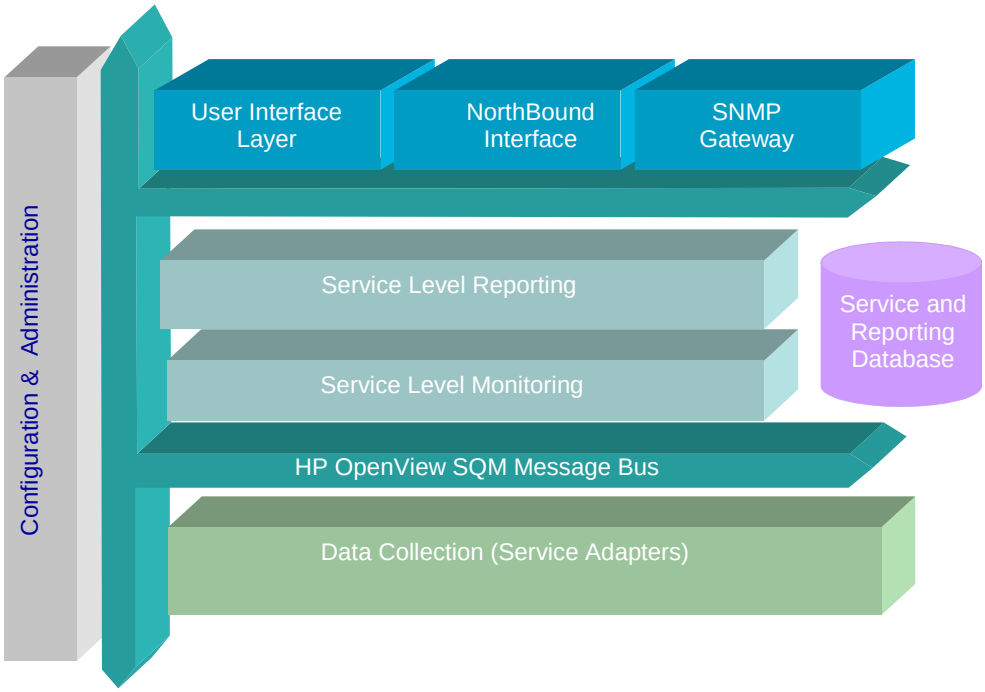
Introduction

1.1 OpenView Service Quality Manager

OpenView SQM provides a complete service quality management solution. It consolidates quality indicators across all domains — telecom, IT networks, servers, and applications — providing end-to-end visibility on service quality. OpenView SQM links service quality degradations to potential effects on business, allowing network support personnel to address problems and prioritize actions proactively.

OpenView SQM monitors the service quality by aggregating information coming from all data sources, such as the network, the IT infrastructure, and the service provider's business processes. Using this information, service operators can pinpoint infrastructure problems and identify their potential affect on customers, services, and Service Level Agreements (SLAs).

Figure 1 OpenView SQM Main Components



For a detailed description of OpenView SQM, refer to the *HP OpenView Service Quality Manager Overview*.

1.2 The SNMP Action Executor

The goal of the SQM SNMP Action Executor is to send SNMP traps (V2c) when an SLA is degraded or violated.

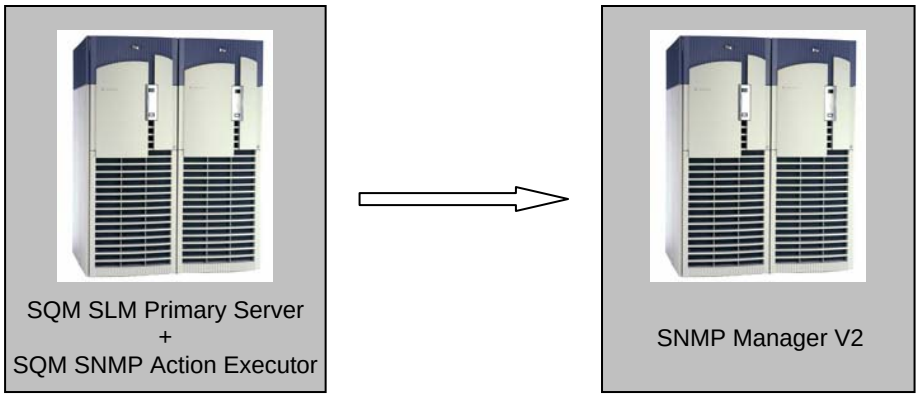
The SNMP Action Executor acts as a bridge between OpenView SQM and a SNMP Manager V2, collecting data from OpenView SQM and sending SNMP traps.

Once SQM and the SNMP Action Executor are installed and configured, the SNMP Action Executor can start to collect the specified service degradation and violation events from OpenView SQM and transform those events into SNMP Traps.

These traps are then available for further processing by any SNMP V2 managers. They can then be monitored to follow-up violation and degradation states.

1.2.1 Architecture Overview

The recommended architecture is to have the SQM SNMP Action Executor running on the SQM SLM Primary Server.



However, it is possible to install the SQM SNMP Action Executor on any host.

Chapter 2

What does the SQM SNMP Action Executor Perform?

The SQM SNMP Action Executor translates, in real-time, SLA degradations, violations, and compliance SQM events into SNMP traps.

The SQM SNMP Action Executor is responsible for mapping the following SQM events into SNMP traps:

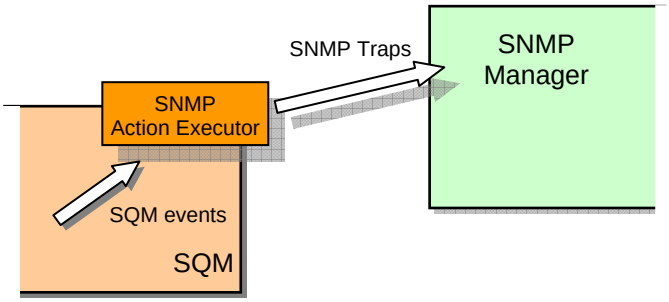
- Parameter Threshold Crossing events
- Service Objective Status Changed events (for SLA, Service Instance or Service Component Instance)
- Compliance Parameter Threshold Crossing events
- Compliance Service Objective Status Changed events (for SLA, Service Instance or Service Component Instance).

The SQM SNMP Action Executor is pass-through, as SNMP traps are forwarded in real-time, generated from the mapping of OpenView SQM degradation and violation events.

It is possible to configure the SQM SNMP Action Executor to indicate which type of SQM events have to be mapped into SNMP Traps (see Section 2.2 Associating SNMP Action Executor on Objective Status Changed events).

These traps are sent to an SNMP Manager. This SNMP Manager is configurable in the Central Repository. Please refer to Section 4.3.1.7 for more information on this configuration field.

Figure 2 Involved components



2.1 Associating SNMP Action Executor to Parameter Threshold Crossing Events

The SNMP Action Executor only processes the SQM Parameter Threshold Crossing event that has the **SNMPTrap** associated action. The associated action is defined at the Service Level Objective (SLO) definition by the Action Executor field.

A Service Level Objective aims at monitoring a Service Parameter, comparing the parameter value to one or several thresholds. Therefore a status is computed for the parameter's parent Service Component Instance, Service Instance and SLA.

Through Action Executor, the SLO definition can indicate actions to trigger when a parameter crosses a threshold. For more information, please refer to the chapter explaining the *'Service Level'* concept in the *HP OpenView Service Quality Manager Information Modeling Reference Guide*.

2.1.1 Associating SNMP Action Executor to a SLO

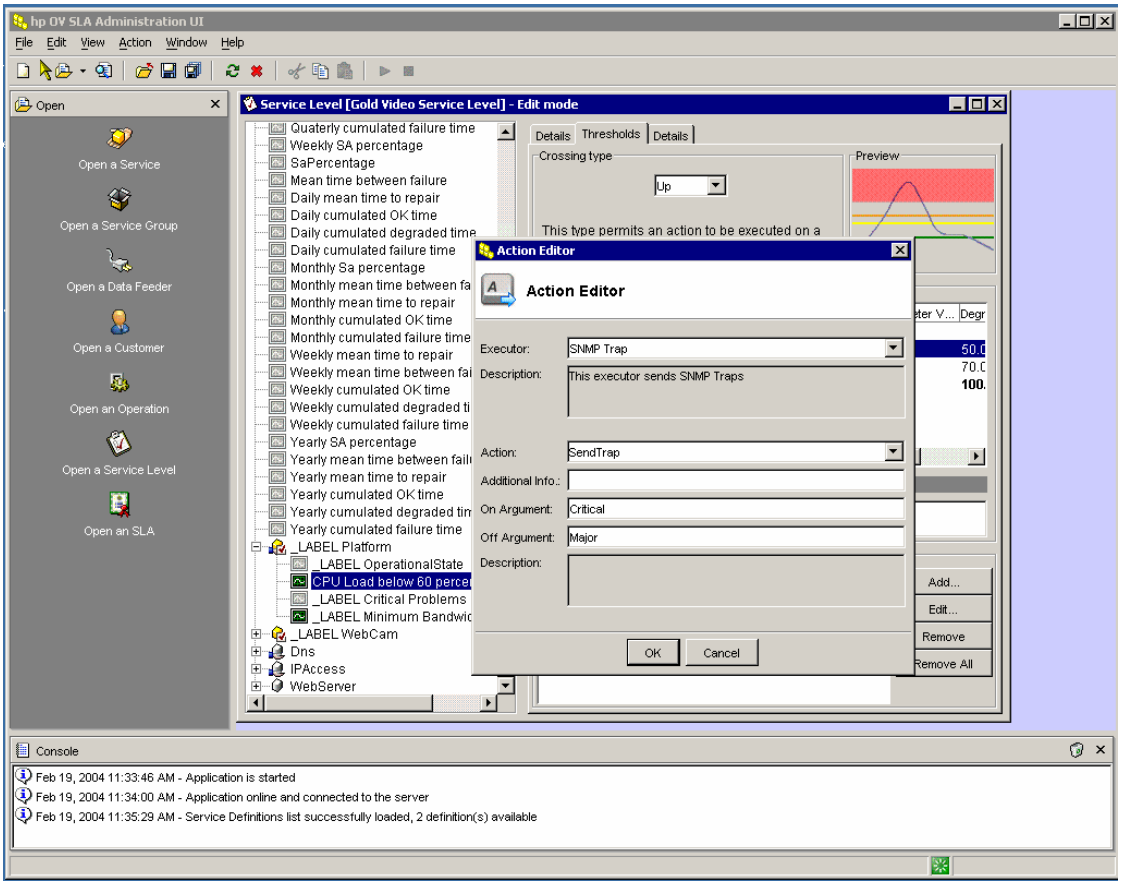
To send SNMP traps when a parameter crosses a SLO threshold, the SLO definition must have the following action executor:

- Executor: '**SNMPTrap'**
- Action: '**SendTrap'**
- The **On Argument** and **Off Argument** provide the information of the emitted SNMP trap when the parameter threshold is crossed. See the following chapter, which explains how these two arguments are used.

Important

The Executor, Action, On Argument and Off Argument fields are case sensitive.

Figure 3 Defining a threshold with Action Executor in the SLA Administration UI



2.1.2 SNMP Trap on Parameter Threshold Crossing

An SNMP trap on Parameter Threshold Crossing is generated when a Service Level Objective associated to a monitored parameter is crossed.

A violation threshold for a Customer SLA type is mapped to a *customerSLAViolationStart* trap. For an operation SLA, the emitted trap is an *operationSLAViolationStart* trap.

For a Customer SLA type, a degradation threshold is mapped to a *customerSLADegradationStart* trap. For an operational SLA, the emitted trap is an *operationSLADegradationStart* trap.

When a parameter is no longer violated, the SNMP Action Executor issues a *customerSLAViolationEnd* trap for a Customer SLA type or an *operationalSLAViolationEnd* for an Operation SLA.

When a parameter is no longer degraded, the SNMP Action Executor issues a *customerSLADegradationEnd* trap for a Customer SLA type or an *operationalSLADegradationEnd* for an Operation SLA.

For a compliance parameter, the same mechanism as described above is applied, but the trap names are:

- *customerComplianceDegradationStart*
- *customerComplianceDegradationEnd*
- *customerComplianceViolationStart*
- *customerComplianceViolationEnd.*

Please refer to Appendix F *'SQM SNMP Action Executor MIB'* for more information about these traps.

SNMP Trap Fields Mapping

The following table explains the mapping between the SQM events fields and the SNMP trap fields for Parameter Threshold Crossing.

Table 1 SNMP Trap Fields Mapping

SQM Objects	SNMP Trap Fields
Customer label	entityLabel
Customer name	entityName
Type (Customer, Operation)	entityType
Parameter Value	measuredParamValue
-	rootCause[1]
Component Definition label	sciDefLabel
Component Definition name	sciDefName
Component label	sciLabel
Component name	sciName
Service or Component flag (Service, Component)	sciType
Service Definition label	siDefLabel
Service Definition name	siDefName
Service label	siLabel
Service name	siName
SLA label	slaLabel
SLA name	slaName
SLA Monitoring type (instance, view)	slaMonitoringType
Service Level label	slaServiceLevelLabel
Service Level name	slaServiceLevelName
Component Service Level label	sloComponentServiceLevelLabel
Component Service Level name	sloComponentServiceLevelName
Crossing Type (valued, equal, notEqual, up, down)	sloCrossingType
Service Level Objective label	sloLabel
Service Level Objective name	sloName
Parameter label	sloMntrdParamLabel
Parameter name	sloMntrdParamName
Parameter datatype	sloMntrdParamType
Threshold Action name	sloThreshActionName

[1] True when Root Alarm

SQM Objects	SNMP Trap Fields
Objective Threshold label	sloThreshLabel
Objective Threshold name	sloThreshName
Threshold Reference Value	sloThreshReferenceValue
Threshold Action**.on** or **.off** field	thresholdActionArg
Acquisition Timestamp	timeStamp

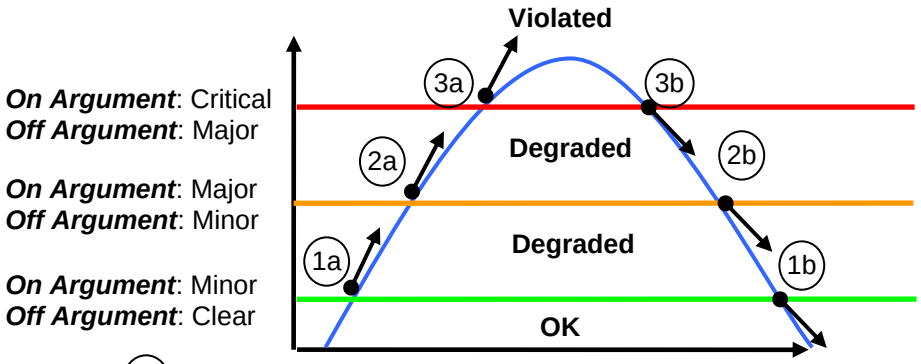
The *thresholdActionArg* field is retrieved in the corresponding SQM event. It is given by the *'On Argument'* or *'Off Argument'* optional attributes of the crossed threshold action executor.

If the threshold crossing corresponds to a Quality of Service decreasing, the *thresholdActionArg* is filled with the *'On Argument (action.on)'* attribute.

If the threshold crossing corresponds to a Quality of Service increasing, the *thresholdActionArg* is filled with the *'Off Argument' (action.off)* attribute.

The ***'On Argument'*** or ***'Off Argument'*** optional attributes can be any action that the trap listener can understand, and interpret. For example they can be used to give a severity notion to a threshold crossing.

Figure 4 Threshold Crossing SNMP Traps Severity



1a QoS is decreasing → *thresholdActionArg* is Minor (On Argument)

1b QoS is increasing → *thresholdActionArg* is Clear (Off Argument)

2a QoS is decreasing → *thresholdActionArg* is Major (On Argument)

2b QoS is increasing → *thresholdActionArg* is Minor (Off Argument)

3a QoS is decreasing → *thresholdActionArg* is Critical (On Argument)

3b QoS is increasing → *thresholdActionArg* is Major (Off Argument)

2.1.3 Root Trap

The Parameter Threshold Crossing events may impact several customers or operation SLAs.

A SNMP Trap called the Root Trap is generated if the event impacts several customers or SLAs and if the Component Instance impacted is a shared component between Services Instances.

This SNMP Trap keeps track of the root of the degradation or violation, and includes the number of impacted SLAs.

Note

If only one SLA/Customer is impacted, only one trap is generated.

SNMP Trap Fields Mapping

The following table explains the mapping between the SQM events fields and the SNMP trap fields for the root alarm

Table 2 SNMP Trap Fields Mapping

SQM Object	SNMP Trap Fields
Number of impacted SLA	impactedSLANumber[2]
ParameterValue	measuredParamValue
-	rootCause 3
Component Definition label	sciDefLabel
Component Definition name	sciDefName
Component label	sciLabel
Component name	sciName
Service or Component flag (Service, Component)	sciType
SLA Monitoring type (instance, view)	slaMonitoringTyp
Component Service Level label	sloComponentServiceLevelLabel
Component Service Level name	sloComponentServiceLevelName
Crossing Type (valued, equal, notEqual, up, down)	sloCrossingType
Service Level Objective label	sloLabel
Service Level Objective name	sloName
Parameter label	sloMntrdParamLabel
Parameter name	sloMntrdParamName
Parameter datatype	sloMntrdParamType
Threshold Action name	sloThreshActionName
Objective Threshold label	sloThreshLabel
Objective Threshold name	sloThreshName
Threshold Reference Value	sloThreshReferenceValue
Threshold Action**.on** or **.off** field	thresholdActionArg
Acquisition Timestamp	timeStamp

[2] Only for Root Alarm
[3] True when Root Alarm

2.1.4 Fan-Out Processing

The SNMP Action Executor generates as many SNMP Traps on service, component and parameter as there are impacted customers and operation SLAs, along with the related root trap.

This processing is controlled by the fan-out local field in the Central Repository (please refer to Section 4.3.1.9 FanOut (TrapMapping) for a description of this field), and is called the alarming fan-out.

When the alarming fan-out feature is disabled, only the root trap is generated.

Note
By default, the Fan-Out processing is enabled

2.2 Associating SNMP Action Executor on Objective Status Changed events

The association of a SNMP Action Executor with an Objective Status Change event depends on the gateway 'ObjectiveStatusFiltering' configuration flag (see chapter 4.3.1 - General Configuration Properties).

2.2.1 Association when 'ObjectiveStatusFiltering' is enabled

This is the default SNMP Action Executor configuration.

When the Objective Status filtering is enabled, the association of the Action Executor with the Objective Status Changes is declared in the Filter Configuration XML file.

The Filter Configuration file contains two trigger: a **'SLA Status trigger'** and a **'SI Status trigger'**.

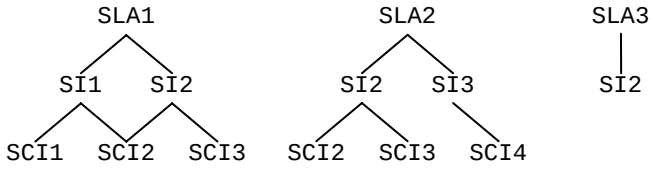
Those triggers are '*regular expressions*' used by the filtering mechanism to determine if an Objective Status Change event matches the Action Executor. When this is the case, the action is executed.

The following table explains how the Action Executor behaves according the SLA and SI triggers:

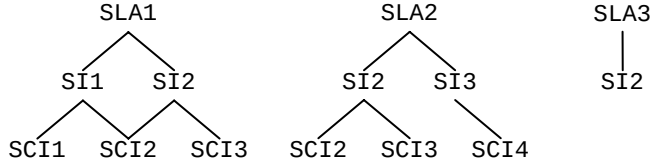
SLA Status trigger	SI Status trigger	Triggered events
-	-	Nothing
-	RegExp Y	Configuration error (no OS Changes will matches)
RegExp X	-	Only OS Changes of SLA matching RegExp X
RegExp X	RegExp Y	OS Changes of SLA matching RegExp X + in these SLA, OS Changes of SI matching RegExp Y + in these SI, OS Changes of all child SCI.

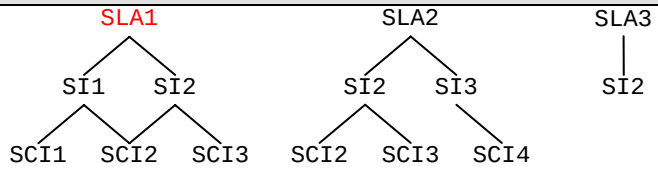
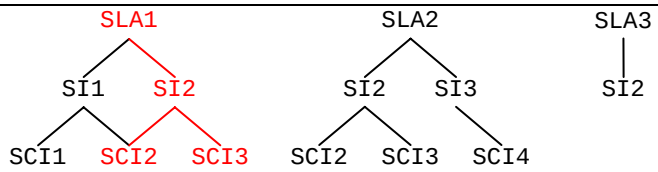
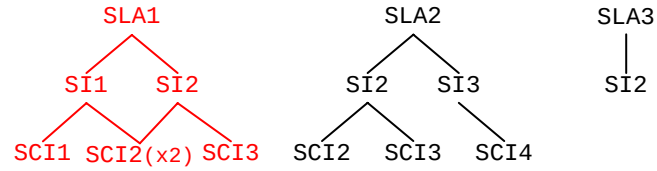
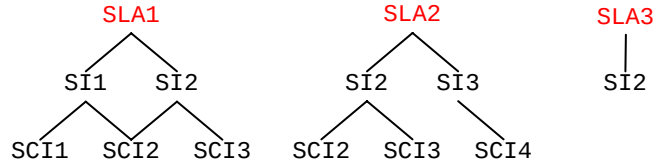
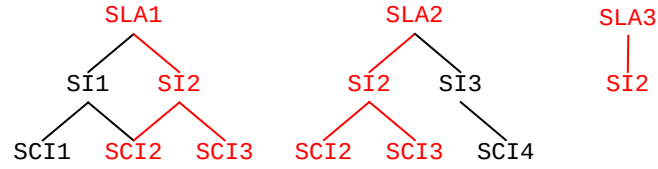
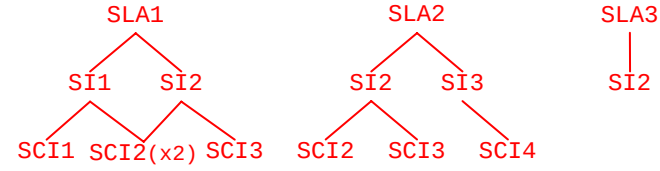
Example

Have the following Objective Status Change trees:



And the following script configurations:

SLA Status	SI Status trigger	Triggered Objective Status Change events (in red)
-	-	

SLA Status	SI Status trigger	Triggered Objective Status Change events (in red)
SLA1	-	
SLA1	SI2	
SLA1	SI.*	
SLA.*	-	
SLA.*	SI2	
SLA.*	SI.*	

Note

Unlike the triggering on Threshold Crossed events, the Objective Status filtering mechanism does not use the event '*ActionExecutorName*' to determine if a trap has to be sent.

2.2.2 Association when 'ObjectiveStatusFiltering' is disabled

When the Objective Status filtering is disabled, the traps are emitted only when the Objective Status Change events are due to Parameter Threshold Crossing on SLO associated with the SNMP Action Executor (see section 2.1.1 - Associating SNMP Action Executor to a SLO).

Note

In this mode, to be sure to send a trap on a component (SLA or SI) Objective Status Change, every parameter monitored under this component has to be associated with the SNMP Action Executor.

2.2.3 SNMP Traps on Service Objective Status Changed

SNMP Trap on Service Objective Status Changed is generated when an SLA, Service Instance (SI) or Service Component Instance (SCI) status has changed.

The SNMP trap type generated is based on the value of the Service Objective Status, as reported by OpenView SQM.

At platform level, SQM defines two global fields:

- SCStateDegradationLevel set to 0.8 by default.
- SCStateViolationLevel set to 0.2 by default.

Objective Status levels are defined in the Central Repository and based on these ranges, the trap type is as indicated in the table below:

Table 3 Trap Type for Service Objective Status

Value Range	Trap Type for Customer SLA	Trap Type for Operation SLA.
0 – 0.2	CustomerServiceObjectiveViolation	OperationServiceObjectiveViolation
0.2 – 0.8	CustomerServiceObjectiveDegradation	OperationServiceObjectiveDegradation
0.8 – 1	CustomerServiceObjectiveNormal	OperationServiceObjectiveNormal

Please refer to Section 4.3.1 General Configuration Properties for the Objective Status level definition in the Central Repository.

Refer to the following configuration fields:

- ObjectiveStatusViolationLevel (TrapMapping) set to the global field SCStateDegradationLevel by default.
- ObjectiveStatusDegradationLevel (TrapMapping) set to the global field SCStateViolationLevel by default

For the Compliance Service Objective Status, the following traps are emitted:

Table 4 Trap Type for Compliance Service Objective Status

Value Range	Compliance Trap Type for Customer SLA
0 – 0.2	CustomerComplianceObjectiveViolation
0.2 – 0.8	CustomerComplianceObjectiveDegradation
0.8 – 1	CustomerComplianceObjectiveNormal

Please refer to Section 4.3.1 General Configuration Properties for the Compliance Objective Status level definition in the Central Repository. Refer to the following configuration fields:

- ObjectiveStatusViolationLevel (ComplianceTrapMapping) set to the global field SCStateDegradationLevel by default.
- ObjectiveStatusDegradationLevel (ComplianceTrapMapping) set to the global field SCStateViolationLevel by default

SNMP Trap Fields Mapping

The following table explains the mapping between the SQM events fields and the SNMP trap fields.

Table 5 SNMP Trap Fields Mapping

SQM Objects	SNMP Trap Fields
Component Objective Status	currentStatus
SLA Objective Status Value	currentStatus
Previous Component Objective Status Value	previousStatus
Previous SLA Objective Status Value	previousStatus
Customer label	entityLabel
Customer name	entityName
Type (Customer, Operation)	entityType
Component Definition label	sciDefLabel
Component Definition name	sciDefName
Component label	sciLabel
Component name	sciName
Service or Component flag (Service, Component)	sciType
Service Definition label	siDefLabel
Service Definition name	siDefName
Service label	siLabel
Service name	siName
SLA label	slaLabel
SLA name	slaName
SLA Monitoring type (instance, view)	slaMonitoringType
Acquisition Timestamp	timeStamp

Please refer to Appendix F *'SQM SNMP Action Executor MIB'* for more information about theses traps.

2.3 SNMP Trap on SQM Service Compliance Event

The SNMP Action Executor performs a similar mapping for the OpenView SQM compliance events. The Compliance SNMP trap is mapped from OpenView SQM compliance degradation and violation events.

The processing of Compliance events in the SNMP Action Executor can be disabled. An SNMP Action Executor local field called «*ComplianceEnabled*» in the Central Repository controls this processing. Please refer to Section 4.3.1.3 ComplianceEnabled for more details.

Note

By default, compliance events are not processed.

Compliance SNMP Traps can be reported on Parameter Threshold crossing and on Service Objective Status changes. By default, SNMP Traps are reported only on Parameter Threshold crossing. An SNMP Action Executor local field in the Central Repository called «*ComplianceMapping*» controls this processing. Please refer to Section 4.3.1.2 - ComplianceMapping for more details.

Chapter 3

SNMP Action Executor Installation

This chapter describes how to install the SQM SNMP Action Executor on an HP system running HP-UX V11i. After you have completed the installation, follow the instructions in Chapter 4 to configure the adapter.

3.1 Software and Hardware Requirements

For software and hardware requirements, refer to the *HP OpenView Service Quality Manager Installation Guide*.

3.2 Installing the Software

This section describes how to install the SQM SNMP Action Executor on the SQM SLM Primary Server (recommended architecture).

This section assumes that the *OV SQM Kernel* is already installed and configured on the primary host. The SQM Kernel installation and configuration are described in the *HP OpenView Service Quality Manager Installation Guide*.

3.2.1 Required environment

To install (or configure) the SQM SNMP Action Executor, you first have to set the SQM environment.

To set the SQM environment, you have to source the *temip_sc_env.sh* file located under the SQM data directory *($TEMIP_SC_VAR_HOME)*.

Example:

```
# . /var/opt/OV/SQM/slmv12/temip_sc_env.sh
```

Note

The temip_sc_env.sh script is created during the SQM Kernel setup.

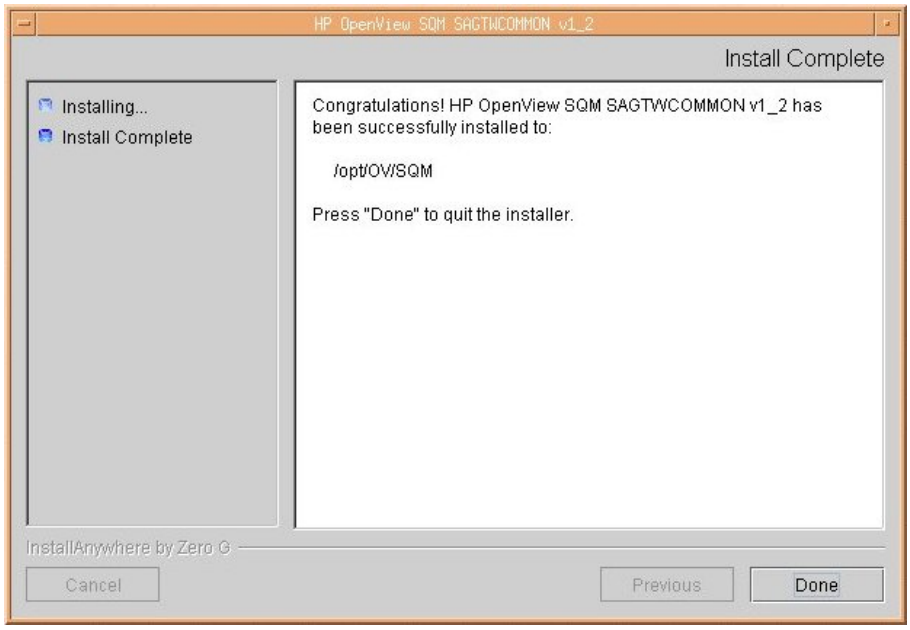
3.2.2 Installing the SQM SNMP Action Executor

To install the SNMP Action Executor, you need to have installed the Service Adapters and Gateways Common subset.

To install it (if not already installed), perform the following steps:

1. Log on as root user.
2. Mount the Service Adapters and Gateways CD-ROM on your system.
3. Go to the SQM-1.20.00-SAGTW/HPUX directory.

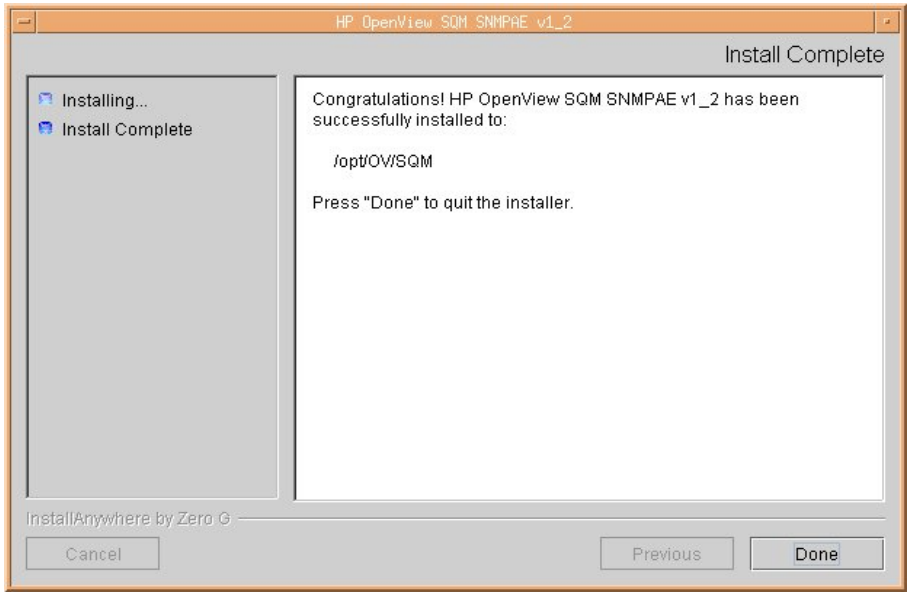
4. Run the **SQMSAGTWCOMMON-1.20.00bin** installer.



5. To end the installation process, click **Done.**

To install the SNMP Action Executor, perform the following steps:

1. Log on as root user.
2. Mount the Service Adapters and Gateways CD-ROM on your system.
3. Go to the SQM-1.20.00-SAGTW/HPUX directory.
4. Run the **SQMSNMPAE-1.20.00.bin** installer.



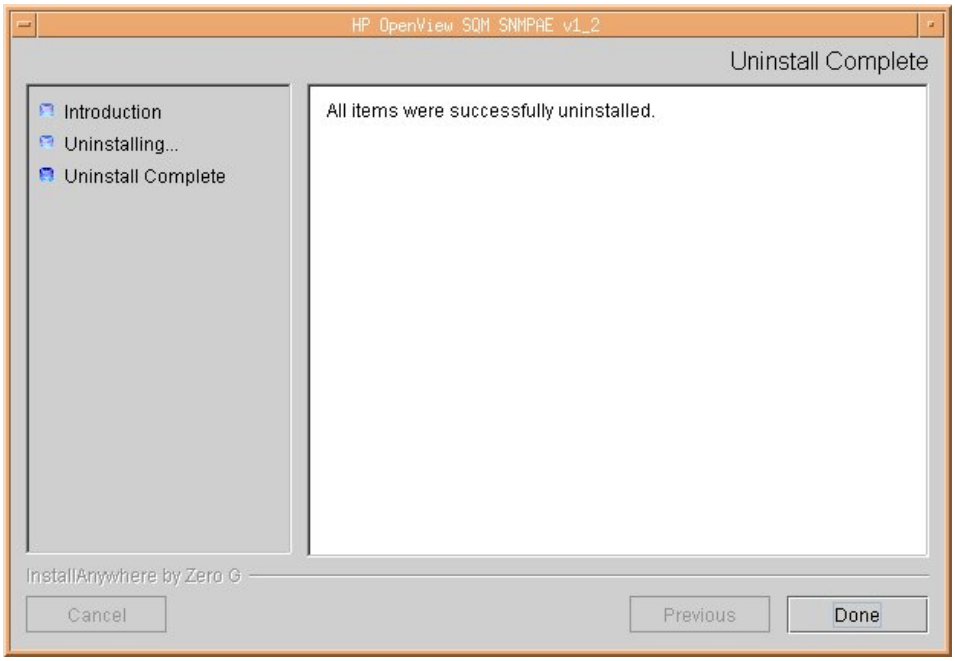
5. To end the installation, click **Done.**

3.3 Uninstalling the Software

To uninstall the SNMP Action Executor, you have to make sure that the SNMP Action Executor Gateway application **is not running**.

To uninstall the subset **SQMSNMPAE-1.20.00**, perform the following commands:

1. Log on as root user and load the SQM environment variables (see Section 3.2.1).
2. Go to the directory `$TEMIP_SC_HOME`
3. Run the uninstaller:
   `./Gateways/SNMP/v1_2/Uninstaller_SNMPAE` **`Uninstall_SNMPAE`**
4. The uninstaller window appears, click **Uninstall**.



5. Click **Done** to accept.

Chapter 4

SNMP Action Executor Setup and Configuration

This chapter describes how to set up and configure the SNMP Action Executor.

4.1 Required Environment

It is important that you ensure that the OpenView SQM environment variables are always set. Go to the $TEMIP_SC_VAR_HOME directory (where $TEMIP_SC_VAR_HOME is the OpenView SQM data directory) and run the `temip_sc_env.sh` file. This sets all OpenView SQM environment variables.

4.2 Creating SNMP Action Executor Application Instance

Once installed, you have to create the SNMP Action Executor application.

To create the SNMP Action Executor application, perform the following steps:

1. Log on as root
2. Go to the `$TEMIP_SC_HOME/setup/bin` directory
3. Invoke the **temip_sc_setup -addOn SNMPGateway_v1_2_addOn.tmpl_cfg -appliName <your SNMP Gateway application name>** command.

   Example:

   ```
   #temip_sc_setup -addOn SNMPGateway_v1_2_addOn.tmpl_cfg
    -appliName SNMPGateway
   ```

Once you have completed this procedure, the SNMP Action Executor application is created in the current OpenView SQM platform. This application belongs to the **gateway** director and it has the specified name (the name behind the *-appliName* option).

It is possible to choose a different director name using the **-dirName <director name>** option.

Example:

```
#temip_sc_setup -addOn SNMPGateway_v1_2_addOn.tmpl_cfg
 -appliName SNMPGateway -dirName myGtwDirector
```

To check that the application has been correctly created you can perform the command: **temip_sc_show_platform –platform slmv12 –director gateway** (using the *sqmadm* user with all OpenView SQM environment variables set).

For more information, refer to the *HP OpenView Service Quality Manager Administration Guide*.

4.3 Configuring the SNMP Action Executor Component

The configuration of the SNMP Action Executor Application is stored in the SQM central repository (TIBCO Repository). They are accessible from the TIBCO Designer.

Refer to the *HP OpenView Service Quality Manager Administration Guide* on how to launch and to use the TIBCO Designer.

In the TIBCO Repository, the SNMP Action Executor configuration is stored at the following locations:

- `/screpos/ServiceCenter/Gateways/SNMP/v1_2/<ApplicationName>`
- `/screpos/ServiceCenter/Gateways/SNMP/v1_2/<ApplicationName>_config`

The application properties are stored in the **Extended Properties** of the `<ApplicationName>_config` adapter.

The logging and tracing properties are stored in the **trace_sink** and **log_sink** of the `/<ApplicationName>` adapter (`.../<ApplicationName>/Advanced/Log Sinks`).

One property must particularly be checked: SNMPManager.

4.3.1 General Configuration Properties

The SNMP Action Executor general properties are stored in the **Extended Properties** of the <InstanceName>_config adapter.

The table below summarizes all the SNMP Action Executor general properties.

Table 6 SNMP Action Executor General Properties

Variable Name	**Default Values**
MessageMapping	Both
ComplianceMapping	Threshold
ComplianceEnabled	False
ActionExecutorName	SNMPTrap
ObjectiveStatusFiltering	True
SNMPCommunityString	public
SNMPManager	localhost
SNMPPort	162
TrapMapping	
-> FanOut	True
-> ObjectiveStatusViolationLevel	%%SCStateViolationLevel%%

Variable Name	Default Values
-> ObjectiveStatusDegradationLevel	%%SCStateDegradationLevel%%
ComplianceTrapMapping	
-> FanOut	True
-> ObjectiveStatusViolationLevel	%%SCStateViolationLevel%%
-> ObjectiveStatusDegradationLevel	%%SCStateDegradationLevel%%

For each previously described parameter, a detailed description is given below.

4.3.1.1 MessageMapping

This field controls the creation of SNMP Traps.

The possible values are:

- **Threshold**: SNMP traps are generated only on threshold crossing violation or degradation events.
- **Status**: SNMP traps are generated only on service objective degradation or violation events.
- **Both**: in this case, SNMP traps are generated for both threshold crossing violation or degradation events and for service objective degradation or violation.

The associated TIBCO configuration field is named *MessageMapping* and its default value is *Both*.

4.3.1.2 ComplianceMapping

This field controls the creation of SNMP traps for Compliance events.

The possible values are:

- **Threshold**: SNMP traps are generated only on compliance threshold crossing violation or degradation events.
- **Status**: SNMP traps are generated only on compliance service objective degradation or violation events.
- **Both**: in this case, SNMP traps are generated for both compliance threshold crossing violation or degradation events and for compliance service objective degradation or violation events.

The associated TIBCO configuration field is named *ComplianceMapping* and its default value is *Threshold*.

4.3.1.3 ComplianceEnabled

This field is used to specify if the SNMP Action Executor processes the compliance events or not.

The associated TIBCO configuration field is named *ComplianceEnabled* and its default value is *false*.

4.3.1.4 ActionExecutorName

Only OpenView SQM degradation and violation events whose action executor name is equal to this field are converted into SNMP traps. This is defined in the SLA Admin UI when creating service levels. See the *OpenView SQM Overview* for more information.

For the SNMP Action Executor, this field default value is: *SNMPTrap*

The associated TIBCO configuration field is named *ActionExecutorName*.

Important

The *ActionExecutorName* is case sensitive and it must respect the following rules:
- have a maximum length of 16 characters,
- start with a letter (capitalized or not),
- be only composed of letters (capitalized or not), digit and "_" character.

4.3.1.5 ObjectiveStatusFiltering

This field is used to specify if the "advanced filtering" mechanism is enabled or not.

- If the mechanism is enabled, the Gateway does not use the 'ActionExecutorName' to take into account or not the Objective Status Change events. This is delegate to a filtering mechanism that analyzes all the SQM Objective Status Change events.

  The Script Gateway filtering mechanism is described in section 2.2.1 – 'Association when 'ObjectiveStatusFiltering' is enabled'.

- If this mechanism is disabled, the Gateway uses the 'ActionExecutorName' to take into account or not the Objective Status Change events. In that case, the default status change action is performed (see section 2.2.2 - Association when 'ObjectiveStatusFiltering' is disabled).

The associated TIBCO configuration field is named *'ObjectiveStatusFiltering'* and its default value is *'True'*.

4.3.1.6 SNMPCommunityString

This field is used to specify the community string that is sent along with all SNMP traps.

The associated TIBCO configuration field is named *SNMPCommunityString* and its default value is *public*.

4.3.1.7 SNMPManager

This field is used to specify the SNMP Manager IP address.

The associated TIBCO configuration field is named *SNMPManager* and the default value is *localhost*.

4.3.1.8 SNMPPort

This field is used to specify the port number to use to send SNMP traps.

The associated TIBCO configuration field is named *SNMPPort* and the default value is *162* (the default SNMP port).

4.3.1.9 FanOut (TrapMapping)

This field is used to specify if the alarm fanning out feature is enabled or not. If this feature is enabled, the SNMP Action Executor generates one trap per impacted SLA/Service Instance/Service Component Instance in case of a shared Service Component Instance.

The associated TIBCO configuration field is named *FanOut* and its default value is *true*.

4.3.1.10 ObjectiveStatusViolationLevel (TrapMapping)

This field defines the threshold level where a service objective status is considered violated.

The associated TIBCO configuration field is named *ObjectiveStatusViolationLevel* and its default value is the global OpenView SQM field *SCStateViolationLevel* with a default of 0.2.

4.3.1.11 ObjectiveStatusDegradationLevel (TrapMapping)

This field defines the threshold where a service objective status is considered degraded.

The associated TIBCO configuration field is named *ObjectiveStatusDegradationLevel* and its default value is the global OpenView SQM field *SCStateDegradationLevel* with a default of 0.8.

4.3.1.12 FanOut (ComplianceTrapMapping)

This field is used to specify if the compliance alarm fanning out feature is enabled or not. If this feature is enabled, the SNMP Action Executor generates one compliance trap per impacted SLA/Service Instance/Service Component Instance in case of a shared Service Component Instance.

The associated TIBCO configuration field is named *FanOut* and its default value is *true*.

4.3.1.13 ObjectiveStatusViolationLevel (ComplianceTrapMapping)

This field defines the threshold where a compliance service objective status is considered violated.

The associated TIBCO configuration field is named *ObjectiveStatusViolationLevel* and its default value is the global OpenView SQM field *SCStateViolationLevel* with a default of 0.2.

4.3.1.14 ObjectiveStatusDegradationLevel (ComplianceTrapMapping)

This field defines the threshold where a compliance service objective status is considered degraded. The associated TIBCO configuration field is named *ObjectiveStatusDegradlationLevel* and its default value is the global OpenView SQM field *SCStateDegradationLevel* with a default of 0.8.

4.3.2 Advanced Configuration Fields

The SNMP Action Executor advanced properties are stored in the **Extended Properties** of the `<ApplicationName>_config` adapter.

The table below summarizes all the SNMP Action Executor advanced properties.

Table 7 SNMP Action Executor Advance Properties

Variable Name	Default Values
ListenerProcessorThreadNb	1
XMLValidationMode	%% SCXmlMsgValidation %%[4]
QuietMode	false
SNMPMaxPacketSize	2000

[4] The variables starting and ending with "%%" refer to TIBCO global variables. Refer to the *hp OpenView Service Quality Manager Administration Guide* for their values.

For each previously described parameter, a detailed description is given below.

4.3.2.1 Listener Processor Thread Number

This field defines the maximum number of concurrent listener threads that may have to run at a time in the component.

The associated TIBCO configuration field is named *ListenerProcessorThreadNb* and its default value is 1.

4.3.2.2 XML Validation

This field can be used for debugging purposes. It is used to check the validity of all the XML messages received by the SNMP Action Executor component. This field should not be modified except if it is explicitly requested by OpenView SQM support.

The associated TIBCO configuration field is named *XMLValidationMode* and its default value is *false*. It is mapped by default to the global OpenView SQM field *SCXmlMsgValidation.*

4.3.2.3 QuietMode

The *QuietMode* field can be used for debugging purposes. The SNMP Action Executor uses it to avoid message publication. It should not be modified unless requested by OpenView SQM support.

The associated TIBCO configuration field is named *QuietMode* and its default value is *false*.

4.3.2.4 SNMPMaxPacketSize

This field is used to specify the maximum size of the trap packet. This is an advanced configuration field. If you receive the error *Packet size > maximum packet size*, you have to augment this value.

The associated TIBCO configuration field is named *SNMPMaxPacketSize* and its default value is *2000 (octets)*.

4.3.3 Logging and Tracing Properties

The logging and tracing properties are stored in the **trace_sink** and **log_sink** of the `/<InstanceName>` adapter (`.../<ApplicationName>/Advanced/Log Sinks`).

The table below summarizes all the SNMP Action Executor logging and tracing properties.

Table 8 SNMP Action Executor Logging and Tracing Properties

Variable Name	Default Values
Traces	
FileCount	10
FileLimit	10000000
AppendMode	True
Logs	
FileCount	10
FileLimit	1000000
AppendMode	True

For each previously described parameter, a detailed description is given below.

4.3.3.1 FileCount

The *FileCount* field is used to set the maximum number of files (traces or log files).

The associated TIBCO configuration field is named *FileCount* and its default value is 10.

4.3.3.2 FileLimit

The *FileLimit* field is used to set the maximum size of files (traces or log files).

The associated TIBCO configuration field is named *FileLimit* and its default value is 10000000.

4.3.3.3 AppendMode

The *AppendMode* is used to specify that traces or logs can be appended to existing files after a restart.

The associated TIBCO configuration field is named *AppendMode* and its default value is true.

4.4 Reload Configuration

The changes made on the SNMP Action Executor application configuration are taken into account at SNMP Action Executor startup.

However, it is possible to take the changes into account while the SNMP Action Executor is running, invoking the SNMP Action Executor *ReloadConfig* AMI method.

This method can be called through the *HawkDisplay* application.

To launch it, type the following command (with 'sqmadm' user):

```
temip_sc_hawk_display_start
```

Please refer to the *HP OpenView Service Quality Manager Administration Guide* for more information on how to use the *HawkDisplay* application.

4.5 Configuring the Objective Status Changed Filter

When the "advanced filtering" mechanism is enabled, it is necessary to define the filter to use on Objective Status Changed events (see section 2.2.1 - Association when 'ObjectiveStatusFiltering' is enabled).

This filter, composed of the *'sla.status.trigger'* and *'si.status.trigger'* regular expressions, is stored in a XML file named:
'<platform>_<director>_<application>_filter.xml'

Example:

```
slmv12_gateway_SNMPGtw_filter.xml
```

This file is located in the SNMP Action Executor data directory:
'$TEMIP_SC_VAR_HOME/Gateways/SNMP/v1_2/config'

Filter file content

The filter file content has to respect the following DTD:

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT Filter EMPTY>
<!ATTLIST Filter
   sla.status.trigger      CDATA         #IMPLIED
   si.status.trigger       CDATA         #IMPLIED
>
```

Example:

```
<?xml version="1.0" encoding="UTF-8"?>
<Filter sla.status.trigger="Gold.*" si.status.trigger="" />
```

In this example, the filter only matches the Objective Status Changes of the SLA having a name starting with "Gold".

Appendix G gives a summary of Regular Expression constructs.

Important

The SNMP Action will takes into account the changes made in its Filter Configurations file at its startup or after having invocated its ReloadConfig AMI (see section 4.4 - Reload Configuration)

4.6 Configuring the SLA Administration GUI

In some cases, you may use an SLA Administration GUI that is not already configured to support the SNMP Action Executor while defining the objective thresholds. You can find this out by checking the available Action Executor types while defining a threshold. If the type "SNMPTrap" is not available, then the following configuration step must be performed.

In order to set **SNMPTrap** as Action Executor when defining the thresholds, a file provided with the SLA Administration has to be modified.

This file is located under the SQM SLA Administration installation folder (by default C:\Program Files\HP OpenView\SQM).

1. Go to the following directory:

```
# cd %TEMIP_SC_HOME%\UI\SLAClient\properties
```

2. Edit the *AvailableActions.xml* file

3. Add the following lines at the end of the file:

```
</Executor>
    <Executor executor.name = "SNMPTrap" executor.label =
"SNMP Trap">
          <Descr>This executor sends SNMP Traps</Descr>
          <Action action.name = "SendTrap" action.label =
"Send trap" action.info = "" > <Descr>Send SNMP trap</Descr>
</Action>
    </Executor>
</AvailableActions>
```

4.7 SNMP Action Executor Monitoring

It is possible to monitor and administrate the SNMP Action Executor using the *TIBCO HawkDisplay* application.

For more information, refer to the *hp OpenView Service Quality Manager Administration Guide*.

Chapter 5

SNMP Action Executor Start/Stop

Before starting or stopping the SNMP Action Executor, ensure that:

- The OpenView SQM Kernel is running.
- The OpenView SQM environment is set.
  (Section 3.2.1 explains how to set the SQM environment).
- You are logged as *sqmadm.*

5.1 Starting the SNMP Action Executor Process

To start the SNMP Action Executor you have to execute the following command:

```
temip_sc_start_application –platform <SQM platform
Name> -director <SNMP Gateway director Name>
-application <SNMP Gateway instance name>
```

Example:

```
# temip_sc_start_application -platform slmv12 -director
gateway -application SNMPGateway
```

For more information, refer to the *HP OpenView Service Quality Manager Administration Guide*.

5.2 Stopping the SNMP Action Executor Process

To stop the SNMP Action Executor you have to execute the following command:

```
temip_sc_stop_application –platform <SQM platform Name>
-director <SNMP Gateway director Name> -application
<SNMP Gateway instance name>
```

Example:

```
# temip_sc_stop_application -platform slmv12 -director
gateway -application SNMPGateway
```

For more information, refer to the *HP OpenView Service Quality Manager Administration Guide*.

Appendix A

Generated Traps Examples

Below are given some examples of traps generated by the SNMP Action Executor. They have been generated with the Video sample model.

This is an example of a customerParamThresCrossing trap.

```
Trap PDU -
   request-id : 5
   error-status : 0
   error-index : 0
   variable-bindings :
   Variable - (
      OID - [11] { 1 3 6 1 6 3 1 1 4 1 0 }
      OID - [14] { 1 3 6 1 4 1 11 2 17 15 1 1 2 1 }     type=val)
   Variable - (
      OID - [22] { 1 3 6 1 4 1 11 2 17 15 1 4 1 3 7 115 108 97 78 97 109
101 }
      Int - 1
      type=val)
   Variable - (
      OID - [14] { 1 3 6 1 4 1 11 2 17 15 1 1 1 1 }
      Int - 2
      type=val)
   Variable - (
      OID - [21] { 1 3 6 1 4 1 11 2 17 15 1 5 1 1 6 115 105 78 97 109 101 }
      OctStr - [11]      _LABEL Lyon
      type=val)
   Variable - (
      OID - [21] { 1 3 6 1 4 1 11 2 17 15 1 5 1 3 6 115 105 78 97 109 101 }
      OctStr - [5]       Video
      type=val)
   Variable - (
      OID - [21] { 1 3 6 1 4 1 11 2 17 15 1 5 1 4 6 115 105 78 97 109 101 }
      OctStr - [5]       Video
      type=val)
   Variable - (
      OID - [25] { 1 3 6 1 4 1 11 2 17 15 1 2 1 1 10 101 110 116 105 116
121 78 97 109 101 }
      OctStr - [6]       Compaq
      type=val)
   Variable - (
      OID - [25] { 1 3 6 1 4 1 11 2 17 15 1 2 1 2 10 101 110 116 105 116
121 78 97 109 101 }
      OctStr - [13]      _LABEL Compaq
      type=val)
   Variable - (
      OID - [25] { 1 3 6 1 4 1 11 2 17 15 1 2 1 3 10 101 110 116 105 116
121 78 97 109 101 }
      Int - 1
```

```
      type=val)
   Variable - (
      OID - [22] { 1 3 6 1 4 1 11 2 17 15 1 4 1 5 7 115 108 97 78 97 109
101 }
      OctStr - [10]      Gold-Video
      type=val)
   Variable - (
      OID - [22] { 1 3 6 1 4 1 11 2 17 15 1 4 1 6 7 115 108 97 78 97 109
101 }
      OctStr - [24]      Gold Video Service Level
      type=val)
   Variable - (
      OID - [22] { 1 3 6 1 4 1 11 2 17 15 1 4 1 1 7 115 108 97 78 97 109
101 }
      OctStr - [12]      GoldVideoFR2
      type=val)
   Variable - (
      OID - [22] { 1 3 6 1 4 1 11 2 17 15 1 6 1 1 7 115 99 105 78 97 109
101 }
      OctStr - [15]      platform-CamWeb
      type=val)
   Variable - (
      OID - [22] { 1 3 6 1 4 1 11 2 17 15 1 6 1 2 7 115 99 105 78 97 109
101 }
      OctStr - [22]      _LABEL platform-CamWeb
      type=val)
   Variable - (
      OID - [22] { 1 3 6 1 4 1 11 2 17 15 1 6 1 4 7 115 99 105 78 97 109
101 }
      OctStr - [9]       PlatformV
      type=val)
   Variable - (
      OID - [22] { 1 3 6 1 4 1 11 2 17 15 1 6 1 5 7 115 99 105 78 97 109
101 }
      OctStr - [8]       Platform
      type=val)
   Variable - (
      OID - [22] { 1 3 6 1 4 1 11 2 17 15 1 6 1 3 7 115 99 105 78 97 109
101 }
      Int - 2
      type=val)
   Variable - (
      OID - [14] { 1 3 6 1 4 1 11 2 17 15 1 1 1 4 }
      OctStr - [11]
0: 07 d4 03 19  0f 19 3b 00  2b 01 00              ......;.+..
      type=val)
   Variable - (
      OID - [14] { 1 3 6 1 4 1 11 2 17 15 1 1 1 3 }
      OctStr - [46]      ADD INFO VIOL CRITICAL
      type=val)
   Variable - (
      OID - [14] { 1 3 6 1 4 1 11 2 17 15 1 1 1 5 }
      OctStr - [3]
0: 31 30 30                                    100
      type=val)
   Variable - (
      OID - [71] { 1 3 6 1 4 1 11 2 17 15 1 7 1 3 19 115 108 97 83 101 114
118 105 99 101 76 101 118 101 108 78 97 109 101 28 115 108 111 67 111 109
112 111 110 101 110 116 83 101 114 118 105 99 101 76 101 118 101 108 78 97
109 101 7 115 108 111 78 97 109 101 }
      OctStr - [13]      PLATFORMV6843
      type=val)
   Variable - (
```

```
     OID - [71] { 1 3 6 1 4 1 11 2 17 15 1 7 1 4 19 115 108 97 83 101 114
118 105 99 101 76 101 118 101 108 78 97 109 101 28 115 108 111 67 111 109
112 111 110 101 110 116 83 101 114 118 105 99 101 76 101 118 101 108 78 97
109 101 7 115 108 111 78 97 109 101 }
     OctStr - [15]      _LABEL Platform
     type=val)
   Variable - (
     OID - [71] { 1 3 6 1 4 1 11 2 17 15 1 7 1 1 19 115 108 97 83 101 114
118 105 99 101 76 101 118 101 108 78 97 109 101 28 115 108 111 67 111 109
112 111 110 101 110 116 83 101 114 118 105 99 101 76 101 118 101 108 78 97
109 101 7 115 108 111 78 97 109 101 }
     OctStr - [12]      CPULoadBelow
     type=val)
   Variable - (
     OID - [71] { 1 3 6 1 4 1 11 2 17 15 1 7 1 2 19 115 108 97 83 101 114
118 105 99 101 76 101 118 101 108 78 97 109 101 28 115 108 111 67 111 109
112 111 110 101 110 116 83 101 114 118 105 99 101 76 101 118 101 108 78 97
109 101 7 115 108 111 78 97 109 101 }
     OctStr - [25]      CPU Load below 60 percent
     type=val)
   Variable - (
     OID - [71] { 1 3 6 1 4 1 11 2 17 15 1 7 1 5 19 115 108 97 83 101 114
118 105 99 101 76 101 118 101 108 78 97 109 101 28 115 108 111 67 111 109
112 111 110 101 110 116 83 101 114 118 105 99 101 76 101 118 101 108 78 97
109 101 7 115 108 111 78 97 109 101 }
     OctStr - [7]       CPULoad
     type=val)
   Variable - (
     OID - [71] { 1 3 6 1 4 1 11 2 17 15 1 7 1 6 19 115 108 97 83 101 114
118 105 99 101 76 101 118 101 108 78 97 109 101 28 115 108 111 67 111 109
112 111 110 101 110 116 83 101 114 118 105 99 101 76 101 118 101 108 78 97
109 101 7 115 108 111 78 97 109 101 }
     OctStr - [14]      _LABEL CPULoad
     type=val)
   Variable - (
     OID - [71] { 1 3 6 1 4 1 11 2 17 15 1 7 1 7 19 115 108 97 83 101 114
118 105 99 101 76 101 118 101 108 78 97 109 101 28 115 108 111 67 111 109
112 111 110 101 110 116 83 101 114 118 105 99 101 76 101 118 101 108 78 97
109 101 7 115 108 111 78 97 109 101 }
     Int - 1
     type=val)
   Variable - (
     OID - [71] { 1 3 6 1 4 1 11 2 17 15 1 7 1 8 19 115 108 97 83 101 114
118 105 99 101 76 101 118 101 108 78 97 109 101 28 115 108 111 67 111 109
112 111 110 101 110 116 83 101 114 118 105 99 101 76 101 118 101 108 78 97
109 101 7 115 108 111 78 97 109 101 }
     Int - 4
     type=val)
   Variable - (
     OID - [71] { 1 3 6 1 4 1 11 2 17 15 1 7 1 9 19 115 108 97 83 101 114
118 105 99 101 76 101 118 101 108 78 97 109 101 28 115 108 111 67 111 109
112 111 110 101 110 116 83 101 114 118 105 99 101 76 101 118 101 108 78 97
109 101 7 115 108 111 78 97 109 101 }
     OctStr - [9]       violation
     type=val)
   Variable - (
     OID - [71] { 1 3 6 1 4 1 11 2 17 15 1 7 1 10 19 115 108 97 83 101 114
118 105 99 101 76 101 118 101 108 78 97 109 101 28 115 108 111 67 111 109
112 111 110 101 110 116 83 101 114 118 105 99 101 76 101 118 101 108 78 97
109 101 7 115 108 111 78 97 109 101 }
     OctStr - [14]      _LABEL Level 2
     type=val)
   Variable - (
```

```
     OID - [71] { 1 3 6 1 4 1 11 2 17 15 1 7 1 11 19 115 108 97 83 101 114
118 105 99 101 76 101 118 101 108 78 97 109 101 28 115 108 111 67 111 109
112 111 110 101 110 116 83 101 114 118 105 99 101 76 101 118 101 108 78 97
109 101 7 115 108 111 78 97 109 101 }
     OctStr - [2]
0: 36 30                                          60
     type=val)
   Variable - (
     OID - [71] { 1 3 6 1 4 1 11 2 17 15 1 7 1 12 19 115 108 97 83 101 114
118 105 99 101 76 101 118 101 108 78 97 109 101 28 115 108 111 67 111 109
112 111 110 101 110 116 83 101 114 118 105 99 101 76 101 118 101 108 78 97
109 101 7 115 108 111 78 97 109 101 }
     OctStr - [8]       EmitTrap
     type=val)
   Variable - (
     OID - [14] { 1 3 6 1 4 1 11 2 17 15 1 1 1 8 }
     OctStr - [8]       Critical
     type=val)
   Discarded vars : 0
   Validity : 0
   Trap OID : OID - [14] { 1 3 6 1 4 1 11 2 17 15 1 1 2 1 }
   Enterprise : OID - [13] { 1 3 6 1 4 1 11 2 17 15 1 1 2 }
   TimeStamp : 0
   Type : 6
```

This is an example of an operationServiceObjectiveViolation trap.

```
Trap PDU -
   request-id : 16
   error-status : 0
   error-index : 0
   variable-bindings :
   Variable - (
     OID - [11] { 1 3 6 1 6 3 1 1 4 1 0 }
     OID - [14] { 1 3 6 1 4 1 11 2 17 15 1 1 5 1 }     type=val)
   Variable - (
     OID - [22] { 1 3 6 1 4 1 11 2 17 15 1 4 1 3 7 115 108 97 78 97 109
101 }
     Int - 2
     type=val)
   Variable - (
     OID - [22] { 1 3 6 1 4 1 11 2 17 15 1 4 1 1 7 115 108 97 78 97 109
101 }
     OctStr - [13]        SilverVideoFR
     type=val)
   Variable - (
     OID - [22] { 1 3 6 1 4 1 11 2 17 15 1 4 1 2 7 115 108 97 78 97 109
101 }
     OctStr - [20]        _LABEL SilverVideoFR
     type=val)
   Variable - (
     OID - [21] { 1 3 6 1 4 1 11 2 17 15 1 5 1 3 6 115 105 78 97 109 101 }
     OctStr - [5]         Video
     type=val)
   Variable - (
     OID - [21] { 1 3 6 1 4 1 11 2 17 15 1 5 1 4 6 115 105 78 97 109 101 }
     OctStr - [5]         Video
     type=val)
   Variable - (
     OID - [21] { 1 3 6 1 4 1 11 2 17 15 1 5 1 1 6 115 105 78 97 109 101 }
     OctStr - [5]         Paris
```

```
      type=val)
   Variable - (
      OID - [21] { 1 3 6 1 4 1 11 2 17 15 1 5 1 2 6 115 105 78 97 109 101 }
      OctStr - [12]         _LABEL Paris
      type=val)
   Variable - (
      OID - [22] { 1 3 6 1 4 1 11 2 17 15 1 6 1 4 7 115 99 105 78 97 109
101 }
      OctStr - [9]          PlatformV
      type=val)
   Variable - (
      OID - [22] { 1 3 6 1 4 1 11 2 17 15 1 6 1 5 7 115 99 105 78 97 109
101 }
      OctStr - [8]          Platform
      type=val)
   Variable - (
      OID - [22] { 1 3 6 1 4 1 11 2 17 15 1 6 1 1 7 115 99 105 78 97 109
101 }
      OctStr - [15]         platform-CamWeb
      type=val)
   Variable - (
      OID - [22] { 1 3 6 1 4 1 11 2 17 15 1 6 1 2 7 115 99 105 78 97 109
101 }
      OctStr - [22]         _LABEL platform-CamWeb
      type=val)
   Variable - (
      OID - [22] { 1 3 6 1 4 1 11 2 17 15 1 6 1 3 7 115 99 105 78 97 109
101 }
      Int - 2
      type=val)
   Variable - (
      OID - [25] { 1 3 6 1 4 1 11 2 17 15 1 2 1 1 10 101 110 116 105 116
121 78 97 109 101 }
      OctStr - [16]         VideoMaintenance
      type=val)
   Variable - (
      OID - [25] { 1 3 6 1 4 1 11 2 17 15 1 2 1 2 10 101 110 116 105 116
121 78 97 109 101 }
      OctStr - [23]         _LABEL VideoMaintenance
      type=val)
   Variable - (
      OID - [25] { 1 3 6 1 4 1 11 2 17 15 1 2 1 3 10 101 110 116 105 116
121 78 97 109 101 }
      Int - 2
      type=val)
   Variable - (
      OID - [14] { 1 3 6 1 4 1 11 2 17 15 1 1 1 4 }
      OctStr - [11]           0: 07 d4 03 19  0f 19 3b 00  2b 01 00
......;.+..
      type=val)
   Variable - (
      OID - [14] { 1 3 6 1 4 1 11 2 17 15 1 1 1 6 }
      Timeticks - 100
      type=val)
   Variable - (
      OID - [14] { 1 3 6 1 4 1 11 2 17 15 1 1 1 7 }
      Timeticks - 0
      type=val)
   Discarded vars : 0
   Validity : 0
   Trap OID : OID - [14] { 1 3 6 1 4 1 11 2 17 15 1 1 5 1 }
   Enterprise : OID - [13] { 1 3 6 1 4 1 11 2 17 15 1 1 5 }
   TimeStamp : 0
```

```
   Type : 6
```

This is an example of a customerServiceObjectiveViolation trap.

```
Trap PDU -
   request-id : 14
   error-status : 0
   error-index : 0
   variable-bindings :
   Variable - (
      OID - [11] { 1 3 6 1 6 3 1 1 4 1 0 }
      OID - [14] { 1 3 6 1 4 1 11 2 17 15 1 1 4 1 }     type=val)
   Variable - (
      OID - [22] { 1 3 6 1 4 1 11 2 17 15 1 4 1 3 7 115 108 97 78 97 109
101 }
      Int - 1
      type=val)
   Variable - (
      OID - [22] { 1 3 6 1 4 1 11 2 17 15 1 4 1 1 7 115 108 97 78 97 109
101 }
      OctStr - [15]      LABELGOLDVI9858
      type=val)
   Variable - (
      OID - [22] { 1 3 6 1 4 1 11 2 17 15 1 4 1 2 7 115 108 97 78 97 109
101 }
      OctStr - [12]      GoldVideoFR2
      type=val)
   Variable - (
      OID - [21] { 1 3 6 1 4 1 11 2 17 15 1 5 1 3 6 115 105 78 97 109 101 }
      OctStr - [5]       Video
      type=val)
   Variable - (
      OID - [21] { 1 3 6 1 4 1 11 2 17 15 1 5 1 4 6 115 105 78 97 109 101 }
      OctStr - [5]       Video
      type=val)
   Variable - (
      OID - [21] { 1 3 6 1 4 1 11 2 17 15 1 5 1 1 6 115 105 78 97 109 101 }
      OctStr - [4]       Lyon
      type=val)
   Variable - (
      OID - [21] { 1 3 6 1 4 1 11 2 17 15 1 5 1 2 6 115 105 78 97 109 101 }
      OctStr - [11]      _LABEL Lyon
      type=val)
   Variable - (
      OID - [22] { 1 3 6 1 4 1 11 2 17 15 1 6 1 4 7 115 99 105 78 97 109
101 }
      OctStr - [9]       PlatformV
      type=val)
   Variable - (
      OID - [22] { 1 3 6 1 4 1 11 2 17 15 1 6 1 5 7 115 99 105 78 97 109
101 }
      OctStr - [8]       Platform
      type=val)
   Variable - (
      OID - [22] { 1 3 6 1 4 1 11 2 17 15 1 6 1 1 7 115 99 105 78 97 109
101 }
      OctStr - [15]      platform-CamWeb
      type=val)
   Variable - (
      OID - [22] { 1 3 6 1 4 1 11 2 17 15 1 6 1 2 7 115 99 105 78 97 109
101 }
```

```
      OctStr - [22]      _LABEL platform-CamWeb
      type=val)
   Variable - (
      OID - [22] { 1 3 6 1 4 1 11 2 17 15 1 6 1 3 7 115 99 105 78 97 109
101 }
      Int - 2
      type=val)
   Variable - (
      OID - [25] { 1 3 6 1 4 1 11 2 17 15 1 2 1 1 10 101 110 116 105 116
121 78 97 109 101 }
      OctStr - [6]       Compaq
      type=val)
   Variable - (
      OID - [25] { 1 3 6 1 4 1 11 2 17 15 1 2 1 2 10 101 110 116 105 116
121 78 97 109 101 }
      OctStr - [13]      _LABEL Compaq
      type=val)
   Variable - (
      OID - [25] { 1 3 6 1 4 1 11 2 17 15 1 2 1 3 10 101 110 116 105 116
121 78 97 109 101 }
      Int - 1
      type=val)
   Variable - (
      OID - [14] { 1 3 6 1 4 1 11 2 17 15 1 1 1 4 }
      OctStr - [11]0: 07 d4 03 19  0f 19 3b 00  2b 01 00     ......;.+..
      type=val)
   Variable - (
      OID - [14] { 1 3 6 1 4 1 11 2 17 15 1 1 1 6 }
      Timeticks - 100
      type=val)
   Variable - (
      OID - [14] { 1 3 6 1 4 1 11 2 17 15 1 1 1 7 }
      Timeticks - 0
      type=val)
   Discarded vars : 0
   Validity : 0
   Trap OID : OID - [14] { 1 3 6 1 4 1 11 2 17 15 1 1 4 1 }
   Enterprise : OID - [13] { 1 3 6 1 4 1 11 2 17 15 1 1 4 }
   TimeStamp : 0
   Type : 6
```

Appendix B

Installation Directory Structure

The following directories and files are installed:

SNMP Action Executor Subset

```
$TEMIP_SC_HOME/Gateways
$TEMIP_SC_HOME/Gateways/SNMP
$TEMIP_SC_HOME/Gateways/SNMP/v1_2
$TEMIP_SC_HOME/Gateways/SNMP/v1_2/config
$TEMIP_SC_HOME/Gateways/SNMP/v1_2/config/SCPlatform_SCDirector_SCApplicati
on.properties
$TEMIP_SC_HOME/Gateways/SNMP/v1_2/config/SCPlatform_SCDirector_SCApplicati
on_filter.xml
$TEMIP_SC_HOME/Gateways/SNMP/v1_2/jar
$TEMIP_SC_HOME/Gateways/SNMP/v1_2/jar/TeSCSNMPGateway.jar
$TEMIP_SC_HOME/Gateways/SNMP/v1_2/jar/snmp4_13.jar
$TEMIP_SC_HOME/Gateways/SNMP/v1_2/properties
$TEMIP_SC_HOME/Gateways/SNMP/v1_2/properties/TeSCSNMPGateway.properties
$TEMIP_SC_HOME/Gateways/SNMP/v1_2/properties/TeSCSNMPGateway_Messages.prop
erties
$TEMIP_SC_HOME/Gateways/SNMP/v1_2/properties/TeSCSNMPGateway_Version.prope
rties
$TEMIP_SC_HOME/Gateways/SNMP/v1_2/repository
$TEMIP_SC_HOME/Gateways/SNMP/v1_2/repository/SNMPGateway_setup.cfg
$TEMIP_SC_HOME/Gateways/SNMP/v1_2/repository/SNMPGateway_template.exp
$TEMIP_SC_HOME/Gateways/SNMP/v1_2/Uninstaller_SNMPAE
$TEMIP_SC_HOME/Gateways/SNMP/v1_2/Uninstaller_SNMPAE/Uninstall_SNMPAE
$TEMIP_SC_HOME/Gateways/SNMP/v1_2/Uninstaller_SNMPAE/Uninstall_SNMPAE.lax
$TEMIP_SC_HOME/Gateways/SNMP/v1_2/Uninstaller_SNMPAE/uninstaller.jar
$TEMIP_SC_HOME/adapter/bin/snmpgtw_v1_2_launch.sh
$TEMIP_SC_HOME/etc/addOn/SNMPGateway_v1_2_addOn.tmpl_cfg
$TEMIP_SC_HOME/fileset/SQMSNMPAE-1.20.00
```

SA/Gateway Common Subset

```
$TEMIP_SC_HOME/DTD/discovery.dtd
$TEMIP_SC_HOME/DTD/inventory.dtd
$TEMIP_SC_HOME/Gateways
$TEMIP_SC_HOME/Gateways/Common
$TEMIP_SC_HOME/Gateways/Common/v1_2
$TEMIP_SC_HOME/Gateways/Common/v1_2/jar
$TEMIP_SC_HOME/Gateways/Common/v1_2/jar/TeSCGtwCommon.jar
$TEMIP_SC_HOME/Gateways/Common/v1_2/properties
$TEMIP_SC_HOME/Gateways/Common/v1_2/properties/TeSCGtwCommon_Messages.prope
rties
$TEMIP_SC_HOME/Gateways/Common/v1_2/properties/TeSCGtwCommon_Version.proper
ties
$TEMIP_SC_HOME/Gateways/Common/v1_2/repository
$TEMIP_SC_HOME/Gateways/Common/v1_2/repository/GtwCommon_setup.cfg
```

```
$TEMIP_SC_HOME/Gateways/Common/v1_2/repository/GtwCommon_template.exp
$TEMIP_SC_HOME/ServiceAdapters
$TEMIP_SC_HOME/ServiceAdapters/Common
$TEMIP_SC_HOME/ServiceAdapters/Common/v1_2
$TEMIP_SC_HOME/ServiceAdapters/Common/v1_2/jar
$TEMIP_SC_HOME/ServiceAdapters/Common/v1_2/jar/TeSCSACommon.jar
$TEMIP_SC_HOME/ServiceAdapters/Common/v1_2/properties
$TEMIP_SC_HOME/ServiceAdapters/Common/v1_2/properties/TeSCSACommon.properti
es
$TEMIP_SC_HOME/ServiceAdapters/Common/v1_2/properties/TeSCSACommon_Messages
.properties
$TEMIP_SC_HOME/ServiceAdapters/Common/v1_2/properties/TeSCSACommon_Version.
properties
$TEMIP_SC_HOME/ServiceAdapters/Common/v1_2/properties/TeSCSAConfig_Messages
.properties
$TEMIP_SC_HOME/ServiceAdapters/Sql
$TEMIP_SC_HOME/ServiceAdapters/Sql/v1_0
$TEMIP_SC_HOME/ServiceAdapters/Sql/v1_0/config
$TEMIP_SC_HOME/ServiceAdapters/Sql/v1_0/config/SCPlatform_SCDirector_SCAppl
ication.properties
$TEMIP_SC_HOME/ServiceAdapters/Sql/v1_0/jar
$TEMIP_SC_HOME/ServiceAdapters/Sql/v1_0/jar/TeSCSASql.jar
$TEMIP_SC_HOME/ServiceAdapters/Sql/v1_0/properties
$TEMIP_SC_HOME/ServiceAdapters/Sql/v1_0/properties/TeSCSql.properties
$TEMIP_SC_HOME/ServiceAdapters/Sql/v1_0/properties/TeSCSql_Messages.propert
ies
$TEMIP_SC_HOME/ServiceAdapters/Sql/v1_0/properties/TeSCSql_Version.properti
es
$TEMIP_SC_HOME/fileset
$TEMIP_SC_HOME/fileset/SQMSAGTWCOMMON-1.20.XX-XXXXXXXXXXX
```

Appendix C

Troubleshooting

Troubleshooting in the OpenView SQM SNMP Action Executor component is achieved using tracing capabilities: debug tracing and error logging. Debug and error messages are both saved in files. Different levels of trace allow specific troubleshooting.

C 1 Dump

The SNMP Action Executor application implements a dump AMI method allowing the display of its internal data:

- **Config**: the SNMP Action Executor configuration
- **Memory**: the internal memory objects
- **Topics**: the listened topics
- **All**: all of the above data

This method can be called through the *HawkDisplay* application.

To launch it, type the following command (with 'sqmadm' user):

```
temip_sc_hawk_display_start
```

In case of a problem, the SQM Support may ask you to invoke this dump AMI command.

Please refer to the *HP OpenView Service Quality Manager Administration Guide* for more information on how to use the *HawkDisplay* application.

C 2 Trace Files

C 2.1 Log Files

In order to help the troubleshooting of the OpenView SQM SNMP Action Executor component, a new log line is appended to the file in the `$TEMIP_SC_VAR_HOME/log` directory when an unexpected error occurs. The log file name is *SCPlatform_SCDirector_SCApplication.log*. You can check the content of the log file to discover and understand the problem.

To activate the logging feature, modify the '*SCPlatform_SCDirector_SCApplication*.properties' file located in the '`$TEMIP_SC_VAR_HOME/Gateways/SNMP/config`' directory. Set the LOG_LEVEL to ALL to have full logs.

```
LOG_LEVEL=ALL
```

The possible values are:

- ALL: full logs.

- MINOR: only CRITICAL, MAJOR and MINOR logs are dumped.
- MAJOR: only the CRITICAL and MAJOR logs are dumped.
- CRITICAL: only the CRITICAL logs are dumped.
- OFF: disables the logging mechanism.

Once you have modified this field in this file, you have to stop and restart the SNMP Action Executor. The Reload Configuration AMI method does not take these modifications into account. You have to use the *temip_sc_stop_application* and *temip_sc_start_application* scripts. Please refer to the *OpenView Service Quality Manager Administration Guide* for more information on how to use these scripts.

Changing the Log level without stopping the SNMP Action Executor Component

If you want to change the Log level without stopping the SNMP Action Executor component, you can use the setMtLogLevel AMI method. This method can be called through the *HawkDisplay* application.

Refer to the *OpenView Service Quality Manager Administration Guide* for more information on the *HawkDisplay* application.

C 2.2 Trace Files

The OpenView SQM SNMP Action Executor component can generate trace files. These traces are located in the `$TEMIP_SC_VAR_HOME/trace` folder.

To activate the trace logging, modify the '*SCPlatform_SCDirector_SCApplication*.properties' file located in the '`$TEMIP_SC_VAR_HOME/Gateways/SNMP/config`' directory. Set the TRACE_LEVEL to **ALL** to have full traces.

```
TRACE_LEVEL=ALL
```

The possible values are:

- ALL: full traces.
- FINEST: only FINE, FINER and FINEST traces are dumped.
- FINER: only the FINE and FINER traces are dumped.
- FINE: only the FINE traces are dumped.
- OFF: disable the tracing mechanism.

Once you have modified this field, you have to stop and restart the SNMP Action Executor component. The Reload Configuration AMI method does not take these modifications into account. You have to use the *temip_sc_stop_application* and *temip_sc_start_application* scripts. Please refer to the *OpenView Service Quality Manager Administration Guide* for more information on how to use these scripts.

Changing the Trace level without stopping the SNMP Action Executor component

If you want to change the Trace level without stopping the SNMP Action Executor component, you can use the setTraceLogLevel AMI method. This method can be called through the *HawkDisplay* application.

Refer to the *OpenView Service Quality Manager Administration Guide* for more information on the *HawkDisplay* application.

C 2.3 SNMP Action Executor Errors

C 2.3.1 Packet Size Error

If you encounter in the log file the following error message:

```
Packet size (value) is > maximum size (2000)
```

You have to change the field called *SNMPMaxPacketSize* in the Tibco Repository and increase the default value (2000).

C 2.3.2 Traps are not Sent

If the traps are not correctly sent, verify that the SNMP Manager set in the SQM SNMP Action Executor configuration is reachable:

```
On SQM Host
# ping <SNMP Manager>
```

Appendix D

Advanced Installation

The recommended architecture is to install the SNMP Action Executor on the SQM SLM Primary host.

However, it is possible to install the SNMP Action Executor on another host.

This appendix describes how to install and configure the SNMP Action Executor on a host different from the SQM SLM Primary host.

D 1 SNMP Action Executor Installation

Before installing the SNMP Action Executor, you have to ensure that the *OV SQM V1.2 Kernel* is installed on the host (see Section D 1.1 Installing the OV SQM Kernel).

Then you have to set the SQM environment (see below, section D 1.2).

Finally, you can install the SNMP Action Executor subset (see Section 3.2.2 Installing the SQM SNMP Action Executor).

For a general overview of the OpenView SQM installation, please refer to the *HP OpenView Service Quality Manager Installation Guide*.

D 1.1 Installing the OV SQM Kernel

To install the Kernel, perform the following steps:

1. Mount the Service Adapters and Gateways CD-ROM on your system
2. Go to the `SQM-1.20.xx-SAGTW` directory
3. Run the **sqm_install** tool under the **root** account.
4. Select the **minimal** feature. Example:

```
#sqm_install /opt/OV/SQM
/mnt/cdrom/SQM-1.20.00-SAGTW/HPUX/KIT minimal
```

5. Press **Enter** to install the Kernel.

D 1.2 Required Environments

sqmadm User and Group

A **sqmadm** user and group are necessary for the kernel/application setup and management.

Please refer to the *HP OpenView Service Quality Manager Installation Guide* for instructions on how to create the **sqmadm** user and group.

Environment Variables

If the kernel setup has already been done, you have to source the *temip_sc_env.sh* file located under SQM data directory (*$TEMIP_SC_VAR_HOME)*.

```
# . /var/opt/OV/SQM/slmv12/temip_sc_env.sh
```

If the kernel setup is not yet done, before installing the SNMP Action Executor, you have to set your environment.

This consists in:

- Setting the TEMIP_SC_HOME and TEMIP_SC_VAR_HOME environment variables.

Example:

```
# export TEMIP_SC_HOME=/opt/OV/SQM
# export TEMIP_SC_VAR_HOME=/var/opt/OV/SQM/slmv12
```

- Sourcing files:

```
# .  $TEMIP_SC_HOME/jre/jre-setup.sh
# .  $TEMIP_SC_HOME/perl/perl-setup.sh
```

D 1.3 Installing SNMP Action Executor

After the SQM Kernel installation, you have to be connected as *root* user, with the previously described environment variables, to perform the SNMP Action Executor installation described in Section 3.2.2 Installing the SQM SNMP Action Executor.

D 2 SNMP Action Executor Setup and Configuration

To setup the SNMP Action Executor on a host different from the SQM SLM Primary Server, it is necessary to set the SQM environment variables and to have a valid SQM Platform Description file.

The procedure is then identical to the SNMP Action Executor setup on the SQM SLM Primary Server. Chapter 4 explains how to setup and configure the SNMP Action Executor.

Note

If needed, the SNMP Action Executor setup automatically performs the SQM kernel setup.

Environment Variable

The section D 1.2 above explains how to set the SQM environment variable.

Platform Description file

You have to check that a Platform Description file is present at the following location:

```
$TEMIP_SC_HOME/tmp/platform_desc.cfg
```

If the Platform Description file is not present, you have to retrieve it from the SQM SLM Primary Server.

```
Copy from SQM SLM Primary Server:
 $TEMIP_SC_VAR_HOME/setupconfig/platform_desc.cfg
To SNMP Action Executor host:
 $TEMIP_SC_HOME/tmp/platform_desc.cfg
```

Appendix E

Executor Name Customization

By default, the SNMP Action Executor triggers traps emissions on SQM events that have the **SNMPTrap** associated action.

However, it is possible to customize an SNMP Action Executor to trigger on SQM events having another associated action.

This advanced configuration feature may be useful to dedicate a SNMP Action Executor instance to a specific SNMP Manager.

Changing the SNMP Action Executor trigger name

In fact, the SNMP Action Executor triggers traps when an incoming SQM event has an associated action equal to its 'ActionExecutorName' property.

This property is stored in the SNMP Action Executor Application configuration, in the SQM central repository (see the section 4.3.1.4 – 'ActionExecutorName').

This property default value is 'SNMPTrap' but it can be changed to another value (limited to 16 characters).

Associating new executor name to a Service Level Objective

Once the SNMP Action Executor configured to triggers traps on another executor name, it is necessary to declare this executor name to the corresponding Service Level Objective (SLO) definitions.

5.3 This operation is described in the section 2.1 – 'Associating SNMP Action Executor to Parameter Threshold Crossing Events

The SNMP Action Executor only processes the SQM Parameter Threshold Crossing event that has the **SNMPTrap** associated action. The associated action is defined at the Service Level Objective (SLO) definition by the Action Executor field.

'. However, instead of setting the executor to 'SNMPTrap' it has to be set to the new executor name.

Configuring the SLA Administration GUI

To have the new executor name available in the SQM SLA Administration GUI, it is necessary to declare it.

This operation is described in the section 4.6 – 'Configuring the SLA Administration GUI'. However, the added executor bloc has to be something like:

```
<Executor executor.name = "newExecutorName" executor.label =
"new executor label">
```

```
   <Descr>This executor sends SNMP Traps to XXX</Descr>
   <Action action.name = "SendTrap" action.label = "Send trap"
action.info = "" > <Descr>Send SNMP trap</Descr> </Action>
</Executor>
```

Appendix F

SQM SNMP Action Executor MIB

The MIB (Management Information Base) is specified in a file written in a subset of the ASN.1 language.

```
P-OV-SQM DEFINITIONS ::= BEGIN

IMPORTS
        DisplayString, DateAndTime, TruthValue, RowPointer
                FROM SNMPv2-TC
   OBJECT-TYPE, MODULE-IDENTITY, NOTIFICATION-TYPE, enterprises
                FROM SNMPv2-SMI
        MODULE-COMPLIANCE, OBJECT-GROUP
                FROM SNMPv2-CONF;

hp    OBJECT IDENTIFIER ::= { enterprises 11 }
nm    OBJECT IDENTIFIER ::= { hp 2 }
openView   OBJECT IDENTIFIER ::= { nm 17 }

-- To Be Registered OIDs
telco      OBJECT IDENTIFIER ::= { openView 15 }

-- ------------------------------------------------------------
--    OV  S e r v i c e  Q u a l i t y  M a n a g e r     --
-- ------------------------------------------------------------

sqmMIB MODULE-IDENTITY
        LAST-UPDATED  "9405120900Z"
        ORGANIZATION  "HP OpenView Business Unit"
        CONTACT-INFO
                "
                        E-mail: sqm_support@hp.com

                        Christophe Laye (editor)
                "
        DESCRIPTION
                "This is the MIB module for mapping Service (SLA/SLO)
Monitoring."
::= { telco 1 }


sqmTraps   OBJECT IDENTIFIER ::= { sqmMIB 1 }

EntityType ::= TEXTUAL-CONVENTION
    STATUS  current
    DESCRIPTION
       "Representing the type of receiving entity."
    SYNTAX  INTEGER  {
                customer       (1),
                operation      (2)
```

```
    }

SQMId ::= TEXTUAL-CONVENTION
    STATUS       current
    DESCRIPTION
       "Identify a SQM Object"
   -- SYNTAX  OCTET STRING (SIZE (0..16))
      SYNTAX DisplayString

SlaType ::= TEXTUAL-CONVENTION
    STATUS  current
    DESCRIPTION
       "Representing the type of SLA."
    SYNTAX  INTEGER  {
                customer       (1),
                operation      (2)
    }

MonitoringType ::= TEXTUAL-CONVENTION
    STATUS  current
    DESCRIPTION
       "Representing the monitor type perform for a SLA."
    SYNTAX  INTEGER  {
                instance         (1),
                aggregation      (2)
    }

InstanceType ::= TEXTUAL-CONVENTION
    STATUS  current
    DESCRIPTION
       "Indicating the type of instance."
    SYNTAX  INTEGER  {
                service         (1),
                component       (2)
    }

ParameterType ::= TEXTUAL-CONVENTION
    STATUS  current
    DESCRIPTION
       "Indicating the type of instance."
    SYNTAX  INTEGER  {
                global          (1),
                customer        (2)
    }

OperatorType ::= TEXTUAL-CONVENTION
    STATUS  current
    DESCRIPTION
       "Indicating the type of instance."
    SYNTAX  INTEGER  {
                valued          (1),
                equal           (2),
                notEqual        (3),
                up              (4),
                down            (5),
                greaterOrEqual  (6),       -- Not Supported in V1.0
                lowerOrEqual    (7),        -- Not Supported in V1.0
                increase        (8),       -- Not Supported in V1.0
                decrease        (9),       -- Not Supported in V1.0
                inRange         (10),      -- Not Supported in V1.0
                outRange        (11),      -- Not Supported in V1.0
                startWith       (12),      -- Not Supported in V1.0
                contains        (13),      -- Not Supported in V1.0
```

```
                patternMatch    (14)        -- Not Supported in V1.0
    }


--
--  Customer/Operation Objects
--
entityTable  OBJECT-TYPE
    SYNTAX     SEQUENCE OF EntityEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
       "This table defines a entity"
    ::= { sqmMIB 2 }

entityEntry  OBJECT-TYPE
    SYNTAX     EntityEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
       "Each conceptual row defines a receiving entity."
    INDEX   { entityName }
    ::= { entityTable 1 }

EntityEntry ::= SEQUENCE {
    entityName  SQMId,
    entityLabel DisplayString,
    entityType  EntityType
    }

entityName  OBJECT-TYPE
    SYNTAX     SQMId
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
       "Entity Identifier"
    ::= { entityEntry 1 }

entityLabel  OBJECT-TYPE
    SYNTAX     DisplayString
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
       "Entity Name"
    ::= { entityEntry 2 }

entityType  OBJECT-TYPE
    SYNTAX     EntityType
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
       "Entity Type"
    ::= { entityEntry 3 }

--
-- Service Defintion Objects
--

serviceDefinitionTable  OBJECT-TYPE
    SYNTAX     SEQUENCE OF ServiceDefinitionEntry
    MAX-ACCESS not-accessible
    STATUS     current
```

```
    DESCRIPTION
       "This table defines a serviceDefinition"
    ::= { sqmMIB 3 }

serviceDefinitionEntry  OBJECT-TYPE
    SYNTAX     ServiceDefinitionEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
       "Each conceptual row defines a receiving serviceDefinition."
    INDEX   { serviceDefName }
    ::= { serviceDefinitionTable 1 }

ServiceDefinitionEntry ::= SEQUENCE {
    serviceDefName         SQMId,
    serviceDefLabel        DisplayString,
    serviceDefDescription  DisplayString
    }

serviceDefName  OBJECT-TYPE
    SYNTAX     SQMId
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
       "ServiceDefinition Identifier"
    ::= { serviceDefinitionEntry 1 }

serviceDefLabel  OBJECT-TYPE
    SYNTAX     DisplayString
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
       ""
    ::= { serviceDefinitionEntry 2 }


serviceDefDescription  OBJECT-TYPE
    SYNTAX     DisplayString
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
       ""
    ::= { serviceDefinitionEntry 3 }


-- ------------------
-- SLA Objects
-- ------------------

slaTable  OBJECT-TYPE
    SYNTAX     SEQUENCE OF SlaEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
       "This table defines a sla"
    ::= { sqmMIB 4 }

slaEntry  OBJECT-TYPE
    SYNTAX     SlaEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
       "Each conceptual row defines a receiving sla."
```

```
    INDEX   { slaName }
    ::= { slaTable 1 }

SlaEntry ::= SEQUENCE {
    slaName               SQMId,
    slaLabel              DisplayString,
    slaType               SlaType,
    slaMonitoringType     MonitoringType,
    slaServiceLevelName   SQMId,
    slaServiceLevelLabel  DisplayString,
    slaServiceDefinitionName     SQMId,
    slaServiceDefinitionLabel    DisplayString,
    slaEntityName         SQMId
}

slaName  OBJECT-TYPE
    SYNTAX     SQMId
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
       "Sla Identifier"
    ::= { slaEntry 1 }

slaLabel  OBJECT-TYPE
    SYNTAX     DisplayString
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
       "Sla Name"
    ::= { slaEntry 2 }

slaMonitoringType     OBJECT-TYPE
     SYNTAX      SlaType
     MAX-ACCESS read-only
     STATUS     current
     DESCRIPTION
         "Sla Monitoring Type"
     ::= { slaEntry 3 }

slaType     OBJECT-TYPE
     SYNTAX      SlaType
     MAX-ACCESS read-only
     STATUS     current
     DESCRIPTION
         "Sla Type"
     DEFVAL { instance }
     ::= { slaEntry 4 }

slaServiceLevelName     OBJECT-TYPE
     SYNTAX    SQMId
     MAX-ACCESS read-only
     STATUS     current
     DESCRIPTION
         "Sla Service Level Name"
     ::= { slaEntry 5 }

slaServiceLevelLabel     OBJECT-TYPE
     SYNTAX   DisplayString
     MAX-ACCESS read-only
     STATUS     current
     DESCRIPTION
         "Sla Service Level Label"
     ::= { slaEntry 6 }
```

```
slaServiceDefinitionName     OBJECT-TYPE
     SYNTAX      SQMId
     MAX-ACCESS read-only
     STATUS     current
     DESCRIPTION
         "Sla Service Definition Name"
     ::= { slaEntry 7 }

slaServiceDefinitionLabel    OBJECT-TYPE
     SYNTAX      DisplayString
     MAX-ACCESS read-only
     STATUS     current
     DESCRIPTION
         "Sla Service Definition Label"
     ::= { slaEntry 8 }

slaEntityName     OBJECT-TYPE
     SYNTAX          SQMId
     MAX-ACCESS read-only
     STATUS     current
     DESCRIPTION
         "Sla Entity Name"
     ::= { slaEntry 9 }

-- --------------------------------
-- Service
-- --------------------------------
serviceTable  OBJECT-TYPE
    SYNTAX     SEQUENCE OF ServiceEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
       "This table defines a Service Instance"
    ::= { sqmMIB 5 }

serviceEntry  OBJECT-TYPE
    SYNTAX     ServiceEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
       "Each conceptual row defines a SI."
    INDEX   { siName }
    ::= { serviceTable 1 }

ServiceEntry ::= SEQUENCE {
    siName         SQMId,
    siLabel        DisplayString,
    siDefName      SQMId,
    siDefLabel     DisplayString
}

siName  OBJECT-TYPE
    SYNTAX     SQMId
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
       "SCI Identifier"
    ::= { serviceEntry 1 }

siLabel  OBJECT-TYPE
    SYNTAX     DisplayString
    MAX-ACCESS read-only
```

```
    STATUS     current
    DESCRIPTION
       "SCI/SI Name"
    ::= { serviceEntry 2 }

siDefName  OBJECT-TYPE
    SYNTAX     SQMId
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
       "Associateed Service Definition"
    ::= { serviceEntry 3 }

siDefLabel  OBJECT-TYPE
    SYNTAX     DisplayString
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
       "Associateed Service Definition"
    ::= { serviceEntry 4 }


-- --------------------------------
-- Service [Component]
-- --------------------------------

serviceComponentTable  OBJECT-TYPE
    SYNTAX     SEQUENCE OF ServiceComponentEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
       "This table defines a Service/ServiceComponent Instance"
    ::= { sqmMIB 6 }

serviceComponentEntry  OBJECT-TYPE
    SYNTAX     ServiceComponentEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
       "Each conceptual row defines either SI or SCI."
    INDEX   { sciName }
    ::= { serviceComponentTable 1 }

ServiceComponentEntry ::= SEQUENCE {
    sciName        SQMId,
    sciLabel       DisplayString,
    sciType        InstanceType,
    sciDefName     SQMId,
    sciDefLabel    DisplayString
}

sciName  OBJECT-TYPE
    SYNTAX     SQMId
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
       "SCI Identifier"
    ::= { serviceComponentEntry 1 }

sciLabel  OBJECT-TYPE
    SYNTAX     DisplayString
    MAX-ACCESS read-only
    STATUS     current
```

```
    DESCRIPTION
       "SCI/SI Name"
    ::= { serviceComponentEntry 2 }

sciType  OBJECT-TYPE
    SYNTAX     InstanceType
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
       "Instance Type"
    DEFVAL { component }
    ::= { serviceComponentEntry 3 }

sciDefName  OBJECT-TYPE
    SYNTAX     SQMId
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
       "Associateed Component Definition"
    ::= { serviceComponentEntry 4 }

sciDefLabel  OBJECT-TYPE
    SYNTAX     DisplayString
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
       "Associateed Component Definition"
    ::= { serviceComponentEntry 5 }



-- ----------------------------------
-- SLO Objects
-- ----------------------------------
-- It is a flat view of the model
-- Threshold should be in a separate table
--

sloTable  OBJECT-TYPE
    SYNTAX     SEQUENCE OF SloEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
       "This table defines a Service/ServiceComponent Instance"
    ::= { sqmMIB 7 }

sloEntry  OBJECT-TYPE
    SYNTAX     SloEntry
    MAX-ACCESS not-accessible
    STATUS     current
    DESCRIPTION
       "Each conceptual row defines a SLO-Threshold of a given ServiceLevel"
    INDEX   {
               slaServiceLevelName,
               sloComponentServiceLevelName,
               sloName
             }
    ::= { sloTable 1 }

SloEntry ::= SEQUENCE {
        sloName              SQMId,
        sloLabel             DisplayString,
        sloComponentServiceLevelName SQMId,
```

```
        sloComponentServiceLevelLabel DisplayString,
        sloMntrdParamName    SQMId,
        sloMntrdParamLabel   DisplayString,
        sloMntrdParamType    ParameterType,
        sloCrossingType      OperatorType,
        sloThreshName        SQMId,
        sloThreshLabel       DisplayString,
        sloThreshReferenceValue DisplayString,
        sloThreshActionName  DisplayString
}

sloName     OBJECT-TYPE
     SYNTAX     SQMId
     MAX-ACCESS read-only
     STATUS     current
     DESCRIPTION
         ""
     ::= { sloEntry 1 }

sloLabel     OBJECT-TYPE
     SYNTAX     DisplayString
     MAX-ACCESS read-only
     STATUS     current
     DESCRIPTION
         ""
     ::= { sloEntry 2 }

sloComponentServiceLevelName  OBJECT-TYPE
     SYNTAX     SQMId
     MAX-ACCESS read-only
     STATUS     current
     DESCRIPTION
         ""
     ::= { sloEntry 3 }

sloComponentServiceLevelLabel OBJECT-TYPE
     SYNTAX     DisplayString
     MAX-ACCESS read-only
     STATUS     current
     DESCRIPTION
         ""
     ::= { sloEntry 4 }

sloMntrdParamName     OBJECT-TYPE
     SYNTAX     SQMId
     MAX-ACCESS read-only
     STATUS     current
     DESCRIPTION
         ""
     ::= { sloEntry 5 }

sloMntrdParamLabel     OBJECT-TYPE
     SYNTAX     SQMId
     MAX-ACCESS read-only
     STATUS     current
     DESCRIPTION
         ""
     ::= { sloEntry 6 }

sloMntrdParamType     OBJECT-TYPE
     SYNTAX     ParameterType
     MAX-ACCESS read-only
     STATUS     current
```

```
     DESCRIPTION
         ""
     DEFVAL { global }
     ::= { sloEntry 7 }

sloCrossingType     OBJECT-TYPE
     SYNTAX     OperatorType
     MAX-ACCESS read-only
     STATUS     current
     DESCRIPTION
         ""
     ::= { sloEntry 8 }

sloThreshName     OBJECT-TYPE
     SYNTAX     SQMId
     MAX-ACCESS read-only
     STATUS     current
     DESCRIPTION
         ""
     ::= { sloEntry 9 }

sloThreshLabel     OBJECT-TYPE
     SYNTAX     DisplayString
     MAX-ACCESS read-only
     STATUS     current
     DESCRIPTION
         ""
     ::= { sloEntry 10 }

sloThreshReferenceValue     OBJECT-TYPE
     SYNTAX     DisplayString
     MAX-ACCESS read-only
     STATUS     current
     DESCRIPTION
         ""
     ::= { sloEntry 11 }

sloThreshActionName     OBJECT-TYPE
     SYNTAX     DisplayString
     MAX-ACCESS read-only
     STATUS     current
     DESCRIPTION
         ""
     ::= { sloEntry 12 }



-- ------------------------------------
--   N o t i f i c a t i o n s      --
-- ------------------------------------

sqmScalars OBJECT IDENTIFIER ::= { sqmTraps 1 }

rootCause OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
       "Indicate if the report problem is the root cause or not"
    ::= { sqmScalars 1 }

impactedSLANumber OBJECT-TYPE
    SYNTAX     Integer32
```

```
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
       "Indicate the nb of impacted SLAs by one problem"
    ::= { sqmScalars 2 }

addText OBJECT-TYPE
    SYNTAX     DisplayString
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
       "Provide add info about the problem"
    ::= { sqmScalars 3 }

timeStamp OBJECT-TYPE
    SYNTAX     DateAndTime
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
       "Acquisition timeStamp"
    ::= { sqmScalars 4 }

measuredParamValue OBJECT-TYPE
    SYNTAX     DisplayString
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
       "Measured value"
    ::= { sqmScalars 5 }

previousStatus  OBJECT-TYPE
    SYNTAX    INTEGER (0..100)
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
       "Percentage"
    ::= { sqmScalars 6 }

currentStatus  OBJECT-TYPE
    SYNTAX    INTEGER (0..100)
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
       "Percentage"
    ::= { sqmScalars 7 }

thresholdActionArg  OBJECT-TYPE
    SYNTAX DisplayString
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
       "Threshold action On or Off"
    ::= { sqmScalars 8 }


-- ------------------------------------------------------------
--  Parameter Threshold Crossing traps
-- ------------------------------------------------------------

customerParamThresCrossing   OBJECT IDENTIFIER ::= { sqmTraps 2 }
operationParamThresCrossing  OBJECT IDENTIFIER ::= { sqmTraps 3 }

customerSLAViolationStart NOTIFICATION-TYPE
```

```
    OBJECTS {
   -- Monitoring Type (instance or View)
   slaMonitoringType,
   -- Parent SI
        siName, siLabel, siDefName, siDefLabel,
        -- SI/SCI
        sciName, sciLabel, sciType, sciDefName, sciDefLabel,
        -- Measure
   timeStamp, measuredParamValue,
        -- Problem & Implacted SLA
        rootCause, impactedSLANumber, addText,
        slaServiceLevelName, slaServiceLevelLabel, slaName, slaLabel,
        -- Customer
        entityName, entityLabel, entityType,
        -- Component Service Level
        sloComponentServiceLevelName, sloComponentServiceLevelLabel,
        -- Crossed Threshold
        sloName, sloLabel, sloMntrdParamName, sloMntrdParamLabel,
sloMntrdParamType, sloCrossingType,
        sloThreshName, sloThreshLabel, sloThreshReferenceValue,
sloThreshActionName,
   -- threshold Action
   thresholdActionArg
    }
    STATUS     current
    DESCRIPTION
       "This trap is sent each time a parameter cross the associated
threshold and that
        event leads to the violation of Customer SLA(s)"

    ::= { customerParamThresCrossing  1 }

customerSLAViolationEnd NOTIFICATION-TYPE
    OBJECTS {
     -- Monitoring Type (instance or View)
     slaMonitoringType,
     -- Parent SI
        siName, siLabel, siDefName, siDefLabel,
        -- SCI
        sciName, sciLabel, sciType, sciDefName, sciDefLabel,
        -- Measure
     timeStamp, measuredParamValue,
        -- Problem & Implacted SLA
        rootCause, impactedSLANumber, addText,
        slaServiceLevelName, slaServiceLevelLabel, slaName, slaLabel,
        -- Customer
        entityName, entityLabel, entityType,
        -- Component Service Level
        sloComponentServiceLevelName, sloComponentServiceLevelLabel,
        -- Crossed Threshold
        sloName, sloLabel, sloMntrdParamName, sloMntrdParamLabel,
sloMntrdParamType, sloCrossingType,
        sloThreshName, sloThreshLabel, sloThreshReferenceValue,
sloThreshActionName,
   -- threshold Action
   thresholdActionArg
    }
    STATUS     current
    DESCRIPTION
       "This trap is sent each time a threshold is cross down
        indicating the end of customer SLA(s) violation"

    ::= { customerParamThresCrossing 2 }
```

```
customerSLADegradationStart NOTIFICATION-TYPE
    OBJECTS {
     -- Monitoring Type (instance or View)
     slaMonitoringType,
     -- Parent SI
      siName, siLabel, siDefName, siDefLabel,
        -- SCI
        sciName, sciLabel, sciType, sciDefName, sciDefLabel,
        -- Measure
     timeStamp, measuredParamValue,
        -- Problem & Implacted SLA
        rootCause, impactedSLANumber, addText,
        slaServiceLevelName, slaServiceLevelLabel, slaName, slaLabel,
        -- Customer
        entityName, entityLabel, entityType,
        -- Component Service Level
        sloComponentServiceLevelName, sloComponentServiceLevelLabel,
        -- Crossed Threshold
        sloName, sloLabel, sloMntrdParamName, sloMntrdParamLabel,
sloMntrdParamType, sloCrossingType,
        sloThreshName, sloThreshLabel, sloThreshReferenceValue,
sloThreshActionName,
   -- threshold Action
   thresholdActionArg
    }
    STATUS     current
    DESCRIPTION
       "This trap is sent each time a parameter cross the associated
threshold and that
        event leads to the degradation of Customer SLA(s)"

    ::= { customerParamThresCrossing  3 }

customerSLADegradationEnd NOTIFICATION-TYPE
    OBJECTS {
     -- Monitoring Type (instance or View)
     slaMonitoringType,
     -- Parent SI
        siName, siLabel, siDefName, siDefLabel,
        -- SCI
        sciName, sciLabel, sciType, sciDefName, sciDefLabel,
        -- Measure
     timeStamp, measuredParamValue,
        -- Problem & Implacted SLA
        rootCause, impactedSLANumber, addText,
        slaServiceLevelName, slaServiceLevelLabel, slaName, slaLabel,
        -- Customer
        entityName, entityLabel, entityType,
        -- Component Service Level
        sloComponentServiceLevelName, sloComponentServiceLevelLabel,
        -- Crossed Threshold
        sloName, sloLabel, sloMntrdParamName, sloMntrdParamLabel,
sloMntrdParamType, sloCrossingType,
        sloThreshName, sloThreshLabel, sloThreshReferenceValue,
sloThreshActionName,
   -- threshold Action
   thresholdActionArg
    }
    STATUS     current
    DESCRIPTION
       "This trap is sent each time a threshold is cross down
        indicating the end of customer SLA(s) degradation"
```

```
    ::= { customerParamThresCrossing 4 }

operationSLAViolationStart NOTIFICATION-TYPE
    OBJECTS {
     -- Monitoring Type (instance or View)
     slaMonitoringType,
     -- Parent SI
        siName, siLabel, siDefName, siDefLabel,
        -- SCI
        sciName, sciLabel, sciType, sciDefName, sciDefLabel,
        -- Measure
     timeStamp, measuredParamValue,
        -- Problem & Implacted SLA
        rootCause, impactedSLANumber, addText,
        slaServiceLevelName, slaServiceLevelLabel, slaName, slaLabel,
        -- Customer
        entityName, entityLabel, entityType,
        -- Component Service Level
        sloComponentServiceLevelName, sloComponentServiceLevelLabel,
        -- Crossed Threshold
        sloName, sloLabel, sloMntrdParamName, sloMntrdParamLabel,
sloMntrdParamType, sloCrossingType,
        sloThreshName, sloThreshLabel, sloThreshReferenceValue,
sloThreshActionName,
   -- threshold Action
   thresholdActionArg
    }
    STATUS     current
    DESCRIPTION
       "This trap is sent each time a parameter cross the associated
threshold and that
        event leads to the violation of Operation SLA(s)"

    ::= { operationParamThresCrossing  1 }

operationSLAViolationEnd NOTIFICATION-TYPE
    OBJECTS {
     -- Monitoring Type (instance or View)
     slaMonitoringType,
     -- Parent SI
        siName, siLabel, siDefName, siDefLabel,
        -- SCI
        sciName, sciLabel, sciType, sciDefName, sciDefLabel,
        -- Measure
     timeStamp, measuredParamValue,
        -- Problem & Implacted SLA
        rootCause, impactedSLANumber, addText,
        slaServiceLevelName, slaServiceLevelLabel, slaName, slaLabel,
        -- Customer
        entityName, entityLabel, entityType,
        -- Component Service Level
        sloComponentServiceLevelName, sloComponentServiceLevelLabel,
        -- Crossed Threshold
        sloName, sloLabel, sloMntrdParamName, sloMntrdParamLabel,
sloMntrdParamType, sloCrossingType,
        sloThreshName, sloThreshLabel, sloThreshReferenceValue,
sloThreshActionName,
   -- threshold Action
   thresholdActionArg
    }
    STATUS     current
    DESCRIPTION
```

```
       "This trap is sent each time a threshold is cross down
        indicating the end of operation SLA(s) violation"

    ::= { operationParamThresCrossing 2 }

operationSLADegradationStart NOTIFICATION-TYPE
    OBJECTS {
     -- Monitoring Type (instance or View)
     slaMonitoringType,
     -- Parent SI
        siName, siLabel, siDefName, siDefLabel,
        -- SCI
        sciName, sciLabel, sciType, sciDefName, sciDefLabel,
        -- Measure
     timeStamp, measuredParamValue,
        -- Problem & Implacted SLA
        rootCause, impactedSLANumber, addText,
        slaServiceLevelName, slaServiceLevelLabel, slaName, slaLabel,
        -- Customer
        entityName, entityLabel, entityType,
        -- Component Service Level
        sloComponentServiceLevelName, sloComponentServiceLevelLabel,
        -- Crossed Threshold
        sloName, sloLabel, sloMntrdParamName, sloMntrdParamLabel,
sloMntrdParamType, sloCrossingType,
        sloThreshName, sloThreshLabel, sloThreshReferenceValue,
sloThreshActionName,
   -- threshold Action
   thresholdActionArg
    }
    STATUS     current
    DESCRIPTION
       "This trap is sent each time a parameter cross the associated
threshold and that
        event leads to the degradation of Operation SLA(s)"

    ::= { operationParamThresCrossing  3 }

operationSLADegradationEnd NOTIFICATION-TYPE
    OBJECTS {
     -- Monitoring Type (instance or View)
     slaMonitoringType,
     -- Parent SI
        siName, siLabel, siDefName, siDefLabel,
        -- SCI
        sciName, sciLabel, sciType, sciDefName, sciDefLabel,
        -- Measure
     timeStamp, measuredParamValue,
        -- Problem & Implacted SLA
        rootCause, impactedSLANumber, addText,
        slaServiceLevelName, slaServiceLevelLabel, slaName, slaLabel,
        -- Customer
        entityName, entityLabel, entityType,
        -- Component Service Level
        sloComponentServiceLevelName, sloComponentServiceLevelLabel,
        -- Crossed Threshold
        sloName, sloLabel, sloMntrdParamName, sloMntrdParamLabel,
sloMntrdParamType, sloCrossingType,
        sloThreshName, sloThreshLabel, sloThreshReferenceValue,
sloThreshActionName,
   -- threshold Action
   thresholdActionArg
    }
```

```
    STATUS     current
    DESCRIPTION
       "This trap is sent each time a threshold is cross down
        indicating the end of operation SLA(s) degradation"

    ::= { operationParamThresCrossing 4 }

-- ------------------------------
-- Service Objective Status Change
-- ------------------------------

customerSrvObjStatus   OBJECT IDENTIFIER ::= { sqmTraps 4 }
operationSrvObjStatus  OBJECT IDENTIFIER ::= { sqmTraps 5 }

customerServiceObjectiveViolation NOTIFICATION-TYPE
    OBJECTS {
     -- Monitoring Type (instance or View)
     slaMonitoringType,
     -- Parent SI
        siName, siLabel, siDefName, siDefLabel,
        -- Source object is either an instance (SCI or SLA)
        sciName, sciLabel, sciType, sciDefName, sciDefLabel,
        slaName, slaLabel,
        -- Customer
        entityName, entityLabel, entityType,

        -- Status change
     timeStamp, previousStatus, currentStatus
    }
    STATUS     current
    DESCRIPTION
       "This trap is sent each time a Service Objective Status indicates
        a violation of a: SLA/SI/SCI/SI-SLO/SCI-SLO"
    ::= { customerSrvObjStatus 1 }

customerServiceObjectiveDegradation NOTIFICATION-TYPE
    OBJECTS {
     -- Monitoring Type (instance or View)
     slaMonitoringType,
     -- Parent SI
        siName, siLabel, siDefName, siDefLabel,
        -- Source object is either an instance (SCI or SLA)
        sciName, sciLabel, sciType, sciDefName, sciDefLabel,
        slaName, slaLabel,
        -- Customer
        entityName, entityLabel, entityType,

        -- Status change
     timeStamp, previousStatus, currentStatus
    }
    STATUS     current
    DESCRIPTION
       "This trap is sent each time a Service Objectuve Status indicates
        a degradation of a: SLA/SI/SCI/SI-SLO/SCI-SLO"
    ::= { customerSrvObjStatus 2 }

customerServiceObjectiveNormal NOTIFICATION-TYPE
    OBJECTS {
     -- Monitoring Type (instance or View)
     slaMonitoringType,
     -- Parent SI
        siName, siLabel, siDefName, siDefLabel,
        -- Source object is either an instance (SCI or SLA)
```

```
        sciName, sciLabel, sciType, sciDefName, sciDefLabel,
        slaName, slaLabel,
        -- Customer
        entityName, entityLabel, entityType,

        -- Status change
     timeStamp, previousStatus, currentStatus
    }
    STATUS     current
    DESCRIPTION
       "This trap is sent each time a Service Objectuve Status indicates
       that SLA/SI/SCI/SI-SLO/SCI-SLO are OK"
    ::= { customerSrvObjStatus 3 }

operationServiceObjectiveViolation NOTIFICATION-TYPE
    OBJECTS {
     -- Monitoring Type (instance or View)
     slaMonitoringType,
     -- Parent SI
          siName, siLabel, siDefName, siDefLabel,
        -- Source object is either an instance (SCI or SLA)
        sciName, sciLabel, sciType, sciDefName, sciDefLabel,
        slaName, slaLabel,
        -- Operation
        entityName, entityLabel, entityType,

        -- Status change
     timeStamp, previousStatus, currentStatus
    }
    STATUS     current
    DESCRIPTION
       "This trap is sent each time a Service Objective Status indicates
        a violation of a: SLA/SI/SCI/SI-SLO/SCI-SLO"
    ::= { operationSrvObjStatus 1 }

operationServiceObjectiveDegradation NOTIFICATION-TYPE
    OBJECTS {
     -- Monitoring Type (instance or View)
     slaMonitoringType,
     -- Parent SI
      siName, siLabel, siDefName, siDefLabel,
        -- Source object is either an instance (SCI or SLA)
        sciName, sciLabel, sciType, sciDefName, sciDefLabel,
        slaName, slaLabel,
        -- Operation
        entityName, entityLabel, entityType,

        -- Status change
     timeStamp, previousStatus, currentStatus
    }
    STATUS     current
    DESCRIPTION
       "This trap is sent each time a Service Objective Status indicates
        a degradation of a: SLA/SI/SCI/SI-SLO/SCI-SLO"
    ::= { operationSrvObjStatus 2 }

operationServiceObjectiveNormal NOTIFICATION-TYPE
    OBJECTS {
     -- Monitoring Type (instance or View)
     slaMonitoringType,
     -- Parent SI
        siName, siLabel, siDefName, siDefLabel,
        -- Source object is either an instance (SCI or SLA)
```

```
        sciName, sciLabel, sciType, sciDefName, sciDefLabel,
        slaName, slaLabel,
        -- Operation
        entityName, entityLabel, entityType,

        -- Status change
     timeStamp, previousStatus, currentStatus
    }
    STATUS     current
    DESCRIPTION
       "This trap is sent each time a Service Objective Status indicates
       that SLA/SI/SCI/SI-SLO/SCI-SLO are OK"
    ::= { operationSrvObjStatus 3 }


-- ------------------------------
-- Compliance Threshold traps
-- ------------------------------
customerComplianceParamThresCrossing   OBJECT IDENTIFIER ::= { sqmTraps 6 }

customerComplianceViolationStart NOTIFICATION-TYPE
    OBJECTS {
     -- Parent SI
        siName, siLabel, siDefName, siDefLabel,
        -- SCI
        sciName, sciLabel, sciType, sciDefName, sciDefLabel,
        -- Measure
     timeStamp, measuredParamValue,
        -- Problem & Implacted SLA
        rootCause, impactedSLANumber, addText,
        slaServiceLevelName, slaServiceLevelLabel, slaName, slaLabel,
        -- Customer
        entityName, entityLabel, entityType,
        -- Component Service Level
        sloComponentServiceLevelName, sloComponentServiceLevelLabel,
        -- Crossed Threshold
        sloName, sloLabel, sloMntrdParamName, sloMntrdParamLabel,
sloMntrdParamType, sloCrossingType,
        sloThreshName, sloThreshLabel, sloThreshReferenceValue,
sloThreshActionName,
   -- threshold Action
   thresholdActionArg
    }
    STATUS     current
    DESCRIPTION
       "This trap is sent each time a compliance parameter (SLA/SI/SCI)
cross the associated threshold and that
        event leads to the violation of Compliance"

    ::= {customerComplianceParamThresCrossing   1 }

customerComplianceViolationEnd NOTIFICATION-TYPE
    OBJECTS {
     -- Parent SI
        siName, siLabel, siDefName, siDefLabel,
        -- SCI
        sciName, sciLabel, sciType, sciDefName, sciDefLabel,
        -- Measure
     timeStamp, measuredParamValue,
        -- Problem & Implacted SLA
        rootCause, impactedSLANumber, addText,
        slaServiceLevelName, slaServiceLevelLabel, slaName, slaLabel,
        -- Customer
```

```
        entityName, entityLabel, entityType,
        -- Component Service Level
        sloComponentServiceLevelName, sloComponentServiceLevelLabel,
        -- Crossed Threshold
        sloName, sloLabel, sloMntrdParamName, sloMntrdParamLabel,
sloMntrdParamType, sloCrossingType,
        sloThreshName, sloThreshLabel, sloThreshReferenceValue,
sloThreshActionName,
   -- threshold Action
   thresholdActionArg
    }
    STATUS     current
    DESCRIPTION
       "This trap is sent each time a threshold is cross down
        indicating the end of Compliance (SLA/SI/SCI) violation"

    ::= {customerComplianceParamThresCrossing 2 }

customerComplianceDegradationStart NOTIFICATION-TYPE
    OBJECTS {
     -- Parent SI
        siName, siLabel, siDefName, siDefLabel,
        -- SCI
        sciName, sciLabel, sciType, sciDefName, sciDefLabel,
        -- Measure
     timeStamp, measuredParamValue,
        -- Problem & Implacted SLA
        rootCause, impactedSLANumber, addText,
        slaServiceLevelName, slaServiceLevelLabel, slaName, slaLabel,
        -- Customer
        entityName, entityLabel, entityType,
        -- Component Service Level
        sloComponentServiceLevelName, sloComponentServiceLevelLabel,
        -- Crossed Threshold
        sloName, sloLabel, sloMntrdParamName, sloMntrdParamLabel,
sloMntrdParamType, sloCrossingType,
        sloThreshName, sloThreshLabel, sloThreshReferenceValue,
sloThreshActionName,
   -- threshold Action
   thresholdActionArg
    }
    STATUS     current
    DESCRIPTION
       "This trap is sent each time a compliance parameter (SLA/SI/SCI)
cross the associated threshold and that
        event leads to the degradation of Compliance"

    ::= { customerComplianceParamThresCrossing  3 }

customerComplianceDegradationEnd NOTIFICATION-TYPE
    OBJECTS {
     -- Parent SI
        siName, siLabel, siDefName, siDefLabel,
        -- SCI
        sciName, sciLabel, sciType, sciDefName, sciDefLabel,
        -- Measure
     timeStamp, measuredParamValue,
        -- Problem & Implacted SLA
        rootCause, impactedSLANumber, addText,
        slaServiceLevelName, slaServiceLevelLabel, slaName, slaLabel,
        -- Customer
        entityName, entityLabel, entityType,
        -- Component Service Level
```

```
        sloComponentServiceLevelName, sloComponentServiceLevelLabel,
        -- Crossed Threshold
        sloName, sloLabel, sloMntrdParamName, sloMntrdParamLabel,
sloMntrdParamType, sloCrossingType,
        sloThreshName, sloThreshLabel, sloThreshReferenceValue,
sloThreshActionName,
   -- threshold Action
   thresholdActionArg
    }
    STATUS     current
    DESCRIPTION
       "This trap is sent each time a threshold is cross down
        indicating the end of compliance (SLA/SI/SCI) degradation"

    ::= { customerCompliianceParamThresCrossing 4 }


-- ------------------------------
-- Compliance Status traps
-- ------------------------------

customerComplianceStatus  OBJECT IDENTIFIER ::= { sqmTraps 7 }

customerComplianceObjectiveViolation NOTIFICATION-TYPE
    OBJECTS {
     -- Parent SI
        siName, siLabel, siDefName, siDefLabel,
        -- Source object is either an instance (SCI or SLA)
        sciName, sciLabel, sciType, sciDefName, sciDefLabel,
        slaName, slaLabel,
        -- Customer
        entityName, entityLabel, entityType,

        -- Status change
     timeStamp, previousStatus, currentStatus
    }
    STATUS     current
    DESCRIPTION
       "This trap is sent each time a Compliance Status calculated by SQM
indicates
        a violation of the contract for either a: SLA/SI/SCI or one
associated SLO of SI/SCI"
    ::= { customerComplianceStatus  1 }

customerComplianceObjectiveDegradation NOTIFICATION-TYPE
    OBJECTS {
     -- Parent SI
        siName, siLabel, siDefName, siDefLabel,
        -- Source object is either an instance (SCI or SLA)
        sciName, sciLabel, sciType, sciDefName, sciDefLabel,
        slaName, slaLabel,
        -- Customer
        entityName, entityLabel, entityType,

        -- Status change
     timeStamp, previousStatus, currentStatus
    }
    STATUS     current
    DESCRIPTION
       "This trap is sent each time a Compliance Status calculated by SQM
indicates
        an issue for either  a: SLA/SI/SCI or one associated SLO of SI/SCI"
    ::= { customerComplianceStatus  2 }
```

```
customerComplianceObjectiveNormal NOTIFICATION-TYPE
    OBJECTS {
     -- Parent SI
        siName, siLabel, siDefName, siDefLabel,
        -- Source object is either an instance (SCI or SLA)
        sciName, sciLabel, sciType, sciDefName, sciDefLabel,
        slaName, slaLabel,
         -- Customer
        entityName, entityLabel, entityType,

        -- Status change
     timeStamp, previousStatus, currentStatus
    }
    STATUS     current
    DESCRIPTION
       ""
    ::= { customerComplianceStatus  3 }

END
```

Appendix G

Summary of Regular Expression constructs

Characters

Construct	Matches
X	The character x
\\	The backslash character
\0*n*	The character with octal value 0*n* (0 <= *n* <= 7)
\0*nn*	The character with octal value 0*nn* (0 <= *n* <= 7)
\0*mnn*	The character with octal value 0*mnn* (0 <= *m* <= 3, 0 <= *n* <= 7)
\x*hh*	The character with hexadecimal value 0x*hh*
\u*hhhh*	The character with hexadecimal value 0x*hhhh*
\t	The tab character ('\u0009')
\n	The newline (line feed) character ('\u000A')
\r	The carriage-return character ('\u000D')
\f	The form-feed character ('\u000C')
\a	The alert (bell) character ('\u0007')
\e	The escape character ('\u001B')
\c*x*	The control character corresponding to *x*

Character classes

Construct	Matches
[abc]	a, b, or c (simple class)
[^abc]	Any character except a, b, or c (negation)
[a-zA-Z]	a through z or A through Z, inclusive (range)
[a-d[m-p]]	a through d, or m through p: [a-dm-p] (union)
[a-z&&[def]]	d, e, or f (intersection)
[a-z&&[^bc]]	a through z, except for b and c: [ad-z] (subtraction)
[a-z&&[^m-p]]	a through z, and not m through p: [a-lq-z](subtraction)

Predefined character classes

Construct	Matches

Construct	Matches
.	Any character (may or may not match line terminators)
\d	A digit: [0-9]
\D	A non-digit: [^0-9]
\s	A whitespace character: [\t\n\x0B\f\r]
\S	A non-whitespace character: [^\s]
\w	A word character: [a-zA-Z_0-9]
\W	A non-word character: [^\w]

POSIX character classes (US-ASCII only)

Construct	Matches
\p{Lower}	A lower-case alphabetic character: [a-z]
\p{Upper}	An upper-case alphabetic character:[A-Z]
\p{ASCII}	All ASCII:[\x00-\x7F]
\p{Alpha}	An alphabetic character:[\p{Lower}\p{Upper}]
\p{Digit}	A decimal digit: [0-9]
\p{Alnum}	An alphanumeric character:[\p{Alpha}\p{Digit}]
\p{Punct}	Punctuation: One of !"#$%&'()*+,-./:;<=>?@[\]^_`{|}~
\p{Graph}	A visible character: [\p{Alnum}\p{Punct}]
\p{Print}	A printable character: [\p{Graph}]
\p{Blank}	A space or a tab: [\t]
\p{Cntrl}	A control character: [\x00-\x1F\x7F]
\p{XDigit}	A hexadecimal digit: [0-9a-fA-F]
\p{Space}	A whitespace character: [\t\n\x0B\f\r]

Classes for Unicode blocks and categories

Construct	Matches
\p{InGreek}	A character in the Greek block (simple block)
\p{Lu}	An uppercase letter (simple category)
\p{Sc}	A currency symbol
\P{InGreek}	Any character except one in the Greek block (negation)
[\p{L}&&[^\p{Lu}]]	Any letter except an uppercase letter (subtraction)

Boundary matchers

Construct	Matches
^	The beginning of a line
$	The end of a line
\b	A word boundary
\B	A non-word boundary
\A	The beginning of the input
\G	The end of the previous match
\Z	The end of the input but for the final terminator, if any
\z	The end of the input

Greedy quantifiers

Construct	Matches
X?	*X*, once or not at all
*X**	*X*, zero or more times
X+	*X*, one or more times
X{*n*}	*X*, exactly *n* times
X{*n*,}	*X*, at least *n* times
X{*n*,*m*}	*X*, at least *n* but not more than *m* times

Reluctant quantifiers

Construct	Matches
X??	*X*, once or not at all
*X**?	*X*, zero or more times
X+?	*X*, one or more times
X{*n*}?	*X*, exactly *n* times
X{*n*,}?	*X*, at least *n* times
X{*n*,*m*}?	*X*, at least *n* but not more than *m* times

Possessive quantifiers

Construct	Matches
X?+	*X*, once or not at all
*X**+	*X*, zero or more times
X++	*X*, one or more times
X{*n*}+	*X*, exactly *n* times
X{*n*,}+	*X*, at least *n* times
X{*n*,*m*}+	*X*, at least *n* but not more than *m* times

Logical operators

Construct	Matches
XY	*X* followed by *Y*
X|*Y*	Either *X* or *Y*
(*X*)	X, as a capturing group

Back references

Construct	Matches
n	Whatever the n^{th} capturing group matched

Quotation

Construct	Matches
\	Nothing, but quotes the following character
\Q	Nothing, but quotes all characters until \E
\E	Nothing, but ends quoting started by \Q

Special constructs (non-capturing)

Construct	Matches
(?:*X*)	*X*, as a non-capturing group
(?idmsux-idmsux)	Nothing, but turns match flags on - off
(?idmsux-idmsux:*X*)	*X*, as a non-capturing group with the given flags on - off
(?=*X*)	*X*, via zero-width positive lookahead
(?!*X*)	*X*, via zero-width negative lookahead
(?<=*X*)	*X*, via zero-width positive lookbehind
(?<!*X*)	*X*, via zero-width negative lookbehind
(?>*X*)	*X*, as an independent, non-capturing group

Backslashes, escapes, and quoting

The backslash character ('\') serves to introduce escaped constructs, as defined in the table above, as well as to quote characters that otherwise would be interpreted as unescaped constructs. Thus the expression \\ matches a single backslash and \{ matches a left brace.

It is an error to use a backslash prior to any alphabetic character that does not denote an escaped construct; these are reserved for future extensions to the regular-

expression language. A backslash may be used prior to a non-alphabetic character regardless of whether that character is part of an unescaped construct.

Backslashes within string literals in Java source code are interpreted as required by the Java Language Specification as either Unicode escapes or other character escapes. It is therefore necessary to double backslashes in string literals that represent regular expressions to protect them from interpretation by the Java bytecode compiler. The string literal `"\b"`, for example, matches a single backspace character when interpreted as a regular expression, while `"\\b"` matches a word boundary. The string literal `"\(hello\)"` is illegal and leads to a compile-time error; in order to match the string `(hello)` the string literal `"\\(hello\\)"` must be used.

Character Classes

Character classes may appear within other character classes, and may be composed by the union operator (implicit) and the intersection operator (&&). The union operator denotes a class that contains every character that is in at least one of its operand classes. The intersection operator denotes a class that contains every character that is in both of its operand classes.

The precedence of character-class operators is as follows, from highest to lowest:

1	Literal escape	`\x`
2	Grouping	`[...]`
3	Range	`a-z`
4	Union	`[a-e][i-u]`
5	Intersection	`[a-z&&[aeiou]]`

Note that a different set of metacharacters are in effect inside a character class than outside a character class. For instance, the regular expression `.` loses its special meaning inside a character class, while the expression `-` becomes a range forming metacharacter.

Line terminators

A *line terminator* is a one- or two-character sequence that marks the end of a line of the input character sequence. The following are recognized as line terminators:

A newline (line feed) character (`'\n'`),

A carriage-return character followed immediately by a newline character (`"\r\n"`),

A standalone carriage-return character (`'\r'`),

A next-line character (`'\u0085'`),

A line-separator character (`'\u2028'`), or

A paragraph-separator character (`'\u2029`).

By default, the regular expressions `^` and `$` ignore line terminators and only match at the beginning and the end, respectively, of the entire input sequence.

Groups and capturing

Capturing groups are numbered by counting their opening parentheses from left to right. In the expression `((A)(B(C)))`, for example, there are four such groups:

1	`((A)(B(C)))`
2	`(A)`
3	`(B(C))`
4	`(C)`

Group zero always stands for the entire expression.

Capturing groups are so named because, during a match, each subsequence of the input sequence that matches such a group is saved. The captured subsequence may be

used later in the expression, via a back reference, and may also be retrieved from the matcher once the match operation is complete.

The captured input associated with a group is always the subsequence that the group most recently matched. If a group is evaluated a second time because of quantification then its previously-captured value, if any, will be retained if the second evaluation fails. Matching the string `"aba"` against the expression `(a(b)?)+`, for example, leaves group two set to `"b"`. All captured input is discarded at the beginning of each match.

Groups beginning with `(?` are pure, *non-capturing* groups that do not capture text and do not count towards the group total.


invent


Java™
COMPATIBLE