HP OpenView
Service Quality Manager

Script Gateway

Installation, Configuration and User's Guide

Edition: 1.4

December 2006

Legal notices

Warranty

The information contained herein is subject to change without notice. The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

License requirement, and U.S. Government legend

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

Copyright notices

Trademark notices

Adobe®, Acrobat®, and PostScript® 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.

UNIX® is a registered trademark of The Open Group.

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.

Origin

Printed in France.

Contents

Preface

This document is intended to provide an overview of the HP OpenView Service Quality Manager (SQM) Script Gateway and describes how to:

- Install the Script Gateway
- Set up the Script Gateway
- Start and stop the Script Gateway
- Write scripts for running by the Script Gateway

Intended audience

This document is intended for OpenView SQM administrators and system software developers.

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
- Shell scripting

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 chapter 3.2.1, “Software and Hardware requirements”.

Typographical conventions

The following typographical conventions have been used throughout this document.

`Courier` font:

- Source code and examples of file contents
- Commands that you enter on the screen
- Pathnames

Italic text:

- Filenames, programs and parameters in the text
- The names of other documents referred to in this guide

Bold text:

- New terms
- Emphasized words
- Keyboard key names

Associated documents

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

Support

Please visit our HP OpenView web site at HP OpenView

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

Terms and acronyms

A list of terms and acronyms used frequently in this document is provided in the Glossary at the back of this document.

Chapter 1

Introduction

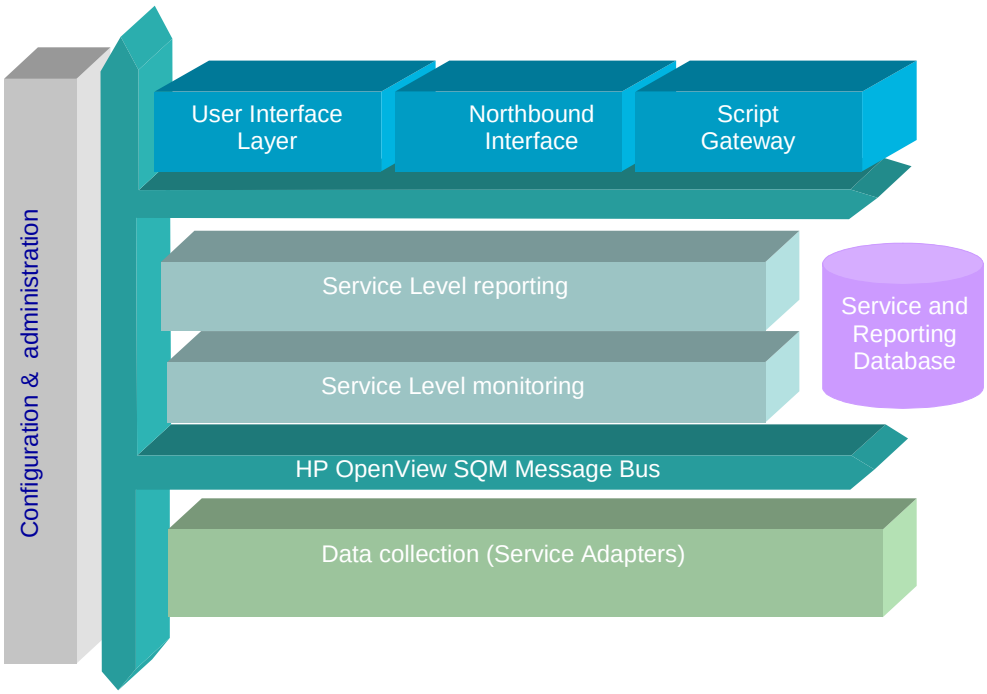
1.1 OpenView Service Quality Manager

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 impacts 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).

OpenView SQM runs under the HP implementation of Unix®, HP-UX. HP-UX, which is compatible with various industry standards, is based on the UNIX System V Release 4 operating system and includes important features from the Fourth Berkeley Software Distribution.

Figure 1 OpenView SQM main components



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

1.2 The Script Gateway

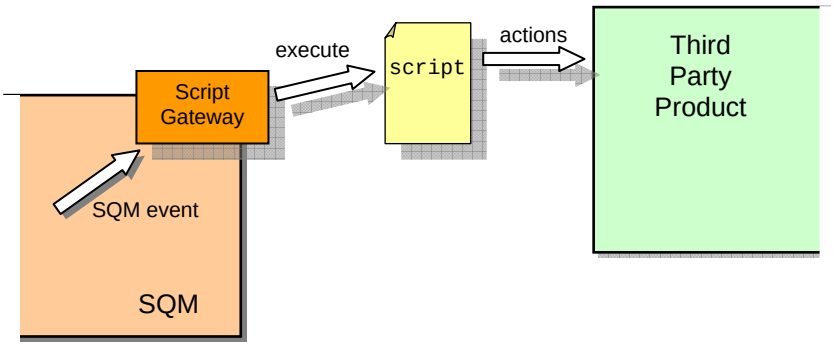
The goal of the SQM Script Gateway is to serve as a bridge between SQM northbound and another product.

The Script Gateway offers an easy and fast way to integrate SQM with another product on HP-UX, using shell script.

The Script Gateway can trigger script execution upon each of the following, in real time:

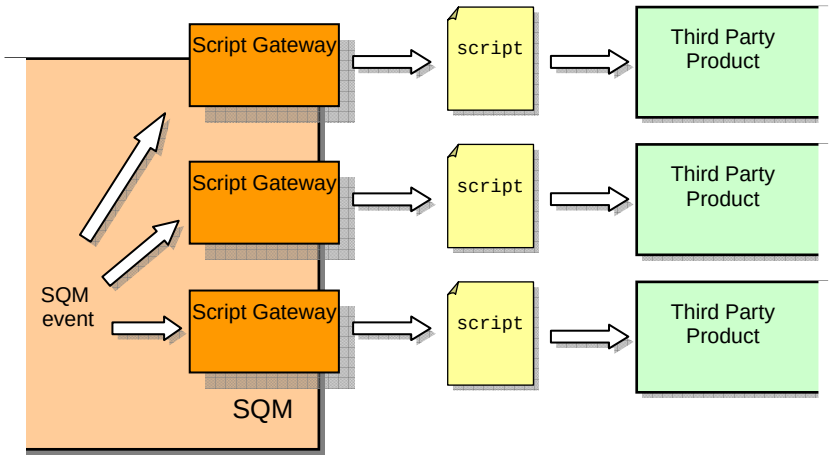
- SLA Service violations and degradations
- Service Objective status changes
- SLA Compliance violation and degradation
- Compliance Objective status

Interaction with the Third Party Product can then be controlled entirely by the triggered script.

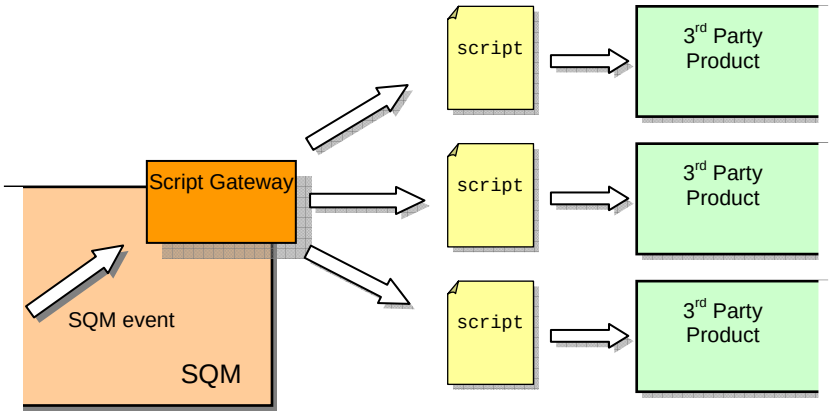


Flexibility

Several different Script Gateways, each dedicated to a different Third-Party Product, can be used at the same time.

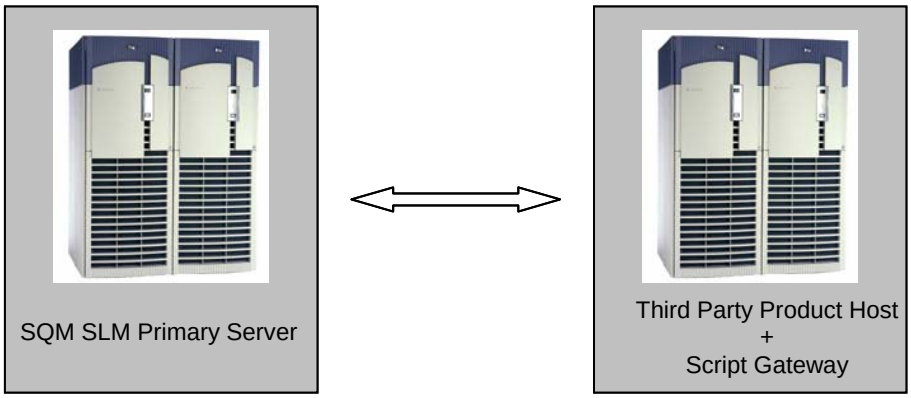


A Script Gateway can be configured to run different scripts when different SQM events occur (depending on the event type, service affected, or other factor). One Script Gateway can therefore be dedicated to several Third Party Products.



Architecture overview

The Script Gateway must be able to access and execute all scripts it launches and must also be able to interact with the Third Party Product concerned. Where possible, installation of the Script Gateway on the Third Party Product host server is therefore recommended. If this host is different than the SQM SLM Primary Server, the host is usually called the SQM Secondary Host.



Chapter 2

Script Gateway features

The Script Gateway is designed to trigger execution of a script when an SQM Service Monitoring Status (SMS) event occurs. It can trigger this when either of the following occurs:

- Parameter Threshold Crossing event
- Service Objective Status Change event (for SLAs and SIs)

The Script Gateway can execute the following types of scripts:

- Asynchronous scripts
- Synchronous scripts
- Daemon scripts

All of the script execution information is contained in a Script Gateway XML properties file. The contents of this file, which is called the **Script Configuration** file, is described in appendix D, “Script Configuration XML file”.

Asynchronous scripts

When a script is executed asynchronously, the Script Gateway launches it but does not check on its progress or any responses from it. This means that the Gateway does not check on script activity or retrieve any status information.

Synchronous scripts

When a script is executed synchronously, the Script Gateway launches it and then tracks its processing, to detect when the script ends. This type of script execution can be used to incorporate a timeout mechanism limiting the length of time a script can run.

Daemon scripts

Daemon scripts involve a significant startup cost (if the script must open a session to communicate with the Third Party Product, for example). Scripts of this type are launched only once (the first time it is triggered) and then continue running until the Script Gateway is shut down.

When the Script Gateway runs a Daemon script, it transmits the SQM events to the script through its standard input.

2.1 Linking script execution to a Service Parameter’s Threshold Crossing Event

The link between execution of a script and a Service Parameter’s Threshold Crossing event is declared in the SQM model’s Service Level Objectives (SLOs).

An SLO is used to monitor a Service Parameter, in which the parameter value is compared with one or more thresholds.

The SLO's **Action Executor** parameter is used to specify what actions must be triggered when a parameter crosses a threshold. For more information, see the "Service Level" chapter in the *HP OpenView Service Quality Manager Information Modeling Reference Guide*.

To trigger script execution when a parameter crosses an SLO threshold, the SLO definition must contain the following action executor information:

- `Executor`: <*script_gateway_executor_name*>
  The Script Gateway executor name is defined in the gateway's Script Configuration information in the central repository as described in chapter 3.6.1.1, "General configuration properties".
- `Action`: <*script_to_trigger*>
  The name of the script triggered by the event. This script must be declared in the gateway's Script Configuration file as described in appendix D, "Script Configuration XML file".
- `Additional Info`: <*free text argument*>
  The text contained in this argument is supplied to the script when it is executed.
- `On Argument`: <*free text argument*>
  This argument can be used to specify a special operation that must be executed by the script when the threshold is crossed upwards (meaning the Quality of Service is decreasing.)
- `Off Argument`: <*free text argument*>
  This argument can be used to specify a special operation that must be executed by the script when the threshold is crossed downwards (meaning the Quality of Service is increasing).

Important

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

2.2 Information supplied to scripts when a Threshold Crossed event occurs

When an SQM Threshold Crossed event is received, the Script Gateway retrieves the information contained in the event and prepares it for passing to the script.

This preparation includes handling the different components involved in the SLA and SI. If a Threshold Crossed event affecting several components is received, the Script Gateway must prepare a list of all components involved and perform 'fan-out processing' if necessary.

2.2.1 Fan-out processing

Fan-out processing is a special form of processing used to 'split' a Threshold Crossed event it receives into several events, each of which relates to a different component branch (SLA or SI).

When fan-out processing is enabled, the Script Gateway processes events as follows:

- If a Threshold Crossed event that does not affect several components is received, the fan-out process is ignored.

- If a Threshold Crossed event affecting several components is received, the Script Gateway simulates an event for each SLA and SI affected.

  Each of these simulated events only provides the subset of information required by the component concerned.

Regardless of whether fan-out processing is enabled or not, the original Threshold Crossed event is always managed by the gateway, however.

Fan-out processing is controlled by the *fanout* Script Gateway property in the SQM central repository (see chapter 3.6.1.1, “General configuration properties”).

Note

By default, fan-out processing is enabled.

2.2.2 Information supplied when an event occurs

Scripts are supplied with all the following information when they are triggered by a *ThresholdCrossed* event.

NotificationType

The notification type is defined by which type of SQM event occurred.

Possible values are *ThresholdCrossed*, and *ThresholdCrossedCompliance*.

2.2.2.1 Information on the current action executor

ExecutorName

The executor name is the gateway’s executor name defined in the gateway configuration.

ActionInfo

The action info contains additional information for the executor. This information is read from the SLO definition.

ActionName

The action name is the name of the script that is triggered by the event.

Warning

The **script name** is different than the **script file**. The script name is the name of the action that is triggered, whereas the script file is the file path of the script that is executed when it is triggered.

A script file can be linked to several different script names, but a script name cannot be linked to more than one script file.

ActionOff

The action off is the executor’s *off argument*. It is read from the SLO definition.

ActionOn

The action on is the executor’s *on argument*. It is read from the SLO definition.

2.2.2.2 Information on the crossed threshold and its parent SLO

AcquisitionTimestamp

The acquisition timestamp is the GMT timestamp recorded when the parameter value crosses the threshold (2004-01-27T09:21:07 for example.)

MonitoredParameterLabel & MonitoredParameterName

The monitored parameter label and monitored parameter name contain the label and the name (identifier) of the parameter value that crossed the threshold.

The monitored parameter label is optional.

MonitoredParameterValue

The monitored parameter value is the value of the parameter that crossed the threshold.

CustomerDependentFlag

The customer dependent flag indicates whether the parameter that crossed the threshold is customer dependent or not.

The flag can contain the value *True*, or *False*.

OTLabel & OTName

The OT label and OT name are the label and name (identifier) of the crossed Objective Threshold.

The OT label is optional.

OTValue

The OT value is the value linked to the crossed Objective Threshold.

isDegradation

This flag indicates whether the crossed threshold is a degradation threshold or not.

The flag can contain the value *True*, or *False*.

isViolation

This flag indicates whether the crossed threshold is a violation threshold or not.

The flag can contain the value *True*, or *False*.

isClear

This flag indicates whether the threshold was crossed downwards in the last Threshold Crossed event (meaning that the SLO is no longer violated or degraded) or not.

The flag can contain the value *True*, or *False*.

SLOLabel & SLOName

The SLO label and SLO name are the label and name (identifier) of the Service Level Objective to which the crossed threshold belongs.

The SLO label is optional.

CrossingType

The crossing type indicates which type of threshold crossing event is linked to the SLO.

It can contain the value *Up*, *Down*, *Equal*, *NotEqual*, or *IsValued*.

SLLabel & SLName

The SL label and SL name are the label and name (identifier) of the parent Service Level of the SLO that owns the crossed threshold.

If the root component flag contains the value *True*, the SLO belongs directly to the SL; if the root component flag contains the value *False*, the CSL that owns the SLO belongs to the SL (for an explanation of *RootComponentFlag*, see chapter 2.2.2.4, “Information on affected components”).

The SL label is optional.

CSLLabel & CSLName

The CSL label and CSL name are the label and name (identifier) of the parent Component Service Level of the SLO to which the crossed threshold belongs.

The information contained in these fields is provided only if the root component flag contains the value *False* (for an explanation of *RootComponentFlag*, see chapter 2.2.2.4, “Information on affected components”).

The CSL label is optional.

2.2.2.3 Information on the event

EventTypeDegradation

The event type degradation flag indicates whether the event relates to the start of degraded service, the end of degraded service, or neither of these.

The *EventTypeDegradation* flag can contain the value *Start*, or *End*.

If the event type does not relate to the start or the end of degraded service, the *EventTypeDegradation* flag is not passed.

EventTypeViolation

The event type violation flag indicates whether the event represents the beginning of violation, the end of violation, or neither of these.

The *EventTypeViolation* flag can contain the value *Start* or *End*.

If the event type does not relate to either the start or end of violation, the *EventTypeViolation* flag is not passed.

QosStatus

The QoS status indicates the ‘direction’ in which the Quality of Service is moving.

The *QosStatus* field can contain the value *Decreasing*, or *Increasing*.

isQosDecreasing

This flag contains the same information as the *QosStatus* field.

When the Quality of Service is decreasing, *isQosDecreasing* contains the value *True*; if not, it contains the value *False*.

isDataForCompliance

This flag indicates whether the Threshold Crossed event relates to component compliance or not.

The flag can contain the value *True*, or *False*.

2.2.2.4 Information on affected components

Which information on affected components is passed depends on how many components are affected and whether fan-out processing is enabled or not.

CompoundMessage

The compound message flag indicates whether the information provided relates to more than one affected component or not.

If the provided information relates to more than one affected component, the flag contains the value *True*; if not, it contains the value *False*.

isSharedSCI

This flag indicates whether the affected SCI is shared by several SIs (and/or SLAs) or not.

The flag can contain the value *True* or *False.*

RootComponentFlag

The root component flag indicates whether the parameter that crossed the threshold belongs to an SI or an SCI.

When the parameter belongs to an SI, the flag contains the value *True*.

When the parameter belongs to an SCI, the flag contains the value *False.*

SDLabel & SDName

The SD label and SD name fields contain the label and name (identifier) of the Service Definition affected by the crossed threshold.

The SD label is optional.

SCDLabel & SCDName

The SCD label and SCD name fields contain the label and name (identifier) of the Service Component Definition affected by the crossed threshold.

The values contained in these fields are processed only if the root component flag contains the value *False.*

The SCD label is optional.

SCILabel & SCIName

The SCI label and SCI name fields contain the label and name (identifier) of the Service Component Instance affected by the crossed threshold.

The information contained in these fields is passed only if the root component flag contains the value *False.*

The SCI label is optional.

CustomerLabel & CustomerName

The customer label and customer name fields contain the label and name (identifier) of the customer corresponding to the parameter that crossed the threshold.

The customer label is optional.

SLALabel & SLAName

The SLA label and SLA name fields contain the label and name (identifier) of the Service Level Agreement affected by the crossed threshold.

This information is provided only if it relates to a single SLA (in other words, the *isSharedSCI* flag contains the value *False* or the *CompoundMessage* flag contains the value *False*).

If not, the affected SLAs are named in the *ImpactedSLAList* or *StructuredImpactedSLAList* fields.

The SLA label is optional.

isOperationalSLA

When the information in the SLA name and label fields are passed, this flag indicates whether the specified SLA is 'Operational' or not.

If the flag contains the value *True,* the SLA is 'Operational'.

If the flag contains the value *False,* the SLA is 'Customer'.

SILabel & SIName

The SI label and SI name fields contain the label and name (identifier) of the Service Instance affected by the crossed threshold.

The information in this field is passed only if the information relates to a single SLA (in other words, the *isSharedSCI* flag contains the value *False* or the *CompoundMessage* flag contains the value *False*).

If not, the affected SLAs are named in the *ImpactedSLAList* or *StructuredImpactedSLAList* fields.

The SI label is optional.

ImpactedSLAListDetails

The impacted SLA list details flag indicates whether the gateway configuration allows the list of affected SLAs to be passed to the scripts or not. (See also the *NoImpactedSlaGeneration* property in chapter 3.6.1.2, "Advanced configuration properties".)

StructuredImpactedSlaMode

The structured impacted SLA mode flag indicates whether this mode has been enabled in the Script Configuration or not.

If the flag contains the value *True,* the list of affected SLAs is passed in structured form, to simplify parsing. If it contains the value *False*, the list of affected SLAs is passed in simplified form for display.

ImpactedSLAListQty

The impacted SLA list quantity field indicates how many SLAs are affected by the crossed threshold.

The information contained in this field is passed only if the *CompoundMessage* flag contains the value *True.*

ImpactedSLAList

The impacted SLA list provides a simplified list of all SLAs affected by the crossed threshold.

This list, which can be displayed directly, is in the following format:

`SLALabel(SLAName): CustLabel(CustName) {SILabel(SIName), SILabel(SIName)}; SLALabel(SLAName)`, etc.

The information contained in this field is passed only if the *ImpactedSLAListDetails* flag contains the value *True*, the *CompoundMessage* flag contains the value *True*, and the *StructuredImpactedSlaMode* flag contains the value *False*.

StructuredImpactedSLAList

The structured impacted SLA list provides a list of all SLAs affected by the crossed threshold, in a structured format.

The format of this list is structured to simplify parsing. Its specific form depends on the chosen script execution mode (Asynchronous, Synchronous, or Daemon).

The information contained in this field is passed only if the *ImpactedSLAListDetails* field contains the value *True*, the *CompoundMessage* field contains the value *True*, and the *StructuredImpactedSlaMode* field contains the value *True*.

2.3 Linking execution of a script to an Objective Status Change event

How you link execution of a script to an Objective Status Change event depends on whether the objective status filtering function has been enabled. This is done by setting the gateway's *ObjectiveStatusFiltering* flag to *True* (see chapter 3.6.1.1, "General configuration properties").

2.3.1 When Objective Status filtering is enabled

This is the default Script Gateway configuration.

When Objective Status Filtering is enabled, the link between execution of a script and an Objective Status Change must be declared in the Script Configuration file.

In the Script Configuration file, each script configuration has an **SLA Status trigger** and an **SI Status trigger**.

Those triggers are regular expressions (represented by 'RegExp' in the following table) that the Script Gateway uses to determine whether an Objective Status Change event matches the script configuration or not. When it matches the configuration, the corresponding script is executed.

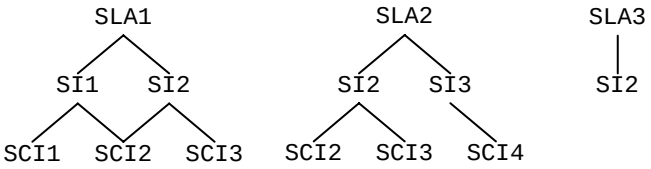
The following table shows how the Script Gateway responds to script SLA and SI triggers.

SLA Status trigger	SI Status trigger	Triggered events
-	-	None
-	RegExp Y	Configuration error; the script configuration is ignored

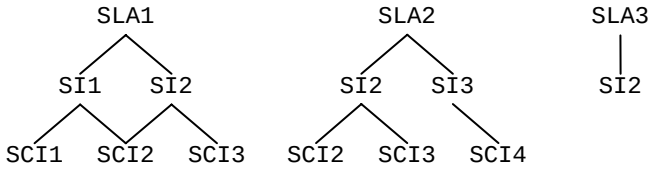
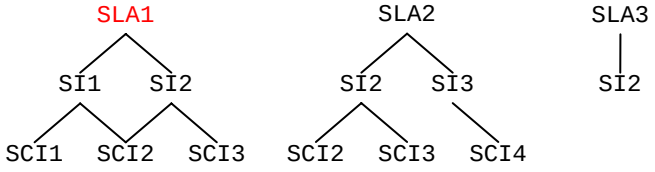
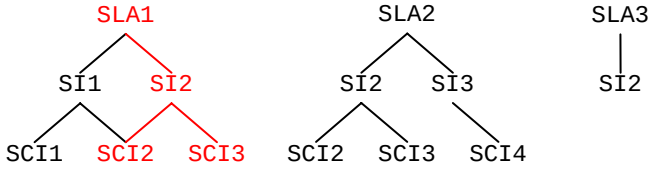
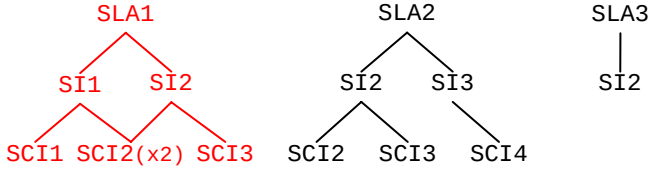
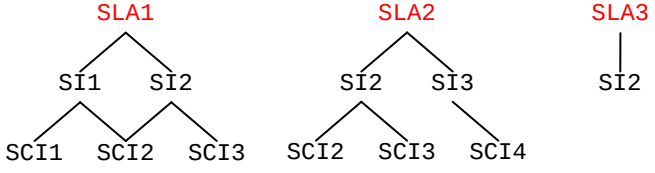
SLA Status trigger	SI Status trigger	Triggered events
RegExp X	-	Only SLA Status changes that match RegExp X
RegExp X	RegExp Y	SLA Status changes that match RegExp X + all SI status changes that match RegExp Y in these SLAs + all child SCI status changes in these SIs.

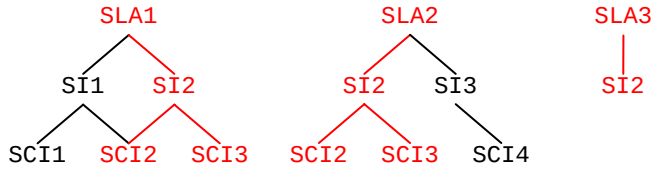
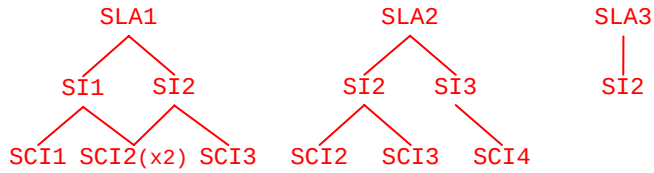
Example

The following status trees:



Result in the following script configurations:

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

SLA Status	SI Status trigger	Triggered events (shown in red)
SLA.*	SI2	
SLA.*	SI.*	

Note

Unlike when scripts are triggered by Threshold Crossed events, the Objective Status filtering mechanism does not use the *ActionExecutorName* event to determine whether a script must be triggered or not.

2.3.2 When Objective Status Filtering is disabled

When Objective Status Filtering is disabled, the same script is triggered, regardless of which Objective Status has changed.

The script must be declared with the name *StatusChanged* in the gateway's Script Configuration file.

When Objective Status Filtering is disabled, the script is triggered only if the Status Change event is the result of a Threshold Crossing parameter on an SLO that specifies an Action Executor declared in the gateway configuration. For further details, see *ActionExecutorName* in chapter 3.6.1.1, "General configuration properties".

Note

In this mode, you must ensure that script execution is triggered by a change in a component's (SLA or SI) Objective Status. Every parameter monitored for this component must have an Action Executor for the Script Gateway to use.

2.4 Information supplied to scripts when Objective Status Change events occur

When the Script Gateway receives an SQM Objective Status Change event, it retrieves the information contained in the event and prepares it for supplying to the script.

The SQM event may contain Objective Status Change information for several components (SLA, SI or SCI). In such cases, one task performed by the Script Gateway consists in simulating a separate SQM event for each component, to simplify the triggered script's processing.

The Script Gateway then determines the Status Change severity for each simulated Objective Status Change event. The relationships between the Objective Status and the different severities are specified in the gateway configuration described in chapter 3.6.1.1, "General configuration properties".

2.4.1 Description of the supplied information

A detailed list of all information supplied to a script when its triggering Objective Status Change event occurs is shown below.

NotificationType

The notification type indicates which type of SQM event has occurred.

The field can contain the value *StatusChanged*, or *StatusChangedCompliance*.

AcquisitionTimestamp

The timestamp is the GMT timestamp generated when the status changes (2004-01-27T09:21:07, for example)

ExecutorName

The executor name is that defined in the gateway's script configuration.

isDataForCompliance

This flag indicates whether the objective status change relates to a **compliance** objective or not.

ObjectiveStatus

The objective status contains the value of the component objective status, which can be between 0 and 1. (*0.9*, for example).

CurrentOS

The current OS parameter contains the same information as *ObjectiveStatus*, except that it is formatted as a percentage of between 0% and 100% (*90%*, for example).

PreviousObjectiveStatus

The previous objective status contains the component's previous objective status, which can be between 0 and 1. (*0.8*, for example).

This information is not supplied when *isDataForCompliance* contains the value *True*.

PreviousOS

The previous OS parameter contains the same information as *PreviousObjectiveStatus*, except that it is formatted as a percentage of between 0% and 100% (*80%*, for example).

This information is not supplied when *isDataForCompliance* contains the value *True*.

CustomerLabel & CustomerName

The customer label and customer name fields contain the label and name (identifier) of the customer whose Objective Status has changed.

The customer label is optional.

RootComponentFlag

The root component flag indicates whether the changed objective status concerns an SI or another component.

When the changed objective status concerns an SI, the flag contains the value *True*.

When the changed objective status concerns another component (an SLA or SCI), the flag contains the value *False*.

SCDLabel & SCDName

The SCD label and SCD name fields contain the label and name (identifier) of the Service Component Definition for the SCI concerned.

These fields are processed only if the root component flag contains the value *False.*

The SCD label is optional.

SCILabel & SCIName

The SCI label and SCI name fields contain the label and name (identifier) of the Service Component Instance whose status has changed.

These fields are processed only if the root component flag contains the value *False.*

The SCI label is optional.

SDLabel & SDName

The SD label and SD name fields contain the label and name (identifier) of the Service Definition for the SI concerned.

The SD label is optional.

SILabel & SIName

The SI label and SI name fields contain the label and name (identifier) of either the Service Instance whose status has changed, or the parent of the SCI whose status has changed.

The component whose status has changed is an SI if the root component flag contains the value *True*.

The SI label is optional.

SLALabel & SLAName

The SLA label and SLA name fields contain the label and name (identifier) of either the Service Level Agreement whose status has changed, or the parent of the component whose status has changed.

The component whose status has changed is an SLA if the SI and SCI information is not supplied.

The SLA label is optional.

isOperationalSLA

This flags indicates whether the SLA concerned is ‘operational’ or not.

If the flag contains the value *True,* the SLA is ‘operational’.

If the flag contains the value *False,* the SLA is ‘customer’.

Severity

This field contains the severity of the Objective Status Change.

The Script Gateway calculates the severity based on both the value of the current (and possibly the previous) *ObjectiveStatus* and the severities definition in the gateway’s configuration in the central repository (see chapter 3.6.1.1, “General configuration properties”.)

The field can contain any of the following values:

- *critical*
- *major*
- *minor*
- *warning*
- *normal*

Chapter 3

Script Gateway lifecycle

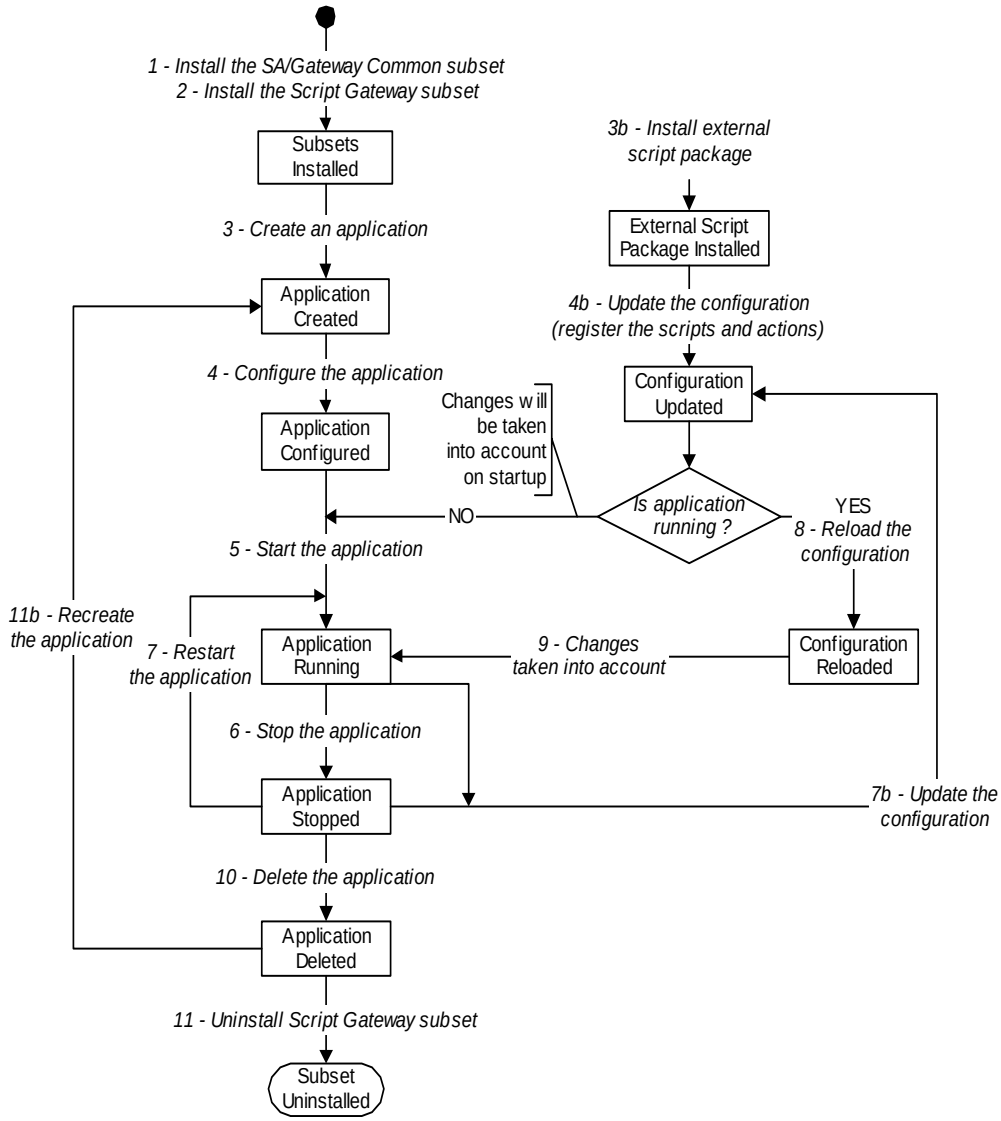
The Script Gateway lifecycle is divided into the following phases:

- Installing the SA/Gateway Common subset (1)
- Installing the Script Gateway subset (2)
- Creating the Script Gateway application (3)
- Configuring the application (4)
- Starting up the application (5)
- Shutting down the application (6)
- Deleting the application (10)
- Uninstalling the Script Gateway subset (11)

These can be combined with the following alternative phases:

- Installing the external scripts package (3b)
- Registering the external scripts (4b)
- Updating the application configuration (7b)
- Hot-reloading the application configuration (8)
- Restarting the application (7)
- Recreating the application (11b)

The sequences in which these steps may be performed are summarized in the following diagram.



3.1 Required environment

The SQM environment must be defined before you can install, set up or configure the Script Gateway.

To define the SQM environment, you must source the *temip_sc_env.sh* file located in the SQM data directory *($TEMIP_SC_VAR_HOME)*, by entering the following command.

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

Note

The *temip_sc_env.sh* script is created when the SQM Kernel is set up.

3.2 Installing the Script Gateway

This chapter describes how to install the Script Gateway on an HP system running HP-UX V11i. After you have completed the installation, follow the instructions in chapter 3.6, “Configuring a Script Gateway application”, to configure the application.

3.2.1 Software and Hardware requirements

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

3.2.2 Installing the software

This section describes how to install the Script Gateway on the SQM SLM Primary Server if the Third Party Product is running on the same host.

Important

The Script Gateway must be able to access and execute the script concerned and be able to interact with the Third Party Product. Using an architecture in which the Script Gateway is installed on the Third Party Product host is therefore recommended.

Appendix C explains how to install the Script Gateway on a host other than the SQM SLM Primary Server.

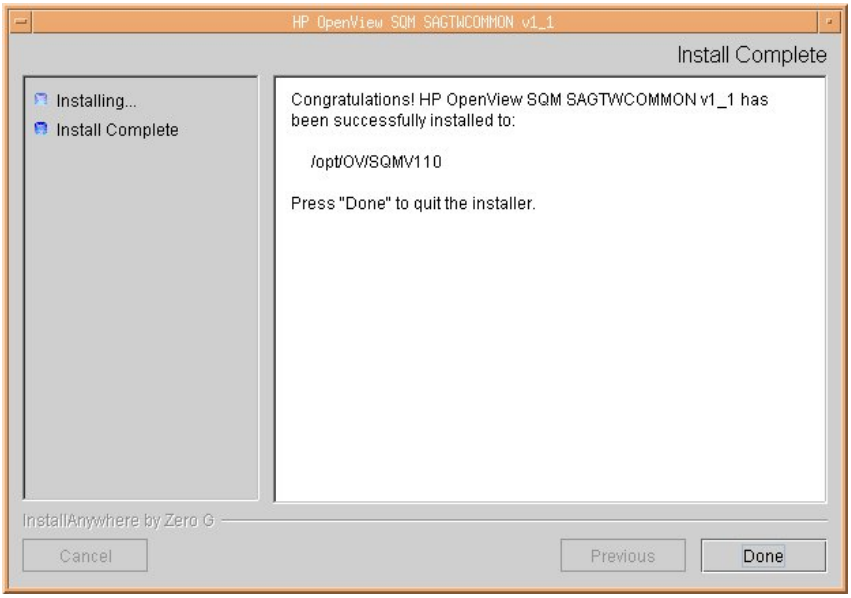
This chapter assumes that the OV SQM Kernel is already installed and configured on the primary host. The procedure for installing and configuring the SQM Kernel is described in the *HP OpenView Service Quality Manager Installation Guide*.

Before you can install the Script Gateway, the Service Adapters and Gateways Common subset, must have been installed already and must have been updated with all patches.

3.2.2.1 Installing the Service Adapters and Gateways Common subset

To install the Service Adapters and Gateways Common subset (if it has not already been installed), do the following.

1. Log on as **root** user, and then load the SQM environment variables as described in chapter 3.1, “Required environment”.
2. Mount the Service Adapters and Gateways CD-ROM on your system.
3. Go to the `SQM-1.20.00-SAGTW/HPUX` directory.
4. Execute the *SQMSAGTWCOMMON-1.20.00.bin* installer.



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

Important

Do not forget to install any SA or Gateway Common subset patches.

3.2.2.2 Installing the Script Gateway subset

To install the Script Gateway, do the following.

1. Log on as **root** user, and then load the SQM environment variables as described in chapter 3.1, “Required environment”.
2. Mount the Service Adapters and Gateways CD-ROM on your system.
3. Go to the `SQM-1.20.00-SAGTW/HPUX` directory.
4. Execute the *SQMGTWSCRIPT-1.20.00.bin* installer.
5. To end the installation process, click **Done**.

Important

Do not forget to install any Script Gateway subset patches.

3.3 Uninstalling the Script Gateway

Caution

No Script Gateways must be running when you uninstall the Script Gateway.

To uninstall the *SQMGTWSCRIPT-1.20.00* subset, do the following.

1. Log on as **root** user, and then load the SQM environment variables as described in chapter 3.1, “Required environment”.
2. Go to the following directory: `$TEMIP_SC_HOME`
3. Display the Uninstaller window by executing the `./Gateways/Script/v1_2/Uninstaller_GTWSCRIPT/Uninstall_GTWSCRIPT` command.
4. Click **Uninstall**.

5. When the uninstallation process is complete, click **Done**.

3.4 Creating a Script Gateway application

After the Script Gateway has been installed, you must create a Script Gateway application.

To create a Script Gateway application, do the following.

1. Log on as **root** user, and then load the SQM environment variables as described in chapter 3.1, “Required environment”.
2. Go to the `$TEMIP_SC_HOME/Gateways/Script/v1_2/bin` directory.
3. Execute the `temip_sc_gtw_setup -gateway create` command to display and reply to the displayed command line prompts as follows:

```
Please enter the platform [slmv12]: slmv12

Please enter the director [gateway]: gateway

Please enter the application [Script Gateway]: Script Gateway
```

The Script Gateway application that is created belongs to the specified director (by default, the director is **gateway**) and has the specified name.

To check that the application has been created properly, you can log on as **sqmadm** user and load the SQM environment variables, and then execute the following command:

```
# temip_sc_show_application -platform <platform name> -director
  <director name> -application <application name>
```

For more information on using this command, see the *HP OpenView Service Quality Manager Administration Guide*.

3.5 Deleting a Script Gateway application

Caution

Before you delete a Script Gateway application, you must shut down the application concerned.

To delete a Script Gateway application, do the following.

1. Log on as **root** user.
2. Load the SQM environment variables as described in chapter 3.1, “Required environment”.
3. Go to the `$TEMIP_SC_HOME/Gateways/Script/v1_2/bin` directory.
4. Execute the `/temip_sc_gtw_setup -gateway delete` command to display and reply to the displayed command line prompts as follows:

```
Please enter the platform [slmv12]: slmv12

Please enter the director [gateway]: gateway

Please enter the application [Script Gateway]: Script Gateway
```

When this procedure is completed, the Script Gateway application is deleted.

To check that the application has indeed been deleted, you can log on as **sqmadm** user and load the SQM environment variables, and then execute the following command:

```
# temip_sc_show_application -platform <platform_name> -director
 <director_name> -application <application_name>
```

3.6 Configuring a Script Gateway application

Script Gateway applications are configured in two stages:

1. Updating the SQM central repository.
2. Registering the scripts. This consists in:
   - Declaring the scripts in the gateway's Script Configuration file
   - Declaring the actions in the SLA Admin UI

The Script Gateway can also be used to hot-reload a configuration.

3.6.1 The Script Gateway central repository

Script Gateway applications store some of their configuration properties in the SQM central repository (TIBCO repository). These properties can be accessed through the TIBCO Designer.

For details of how to launch and use the TIBCO Designer, see the *HP OpenView Service Quality Manager Administration Guide*.

The Script Gateway configuration is stored in the following places in the TIBCO repository:

- `/screpos/ServiceCenter/Gateways/Script/v1_2/<`*ApplicationName*`>`
- `/screpos/ServiceCenter/Gateways/Script/v1_2/<`*ApplicationName*`>_config`

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

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

3.6.1.1 General configuration properties

Script Gateway general properties are held in the `<`*ApplicationName*`>_config` adapter's **Extended Properties**.

Table 1 below lists all Script Gateway general properties.

Note:

All variables with values starting and ending with "%%" in the following table refer to TIBCO global variables. See the *HP OpenView Service Quality Manager Administration Guide* for details of their values.

Table 1 Script Gateway general properties

Variable Name	Default Values
MessageMapping	Both
ComplianceMapping	Threshold
ComplianceEnabled	False
ActionExecutorName	ScriptGateway
ObjectiveStatusFiltering	True

Variable Name	Default Values
ServiceAlarmMapping	
-> *FanOut*	True
-> SeverityCode	
-> -> *ThresholdDefaultSeverity*	Major (not used by script gateways)
-> -> *ObjectiveStatusCriticalSeverity*	%%SCStateViolationLevel%%
-> -> *ObjectiveStatusMajorSeverity*	0.4
-> -> *ObjectiveStatusMinorSeverity*	0.6
-> -> *ObjectiveStatusWarningSeverity*	%%SCStateDegradationLevel%%
ServiceComplianceAlarmMapping	
-> *FanOut*	True
-> SeverityCode	
-> -> *ThresholdDefaultSeverity*	Major (not used by script gateways)
-> -> *ObjectiveStatusCriticalSeverity*	%%SCStateViolationLevel%%
-> -> *ObjectiveStatusMajorSeverity*	0.4
-> -> *ObjectiveStatusMinorSeverity*	0.6
-> -> *ObjectiveStatusWarningSeverity*	%%SCStateDegradationLevel%%

Each parameter in the above table is described in detail below.

MessageMapping

This parameter controls how the Script Gateway maps SQM events.

It can contain any of the following values:

- *Threshold*: Actions are then mapped only when a Threshold Crossing violation or degradation event occurs.
- *Status*: Actions are then mapped only when a service objective degradation or violation event occurs.
- *Both*: Actions are then mapped for Threshold Crossing violation and degradation events as well as for service objective degradation or violation (status) events.

This parameter is associated with the TIBCO configuration parameter *MessageMapping*. By default, its value is *Both*.

ComplianceMapping

This parameter controls how the Script Gateway maps SQM Compliance events.

It can contain any of the following values:

- *Threshold*: Actions are then mapped only when a compliance Threshold Crossing violation or degradation event occurs.
- *Status*: Actions are then mapped only when a compliance service objective degradation or violation event occurs.
- *Both*: Actions are then mapped for compliance Threshold Crossing violation or degradation events as well as for compliance service objective degradation or violation (status) events.

This parameter is associated with the TIBCO configuration parameter *ComplianceMapping*. By default, its value is *Threshold*.

ComplianceEnabled

This parameter specifies whether the Script Gateway processes compliance events or not.

This parameter is associated with the TIBCO configuration parameter *ComplianceEnabled*. By default, its value is *False*.

ActionExecutorName

The gateway processes only OpenView SQM degradation and violation events with action executor names specified in this parameter. This parameter is set using the SLA Admin UI (also called the 'Admin UI') when service levels are created. For more details, see the *OpenView SQM Overview*.

This parameter is associated with the TIBCO configuration parameter *ActionExecutorName*. By default, its value is *ScriptGateway*.

Important

The *ActionExecutorName* parameter 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.

Note

If the *ObjectiveStatusFiltering* parameter contains the value *True*, the *ActionExecutorName* parameter is not used in deciding whether to discard SQM events or not.

ObjectiveStatusFiltering

This parameter is used to specify whether the 'advanced filtering' mechanism is enabled or not.

- If the mechanism is enabled, the Gateway does not use the *ActionExecutorName* parameter to decide whether to include Objective Status Change events or not. Instead, it delegates this task to a filtering mechanism that analyzes all SQM Objective Status change events.

  The Script Gateway filtering mechanism is described in chapter 2.3.1, "When Objective Status filtering is enabled".

- If this mechanism is disabled, the Gateway uses the *ActionExecutorName* parameter to decide whether to include the Objective Status change events or not. In this case, the default status change action is performed as described in chapter 2.3.2, "When Objective Status Filtering is disabled").

This parameter is associated with the TIBCO configuration parameter *ObjectiveStatusFiltering*. By default, its value is *True*.

FanOut (ServiceAlarmMapping)

This parameter is used to specify whether the "fan-out" feature is enabled or not. If this feature is enabled, the Gateway maps one event for each SLA, SI, and SCI instance affected by the Threshold Crossing event for shared SCIs.

This parameter is associated with the TIBCO configuration parameter *FanOut*. By default, its value is *True*.

ThresholdDefaultSeverity (ServiceAlarmMapping)

This parameter defines what severity is assigned to a Script Gateway action performed when a Threshold Crossing violation or degradation event occurs.

Note

This parameter is not used by Script Gateways.

ObjectiveStatusCriticalSeverity (ServiceAlarmMapping)

This parameter specifies the threshold objective beyond which a service objective is considered violated. The corresponding Script Gateway action is classified as being of 'Critical' severity

This parameter is associated with the TIBCO configuration parameter *ObjectiveStatusCriticalSeverity*. By default, it is set to the value contained in the global OpenView SQM variable *SCStateViolationLevel* (by default, its value is *0.2*).

ObjectiveStatusMajorSeverity (ServiceAlarmMapping)

This parameter specifies the threshold objective beyond which a service is considered degraded. The corresponding Script Gateway action is classified as being of 'Major' severity.

This parameter is associated with the TIBCO configuration parameter *ObjectiveStatusMajorSeverity*. By default, its value is *0.4*.

ObjectiveStatusMinorSeverity (ServiceAlarmMapping)

This parameter specifies the threshold objective beyond which a service is considered degraded. The corresponding Script Gateway action is classified as being of 'Minor' severity.

This parameter is associated with the TIBCO configuration parameter *ObjectiveStatusMinorSeverity*. By default, its value is *0.6*.

ObjectiveStatusWarningSeverity (ServiceAlarmMapping)

This parameter specifies the threshold objective beyond which a service is considered as degraded. The corresponding Script Gateway action is categorized as being of 'Warning' severity.

This parameter is associated with the TIBCO configuration parameter *ObjectiveStatusWarningSeverity*, and is set, by default' value to the value contained in the global OpenView SQM variable *SCStateDegradationLevel* (by default, its value is *0.8*).

FanOut (ServiceComplianceAlarmMapping)

This parameter specifies whether the compliance 'fan-out' feature is enabled or not. If this feature is enabled, the Gateway maps one compliance event for each SLA, SI, and SCI instance affected by the Threshold Crossing event in the case of shared SCIs.

This parameter is associated with the TIBCO configuration parameter *FanOut*. By default, its value is *True*.

ThresholdDefaultSeverity (ServiceComplianceAlarmMapping)

This parameter specifies what default severity must be assigned to actions performed by the Script Gateway when a compliance Threshold Crossing violation or degradation event occurs.

Note

This parameter is not used by Script Gateways.

ObjectiveStatusCriticalSeverity (ServiceComplianceAlarmMapping)

This parameter specifies the threshold objective beyond which a service compliance objective is considered violated. The corresponding Script Gateway service compliance action is classified as being of 'Critical' severity.

This parameter is associated with the TIBCO configuration parameter *ObjectiveStatusCriticalSeverity*. By default, its value is the global OpenView SQM variable '*SCStateViolationLevel'* (by default, its value is *0.2*).

ObjectiveStatusMajorSeverity (ServiceComplianceAlarmMapping)

This parameter specifies the threshold objective beyond which service compliance is considered degraded. The corresponding Script Gateway service compliance action is classified as being of 'Major' severity.

This parameter is associated with the TIBCO configuration parameter *ObjectiveStatusMajorSeverity*. By default, its value is *0.4*.

ObjectiveStatusMinorSeverity (ServiceComplianceAlarmMapping)

This parameter specifies the threshold objective beyond which service compliance is considered degraded. The corresponding Script Gateway service compliance action is classified as being of 'Minor' severity.

This parameter is associated with the TIBCO configuration parameter *ObjectiveStatusMinorSeverity*. By default, its value is *0.6*.

ObjectiveStatusWarningSeverity (ServiceComplianceAlarmMapping)

This parameter specifies the threshold objective beyond which service compliance is considered degraded. The corresponding Script Gateway service compliance action is classified as being of 'Warning' severity.

This parameter is associated with the TIBCO configuration parameter *ObjectiveStatusWarningSeverity*. By default, it is set to the value contained in the global OpenView SQM variable *SCStateDegradationLevel* (by default, its value is *0.8*).

3.6.1.2 Advanced configuration properties

Script Gateway advanced properties are held in the `<`*ApplicationName*`>_config` adapter's **Extended Properties**.

Table 2 below lists all Script Gateway advanced properties.

Note:

All variables with values starting and ending with "`%%`" in the following table refer to TIBCO global variables. See the *HP OpenView Service Quality Manager Administration Guide* for details of their values.

Table 2 Script Gateway advanced properties

Variable Name	Default Values
ListenerProcessorThreadNb	1
XMLValidationMode	%% SCXmlMsgValidation %%
QuietMode	false
NoFileGeneration	true
NoImpactedSlaGeneration	true
EventQueue	
->EventQueueSize	100
->MaxThreshold	0.9
->MinThreshold	0.8

Each parameter in the above table is described in detail below.

ListenerProcessorThreadNumber

This parameter defines how many concurrent listener threads can run in the component at the same time.

This parameter is associated with the TIBCO configuration parameter *ListenerProcessorThreadNb*. By default, its value is *1*. You must not change this parameter unless you are asked to do so by OpenView SQM support.

XMLValidation

This parameter serves only in debugging, and is used to check the validity of all XML messages received by a Script Gateway component. You must not change this parameter unless you are asked to do so by OpenView SQM support.

This parameter is associated with the TIBCO configuration parameter *XMLValidationMode*. By default, it is set to the value contained in the global OpenView SQM variable *SCXmlMsgValidation* (by default, its value is *False*).

QuietMode

This field serves only in debugging and is used to prevent messages from being published by the Script Gateway. You must not change this parameter unless you are asked to do so by OpenView SQM support.

This parameter is associated with the TIBCO configuration parameter *QuietMode*. By default, its value is *False*.

NoFileGeneration

This parameter overrides the script configuration *file.mode* property, which is described in appendix D, “Script Configuration XML file”. If it contains the value *True*, it serves to prevent an SMS event file being created when the script is executed, even if the script configuration specifies that the file must be created.

This parameter is associated with the TIBCO configuration parameter *NoFileGeneration*. By default, its value is *True*.

NoImpactedSlaGeneration

When this parameter contains the value *True*, it serves to prevent the script from being given the list of SLAs, SIs, and Customers affected by a Threshold Crossed event.

When a Threshold Crossed event affects SLAs, SIs, or Customers, the script is given this list of affected components. If many SLAs, SIs, or Customers may be simultaneously affected in the SQM model concerned, this may result in an environment variable exceeding the script shell environment variable’s maximum size, causing an error. In this case, *NoImpactedSlaGeneration* must be set to *True*.

This parameter is associated with the TIBCO configuration parameter *NoImpactedSlaGeneration.* By default, its value is *True*.

EventQueue

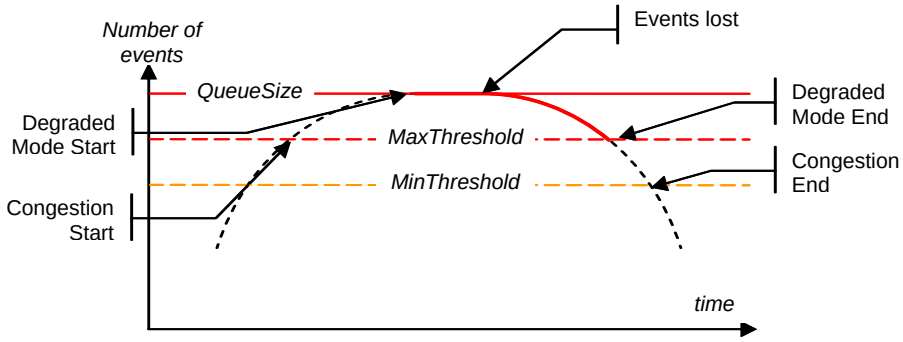
The Script Gateway makes use of a message queuing mechanism to manage SMS messages from the time they are received to the time the script concerned is called. To avoid congestion, the Script Gateway includes a threshold mechanism (Max and Min) to control the queue's capacity.

When the queue's occupation level reaches its Max threshold, the Script Gateway logs a warning that congestion is beginning to occur.

When the queue's capacity reaches its maximum size, the gateway enters in degraded mode and begins to drop incoming SQM events. These events are lost.

When the queue occupation level drops to the Max threshold, the gateway leaves degraded mode and resumes managing incoming SQM events.

When the queue capacity drops to the Min threshold, the gateway logs an alert indicating that the congestion has ended.



EventQueueSize (EventQueue)

This parameter defines the size of the SQM event queue, in terms of the number of events it can hold.

This parameter is associated with the TIBCO configuration parameter *EventQueueSize.* By default, its value is *100.*

MaxThreshold (EventQueue)

This parameter defines the SQM event queue's Maximum threshold of messages.

This value is a factor of the *EventQueueSize*, and it is between the *MinThreshold* value and 1.

This parameter is associated with the TIBCO configuration parameter *MaxThreshold.* By default, its value is *0.9.*

MinThreshold (EventQueue)

This parameter defines the SQM Event queue's Minimum threshold of messages.

This value is a factor of the value contained in *EventQueueSize*, and it is between 0 and the *MaxThreshold* value.

This parameter is associated with the TIBCO configuration parameter *MinThreshold.* By default, its value is *0.8.*

3.6.1.3 Log and trace properties

Log and trace related properties are held in the /<*ApplicationName*> adapter's **trace_sink** and **log_sink** parameters (. . . /<*ApplicationName*>/Advanced/Log Sinks).

Table 3 below lists all Script Gateway logging and tracing properties.

Table 3 Script Gateway's log and trace properties

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

Each parameter in the above table is described in detail below.

FileCount

This parameter is used to set the maximum number of trace or log files.

This parameter is associated with the TIBCO configuration parameter *FileCount.* By default, its value is *10*.

FileLimit

This parameter is used to set the maximum size of trace or log files. The size is specified in bytes.

This parameter is associated with the TIBCO configuration parameter *FileLimit.* By default, its value is *10000000* (bytes) for traces and *1000000* for logs.

AppendMode

This parameter is used to specify whether traces or logs can be appended to existing files after the application is restarted, or not.

This parameter is associated with the TIBCO configuration parameter *AppendMode.* By default, its value is *True*.

3.6.2 Registering scripts

This section explains how to register a script for triggering by the Script Gateway.

There are two stages in the script registration process:

- Declaring the script in the gateway's Script Configuration file. This step is compulsory.
- Declaring the new action in the list of available actions in the SLA Admin UI. This step is optional.

Both these steps are described in detail below.

3.6.2.1 Declaring scripts in the Script Gateway

All scripts that can be triggered by the Script Gateway must be declared in the gateway's Script Configuration file. This step is called 'script registration'.

To declare a script in the Script Configuration file, you must ensure the following.

- The OpenView SQM environment variables must have been set. (See chapter 3.1 for details of how to set the SQM environment.)

- You must be logged in as **sqmadm** user.

Next, do the following.

1. Go to the `$TEMIP_SC_HOME/Gateways/Script/v1_2/bin` directory.
2. Execute the `temip_sc_gtw_setup -script create <`*script_name*`>` command.
3. Enter the required platform, director and application names as described in appendix E, "*temip_sc_gtw_setup* tool").
4. Enter the requested script configuration properties as described in appendix D, "Script Configuration XML file").

3.6.2.2 Declaring scripts in the SLA Admin UI

Scripts must be declared in the SLA Admin UI's list of available actions before they can be selected to perform an action by setting an Action Executor on a SLO (see chapter 2.1, "Linking script execution to a Service Parameter's Threshold Crossing Event").

Each SLA Admin UI has its own list of available actions. This list is located in the following XML file (on the SLA Admin host):

```
$TEMIP_SC_HOME/UI/SLAClient/properties/AvailableActions.xml
```

Extracting the Script Gateway's actions

To declare a Script Gateway's actions in the `AvailableActions.xml` file, you must firstly extract all available actions from the gateway's Script Configuration file.

To extract all of the gateway's actions, you must ensure the following.

- The OpenView SQM Kernel must be running.
- You must be logged on as **sqmadm** user.
- The OpenView SQM environment variables must have been set. (See chapter 3.1 for details of how to set the SQM environment.)

Next, do the following.

1. Go to the `$TEMIP_SC_HOME/Gateways/Script/v1_2/bin` directory.
2. Execute the `temip_sc_gtw_setup -script uia [<`*script_name_list*`>]` command.
3. Enter the required platform, director and application names as described in appendix E, "*temip_sc_gtw_setup* tool").

An Addon xml file is generated at the following location.

```
$TEMIP_SC_VAR_HOME/Gateways/Script/config/UIAdmin_<day_i
n_year>_<hh>_<mm>_<ss>.xml
```

An example of a generated file is shown below.

```
<?xml version="1.0" encoding="UTF-8"?><!DOCTYPE
AvailableScripts>
<AvailableScripts>
  <executor executor.name="ScriptGateway" executor.label="Scrip
tGateway Script Gateway">
    <Desc>Script Gateway is a shell script executor. It maps SM
S messages and activates script execution to perform 'action' t
owards third party</Desc>
    <action action.name="SendMail" action.label="Send Email on
SLA status change" />
    <action action.name="Dump" action.label="Script that dumps
all SLA SQM events in file" />
  </executor>
</AvailableScripts>
```

Adding the extracted actions to the SLA Admin UI's Available Actions file

The second operation consists in adding the extracted actions to each SLA Admin UI's Available Actions file.

To add the extracted actions to an SLS Admin UI's Available Actions file, do the following for each SLA Admin UI.

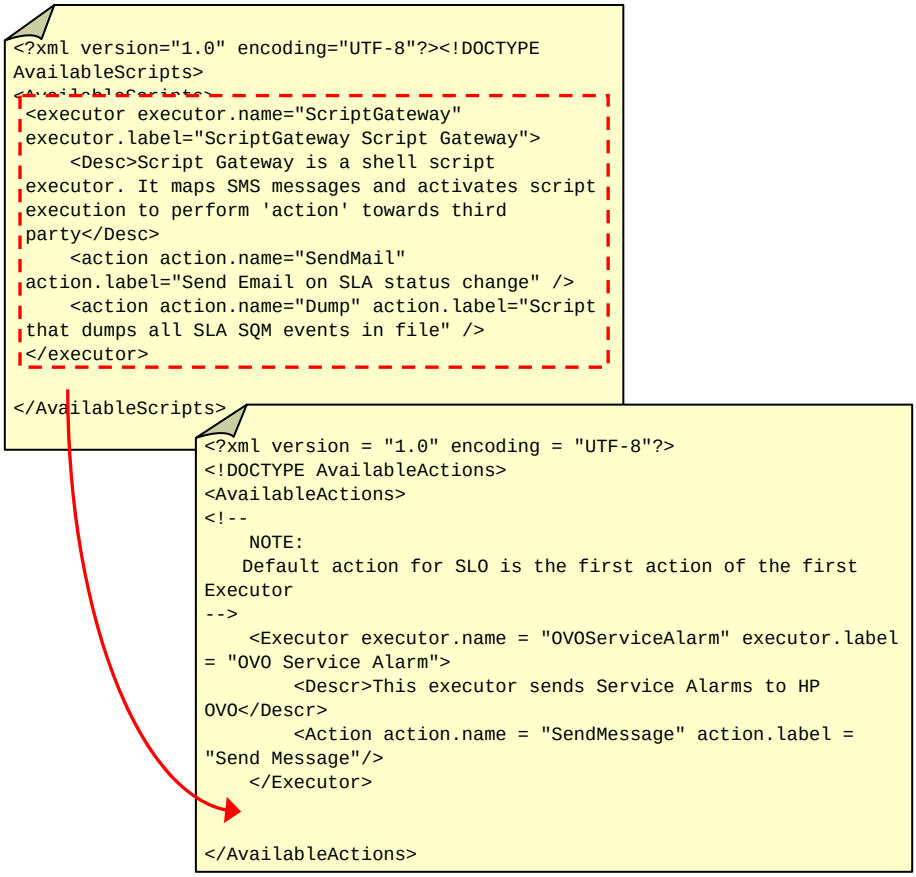
1. Copy the generated Addon file onto the SLA Admin UI's host computer.
2. Edit the SLA Admin UI's Available Actions file.

An example of the file's contents is shown below.

```
<?xml version = "1.0" encoding = "UTF-8"?>
<!DOCTYPE AvailableActions>
<AvailableActions>
<!--
    NOTE:
   Default action for SLO is the first action of the first Exec
utor
-->
    <Executor executor.name = "OVOServiceAlarm" executor.label
= "OVO Service Alarm">
        <Descr>This executor sends Service Alarms to HP OVO</De
scr>
        <Action action.name = "SendMessage" action.label = "Sen
d Message"/>
    </Executor>

</AvailableActions>
```

3. Copy the 'Executor' definition from the Addon file to the Available Actions file as shown in the following diagram.



Note

The SLA Admin UI must be restarted in order to introduce the new actions.

3.6.3 Reloading the configuration

Script Gateway configuration changes made in the SQM central repository and the Script Configuration file are normally introduced when the Script Gateway is started.

You can also introduce the changes while the Script Gateway is running, by running its ReloadConfig AMI, however.

Running the Script Gateway ReloadConfig AMI

Before running the Script Gateway ReloadConfig AMI, you must ensure the following.

- The OpenView SQM Kernel must be running.
- You must be logged on as **sqmadm** user.
- The OpenView SQM environment variables must have been set. (See chapter 3.1 for details of how to set the SQM environment.)

Next, do the following.

1. Go to the `$TEMIP_SC_HOME/Gateways/Script/v1_2/bin` directory.
2. Execute the `temip_sc_gtw_setup -gateway reload` command and enter the requested platform, director and application names as described in appendix E, “*temip_sc_gtw_setup* tool”).

What the Script Gateway does when the configuration is reloaded

When the Script Gateway ReloadConfig AMI is run, the gateway reloads its configuration from the SQM central repository and reloads its Script Configuration file.

Effects on the event queue

When the configuration is reloaded, its *EventQueueSize*, *MaxThreshold* and *MinThreshold* configuration parameters are used to redefine the event queue's maximum size, Max threshold and Min threshold.

This can have the following results:

- If the *EventQueueSize* parameter value is increased:
  The queue size is updated and is effective immediately.
- If the *EventQueueSize* parameter value is decreased:
  The queue size is updated and is effective immediately, but the queue's current content is not modified, meaning that the number of events in the queue might temporarily exceed the *EventQueueSize*.
- If the *MaxThreshold* parameter value is increased:
  The new value is effective immediately. If degraded mode was enabled and the new value exceeds the number of events in the queue, degraded mode is disabled.
- If the *MaxThreshold* parameter value is decreased:
  The new value is effective immediately.
- If the *MinThreshold* parameter value is increased or decreased:
  The new value is effective immediately.

Effect on daemon scripts

Any daemon scripts running may be affected when the gateway's Script Configuration file is reloaded.

The effects may be as follows:

- If the script no longer exists in the Script Configuration file:
  The daemon script is stopped.
- If the script has been disabled:
  The daemon script is stopped.
- If the script execution context has changed (if the *script.file*, *script.dir*, or the *ScriptEnv* parameter has changed, for example):
  The daemon script is restarted.

3.7 Starting a Script Gateway application

Before starting a Script Gateway application, you must ensure the following:

- The OpenView SQM Kernel must be running.
- You must be logged on as **sqmadm** user.
- The OpenView SQM environment variables must have been set. (See chapter 3.1 for details of how to set the SQM environment.)

To start a Script Gateway application, do the following.

1. Go to the $TEMIP_SC_HOME/Gateways/Script/v1_2/bin directory.

2. Execute the `following` command:

```
> temip_sc_gtw_setup –gateway start
```

3. Enter the appropriate platform, director and application names as described in appendix E, "*temip_sc_gtw_setup* tool").

Note

This command functions identically to the `temip_sc_start_application` command:

```
> temip_sc_start_application -platform <platform> -director <g
ateway_director> -application <gateway_application_name>
```

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

When it starts, the Script Gateway loads its Script Configuration file and checks the configuration of all scripts.

If any of them is not valid (if one of the status trigger regular expressions is not valid, for example), the script configuration is ignored and an error is logged. This does not prevent the gateway starting, however.

3.8 Stopping a Script Gateway

Before stopping a Script Gateway, you must ensure the following:

- The OpenView SQM Kernel must be running.
- You must be logged on as **sqmadm** user.
- The OpenView SQM environment variables must have been set. (See chapter 3.1 for details of how to set the SQM environment.)

To stop a Script Gateway, do the following.

1. Go to the $TEMIP_SC_HOME/Gateways/Script/v1_2/bin directory.
2. Execute the following command:

```
> temip_sc_gtw_setup –gateway stop
```

3. Enter the appropriate platform, director and application names as described in appendix E, "*temip_sc_gtw_setup* tool").

Note

This command functions identically to the `temip_sc_stop_application` command shown below:

```
> temip_sc_stop_application -platform <platform> -director <ga
teway_director> -application <gateway_application_name>
```

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

This command does the following:

- Stops the Script Gateway's SQM event listener
- Completes all management tasks on the events stored in its event queue
- Stops all daemon scripts and waits for all synchronous scripts to finish running

Depending on how many SQM events are in the queue, the Script Gateway may take longer than expected to stop. If it takes longer than 3 minutes, however, the gateway is stopped automatically by the SQM kernel.

3.9 Monitoring a Script Gateway

The **TIBCO HawkDisplay** application can be used to perform Script Gateway administration and monitoring tasks.

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

Chapter 4

Writing scripts

This chapter describes each step in the process of writing scripts for triggering by the Script Gateway.

4.1 Determining what SQM information is sent to a Third Party Product

The first step in the script creation process consists in deciding which items of SQM information must be sent to the Third Party Product. Which items of information are available depends on the type of SQM event involved. For a description of which information is available for each type, see chapter 2.2, “Information supplied to scripts when a Threshold Crossed event occurs”, and chapter2.4, “Information supplied to scripts when Objective Status Change events occur”.

4.2 Determining how the script forwards this information a Third Party Product

In this step, you must choose how the script must be executed.

Script Gateway proposes three modes of execution: asynchronous, synchronous and daemon.

Which mode of execution you should choose depends on how the script must interact with the Third Party Product, and what constraints are involved.

This choice is important because it affects how the script will manage the SQM events it handles.

The advantages and disadvantages of each mode are listed in the following table.

Script Type	Advantages/Disadvantages
Asynchronous	Advantages
	Low resource use
	Disadvantages
	Less control over script execution Script return status is not handled Script standard and error outputs are not handled Script is launched for each event
	Typical use
	‘Fire and forget,’ such as event logging and tracing

Script Type	Advantages/Disadvantages
Synchronous	Advantages
	Great control over script execution Script return status is handled Script standard and error outputs are traced (in debug mode) Script is launched for each event
	Disadvantages
	Considerable resource use
	Typical use
	Situations in which the execution status must be handled
Daemon	Advantages
	Script is started once only Events are communicated in messages pushed as standard script input (requiring fewer resources than when the script is launched). Script standard and error outputs are traced (in debug mode)
	Disadvantages
	Script must handle pushed messages
	Typical use
	Interfacing with Third Party Products requires considerable resources in the startup and/or login phase (for Service Desk incident creation, for example).

4.3 Determining the script execution context

- The Script Gateway must be able to access and execute all scripts that it triggers.
  - To be accessed, the scripts must be located in directories that can be seen and accessed by the **sqmadm** user. The default location for scripts is as follows:

    ```
    $TEMIP_SC_HOME/Gateways/Script/v1_2/scripts/
    ```

    The *script.file* property in the Script Configuration file can be specified relative to this directory. For further details, see appendix D, “Script Configuration XML file”.

    Alternatively, however, you can specify the full path in the *script.file* property (beginning with the ‘/’ character).
  - To be executed by the Script Gateway, the scripts must be executable by the **sqmadm** user. You must bear this in mind if the script must perform operations with another user.
- The default directory for scripts executed by Script Gateway is `$TEMIP_SC_HOME`.

  You can use the *script.dir* property to specify another directory that can be reached by the Script Gateway, however.

4.4 Deciding how to manage information the gateway passes to the script

You should decide this based on which execution mode you chose.

4.4.1 Synchronous mode, and asynchronous mode

In these modes, the script is passed information using shell script environment variables. These variables can be accessed in the same way as any other shell script environment variables.

As well as passing these variables, the Script Gateway passes the environment variables specified in the Script Configuration file's *ScriptEnv* field. These are:

- *$TEMIP_SC_HOME*
- *$TEMIP_SC_VAR_HOME*
- *$SCRIPTGTW_HOME* (the Script Gateway's release directory, in *TEMIP_SC_HOME*)
- *$SCRIPTGTW_DATA* (the Script Gateway's data directory, in *TEMIP_SC_VAR_HOME*).

When the `file.mode` script is enabled, the *$FileName* variable indicates where the generated SMS file must be stored.

An example of the generated SMS file is shown in appendix G, "SMS message example".

The Script Gateway also passes the script the following information in the form of arguments:

- *type=<script_type>*, which specifies in which mode the script must be executed (Synchronous, Asynchronous or Daemon).
- The *debug* flag, which specifies whether debug mode must be enabled when the script runs, or not.

Information on affected SLAs

If the Script Configuration file specifies that structured information on affected SLAs is provided when a Threshold Crossed event occurs, do the following to access this information.

1. Retrieve the number of SLAs affected. This information is contained in the *$ImpactedSLAListQty* environment variable.
2. Retrieve each affected SLA's SLA name, SLA label, Customer name and Customer label. This information is contained in the following environment variables:
   - *$ImpactedSLAList_<SLA_index>_SLALabel*
   - *$ImpactedSLAList_<SLA_index >_SLALabel*
   - *$ImpactedSLAList_<SLA_index>_CustomerName*
   - *$ImpactedSLAList_<SLA_index>_CustomerLabel*, where *SLA_index* is the affected SLA's counter value (which is between 1 and *$ImpactedSLAListQty*)
3. Retrieve the number of SIs affected in the current affected SLA. This information is contained in the *$ImpactedSLAList_<SLA_index>_SIListQty* environment variable.
4. Retrieve each impacted SI's SI name and SI label. This information is contained in the *$ImpactedSLAList_<SLA_index>_SIList_<SI_index>_SILabel* environment variable, where *SI_index* is the affected SI's index in the SLA (between 1 and the value contained in the *$ImpactedSLAList_<SLA_index>_SIListQty* variable).

Note

The `dump.sh` script provides an example of a synchronous or asynchronous script. This script is contained in the following directory:
`$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts`

4.4.2 Daemon mode

Scripts executed in daemon mode are passed information as follows.

On script startup

On script startup, the script is passed general information using shell script environment variables. You can access these variables in the same way as for other shell script environment variables.

These variables are specified in the Script Configuration file's *ScriptEnv* field. They are:

- *$TEMIP_SC_HOME*
- *$TEMIP_SC_VAR_HOME*
- *$SCRIPTGTW_HOME* (the Script Gateway's release directory, in *TEMIP_SC_HOME*)
- *$SCRIPTGTW_DATA* (the Script Gateway's data directory, in *TEMIP_SC_VAR_HOME*).

The Script Gateway also passes the script the following information in the form of arguments:

- *type=<script_type>*, which specifies in which execution mode the script must run (synchronous, asynchronous, or daemon).
- The *debug* flag, which specifies whether debug mode must be enabled when the script runs.

When an SQM event is received

When the first SQM event is received, the Script Gateway application opens the daemon session and sends the following message to the daemon script's standard input:

```
<sqmscriptgtw>
```

Each time that subsequent SQM events trigger the script, their information is passed to the script's standard input in 'pseudo-XML' format.

The information is in the following form:

```
<execute debug="value_of_the_script_debug_flag">
<information_name>information_value</information_name>
<information_name>information_value</information_name>
…
</execute>
```

where each information block ends in a line feed character.

When the *file.mode* script is enabled, the *<FileName> … </FileName>* field specifies where the generated SMS file is stored.

An example of a generated SMS file is shown in appendix G, "SMS message example".

An example of the information generated when a Status Changed event occurs is shown below.

```
<execute debug=”false”>
<AcquisitionTimestamp>2004-01-27T09:21:07</AcquisitionTimestamp>
<CustomerLabel>HP customer</CustomerLabel>
<CustomerName>HPCust</CustomerName>
...
<CurrentOS>30%</CurrentOS>
<PreviousOS>100%</PreviousOS>
</execute>
```

Important

The information is not necessarily in the same order in all SQM events.

Information on affected SLAs

If the Script Configuration file specifies that structured information on affected SLAs is provided when a Threshold Crossed event occurs, this information is in the following format.

```
<StructuredImpactedSLAList>
<ImpactedSLA name=”SLA_name” label=”SLA_label” customer.name=”C
ust_name” customer.label=”Cust_label”>
<ImpactedSI name=”SI_name” label=”SI_label” />
<ImpactedSI name=”SI_name” label=”SI_label” />
…
</ImpactedSLA>
…
</ StructuredImpactedSLAList>
```

An example of the SLA information generated is shown below.

```
<StructuredImpactedSLAList>
<ImpactedSLA name=”HPGold” label=”” customer.name=”HPCust”
customer.label=”HP Customer”>
<ImpactedSI name=”VIDEO253” label=”Video Paris” />
<ImpactedSI name=”VIDEO325” label=”Video Lyon” />
</ImpactedSLA>
<ImpactedSLA name=”HPSilver” label=”” customer.name=”HPCust”
customer.label=”HP Customer”>
<ImpactedSI name=”VIDEO253” label=”Video Paris” />
</ImpactedSLA>
</StructuredImpactedSLAList>
```

Note

The `DaemonDump` script provides an example of a daemon script in which processing is performed in Java. This script is contained (in both source and compiled form) in the following directory:
`$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts directory`

To use this example script, the *script.file* property in the Script Configuration file must be similar to the following.

```
/opt/OV/SQMV12/jre/bin/java -cp /opt/OV/SQMV12/Gateways/Script/v1_2/scripts/Daem
onDump/lib/TeSCDaemonDump.jar com.compaq.temip.servicecenter.actionexecutor.scri
pt.example.daemondump.DaemonScript
```

4.5 Script debug mode

If debug mode is enabled when a script is executed in synchronous or daemon mode, Script Gateway traces the messages the script sends in its standard or error outputs.

This means that the Script Gateway trace feature must be activated (to at least the *FINE* trace level).

The gateway's trace level is stored in the SQM central repository. For details of how this is stored, see chapter 3.6.1, "The Script Gateway central repository").

4.6 Managing how script processing ends

4.6.1 Exit code

A **0** exit code is generated in the Script Gateway application when a script ends normally. All other exit codes indicate that the script exited due to an error.

In synchronous mode, a minor error is logged when an error code other than **0** is returned.

4.6.2 Generated SMS file

When the *file.mode* script is enabled, the application generates a file containing information on the original SQM event (the SMS).

In this case, the script must include an action to remove this file.

Note

Script Gateway may terminate synchronous scripts suddenly if their execution times out.

In this case, Script Gateway removes the SMS file if one exists.

4.6.3 Daemon scripts

Once a daemon script has been started, it is not stopped until the gateway is shut down (unless the configuration is reloaded as described in chapter 3.6.3, "Reloading the configuration").

Unlike in other script execution modes, therefore, the gateway must trigger the shutdown of daemon scripts.

To do so, the gateway sends the following special 'pseudo-XML' message to the daemon script's standard input:

```
</sqmscriptgtw>
```

This message closes the daemon session that started with the following message.

```
<sqmscriptgtw>
```

Appendix A

Installation directory structure

The following directories and files are installed.

Script Gateway subset

```
$TEMIP_SC_HOME
$TEMIP_SC_HOME/Gateways
$TEMIP_SC_HOME/Gateways/Script
$TEMIP_SC_HOME/Gateways/Script/v1_2
$TEMIP_SC_HOME/Gateways/Script/v1_2/jar
$TEMIP_SC_HOME/Gateways/Script/v1_2/jar/TeSCScriptGateway.jar
$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts
$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts/DaemonDump
$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts/DaemonDump/lib
$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts/DaemonDump/lib/TeSCDaemonDum
p.jar
$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts/DaemonDump/dev
$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts/DaemonDump/dev/src
$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts/DaemonDump/dev/src/com
$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts/DaemonDump/dev/src/com/compa
q
$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts/DaemonDump/dev/src/com/compa
q/temip
$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts/DaemonDump/dev/src/com/compa
q/temip/servicecenter
$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts/DaemonDump/dev/src/com/compa
q/temip/servicecenter/actionexecutor
$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts/DaemonDump/dev/src/com/compa
q/temip/servicecenter/actionexecutor/script
$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts/DaemonDump/dev/src/com/compa
q/temip/servicecenter/actionexecutor/script/example
$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts/DaemonDump/dev/src/com/compa
q/temip/servicecenter/actionexecutor/script/example/daemondump
$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts/DaemonDump/dev/src/com/compa
q/temip/servicecenter/actionexecutor/script/example/daemondump/DaemonScr
ipt.java
$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts/DaemonDump/dev/src/com/compa
q/temip/servicecenter/actionexecutor/script/example/daemondump/GtwArgume
ntParser.java
$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts/DaemonDump/dev/src/com/compa
q/temip/servicecenter/actionexecutor/script/example/daemondump/ImpactedS
LA.java
$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts/DaemonDump/dev/src/com/compa
q/temip/servicecenter/actionexecutor/script/example/daemondump/MANIFEST.
MF
$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts/DaemonDump/dev/src/com/compa
q/temip/servicecenter/actionexecutor/script/example/daemondump/ImpactedS
I.java
$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts/DaemonDump/dev/src/com/compa
```

```
q/temip/servicecenter/actionexecutor/script/example/daemondump/SQMNotifi
cationHandler.java
$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts/DaemonDump/dev/src/com/compa
q/temip/servicecenter/actionexecutor/script/example/daemondump/SimpleLog
ger.java
$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts/DaemonDump/dev/build.xml
$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts/DaemonDump/test
$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts/DaemonDump/test/run.sh
$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts/DaemonDump/test/input.xml
$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts/dump.sh
$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts/SendMail
$TEMIP_SC_HOME/Gateways/Script/v1_2/bin
$TEMIP_SC_HOME/Gateways/Script/v1_2/bin/temip_sc_gtw_setup
$TEMIP_SC_HOME/Gateways/Script/v1_2/lib
$TEMIP_SC_HOME/Gateways/Script/v1_2/lib/temip_sc_gtw_setup.pl
$TEMIP_SC_HOME/Gateways/Script/v1_2/config
$TEMIP_SC_HOME/Gateways/Script/v1_2/config/SCPlatform_SCDirector_SCAppli
cation.properties
$TEMIP_SC_HOME/Gateways/Script/v1_2/config/SCPlatform_SCDirector_SCAppli
cation_scripts.xml
$TEMIP_SC_HOME/Gateways/Script/v1_2/repository
$TEMIP_SC_HOME/Gateways/Script/v1_2/repository/ScriptGateway_setup.cfg
$TEMIP_SC_HOME/Gateways/Script/v1_2/repository/ScriptGateway_template.ex
p
$TEMIP_SC_HOME/Gateways/Script/v1_2/properties
$TEMIP_SC_HOME/Gateways/Script/v1_2/properties/TeSCScriptGateway.propert
ies
$TEMIP_SC_HOME/Gateways/Script/v1_2/properties/TeSCScriptGateway_Message
s.properties
$TEMIP_SC_HOME/Gateways/Script/v1_2/properties/TeSCScriptGateway_Version
.properties
$TEMIP_SC_HOME/DTD/scriptGtw.dtd
$TEMIP_SC_HOME/etc/addOn/ScriptGateway_v1_2_addOn_unix.tmpl_cfg
$TEMIP_SC_HOME/adapter/bin/scriptgtw_v1_2_launch.sh
$TEMIP_SC_HOME/fileset/SQMGTWSCRIPT-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.pr
operties
$TEMIP_SC_HOME/Gateways/Common/v1_2/properties/TeSCGtwCommon_Version.pro
perties
$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.prope
rties
$TEMIP_SC_HOME/ServiceAdapters/Common/v1_2/properties/TeSCSACommon_Messa
ges.properties
```

```
$TEMIP_SC_HOME/ServiceAdapters/Common/v1_2/properties/TeSCSACommon_Versi
on.properties
$TEMIP_SC_HOME/ServiceAdapters/Common/v1_2/properties/TeSCSAConfig_Messa
ges.properties
$TEMIP_SC_HOME/fileset
$TEMIP_SC_HOME/fileset/SQMSAGTWCOMMON-1.20.XX-XXXXXXXXXXX
```

Appendix B

Troubleshooting Script Gateway

See the *HP OpenView Service Quality Manager Administration Guide*.

Appendix C

Advanced installation

HP recommends that you install the Script Gateway on the SQM SLM primary host.

You can install it on another host if necessary, however.

This appendix describes how to install and configure the Script Gateway on a host other than the SQM SLM primary host.

C 1 Installing the Script Gateway

Before you install the Script Gateway, you must ensure that the OV SQM V1.2 Kernel has been installed on the host as described in appendix C 1.1, “Installing the OV SQM Kernel”, below).

You must then set the SQM environment as described in appendix C 1.2, “Required environment”, below).

Finally, you can install the Script Gateway subset as described in chapter 3.2.2.2, “Installing the Script Gateway subset”).

For a general overview of the OpenView SQM installation process, see the *HP OpenView Service Quality Manager Installation Guide*.

C 1.1 Installing the OV SQM Kernel

To install the OV SQM Kernel, do the following.

1. Mount the Service Adapters and Gateways CD-ROM on your system.
2. Go to the `SQM-1.20.xx-SAGTW` directory.
3. Log on as **root** user, and then run the `sqm_install` tool., selecting the *minimal* feature as shown in the example command below.

```
#sqm_install /opt/OV/SQMV120
/mnt/cdrom/SQM-1.20.00-SAGTW/HPUX/KIT minimal
```

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

C 1.2 Required environment

‘sqmadm’ user and group

An **sqmadm** group and user are needed to perform kernel and application setup and management tasks.

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

Environment variables

- If the OV SQM Kernel has already been set up, you have to specify the source of the `temip_sc_env.sh` file located under the SQM data directory (*$TEMIP_SC_VAR_HOME), as shown below.*

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

- If the OV SQM Kernel has not yet been set up, you must set your environment before you install the Script Gateway. This consists in doing the following:
  - Setting the *TEMIP_SC_HOME* and *TEMIP_SC_VAR_HOME* environment variables, as shown in the following examples:

```
# export TEMIP_SC_HOME=/opt/OV/SQMV120
# export TEMIP_SC_VAR_HOME=/var/opt/OV/SQM/slmv12
```

  - Sourcing the files:

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

C 1.3 Installing Script Gateway

After you have installed the SQM Kernel, you must log on as **root** user and then set the above environment variables in order to install the Script Gateway as described in chapter 3.2, “Installing the Script Gateway”.

C 2 Setting up and configuring Script Gateway

Before you can set up Script Gateway on a host other than the SQM SLM Primary Server, you must set the SQM environment variables and have a valid SQM Platform Description file.

You can then set up and configure Script Gateway by following the same procedure as that used to install Script Gateway on the SQM SLM Primary Server. For instructions on how to setup and configure Script Gateway on the SQM SLM Primary Server, see chapter 3.2, “Installing the Script Gateway”.

Note

The Script Gateway setup procedure sets up the SQM Kernel if necessary.

Environment variable

For an explanation of how to set the SQM environment, see appendix C 1.2, “Required environment”.

Platform Description file

The `platform_desc.cfg` Platform Description file must be installed at the following location:

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

If the Platform Description file is not installed there, you must retrieve it from the SQM SLM Primary Server and then copy it to this location.

Appendix D

Script Configuration XML file

Every script that can be triggered by a Script Gateway application must be declared in the Script Gateway application's Script Configuration file.

The Script Configuration file is an XML file named:
<*platform*>_<*director*>_<*application*>_script.xml

For example, the Script Configuration file for the *Script Gateway* application whose director is *gateway* running on the *slmv12* platform is named as follows:

```
Slmv12_gateway_ScriptGateway_script.xml
```

It is contained in the Script Gateway's data directory:

```
$TEMIP_SC_VAR_HOME/Gateways/Script/v1_2/config
```

Note

You can modify the Script Configuration file easily by using the `temip_sc_gtw_setup` tool described in appendix E.

Important

The Script Gateway application introduces changes made to its Script Configuration file the next time it is started or when its ReloadConfig AMI is run. For details of how to run the ReloadConfig AMI, see chapter 3.6.3, "Reloading the configuration".

D 1 Script configuration information

The Script Configuration file contains the following information for each script:

name	The action name referred to by UI Admin. This name must be unique within the Script Configuration file.
Description	A free text description of the action.
script.file	The pathname of the script that must be triggered. This path must be either relative to the `$TEMIP_SC_HOME/Gateways/Script/v1_2/scripts` directory, or an absolute path (beginning with /).
script.dir	The script's working directory. If this is not defined, the script inherits the current process's working directory (`$TEMIP_SC_HOME`).

sla.status.trigger	The pattern (a regular expression) used to trigger the Objective Status Change event script for *sla.name*. (See chapter 2.3, “Linking execution of a script to an Objective Status Change event”, for an explanation of how to link script execution to an event.)
si.status.trigger	The pattern (a regular expression) used to trigger the Objective Status Change event script for *si.name*. (See chapter 2.3, “Linking execution of a script to an Objective Status Change event”, for an explanation of how to link script execution to an event.)
admin.status	The administrative status used to prevent or allow script execution.
script.type	Defines in which mode the script is executed. (*Synchronous*|*Asynchronous*|*Daemon*)
daemon.restart	Daemon execution mode only. Defines how many times restarting of the daemon script is attempted (*0* means the script is not restarted; *-1* means that restarting of the script will be attempted indefinitely).
timeout	Synchronous execution mode only. Defines the execution timeout in seconds (*0* means that there is no timeout).
debug.mode	Debug mode is used to activate script summary traces.
file.mode	Activates SMS file generation. This feature is normally enabled only if Script Gateway has disabled the script’s *NoFileGeneration* property. For an explanation of its use, see chapter 3.6.1.2, “Advanced configuration properties”.
impacted_sla.structured	If this parameter is set to *False*, the affected SLAs information is presented as a single string. If this parameter is set to *True*, the information is presented in a structured form containing the XML tree in the case of daemon scripts or a list of environment variables for other script execution modes.
ScriptEnv	A list of ‘environment variable/value’ pairs for passing to the script.

Example

```
Name: 'Dump'
Description :
      Script that dumps HP.*SLA SQM events in file
script.file        : dump.sh
script.dir         :
admin.status       : Enabled
script.type        : Synchronous
timeout            : 0
debug.mode         : True
file.mode          : False
sla.status.trigger : HP.*SLA
si.status.trigger  : .*
impacted_sla.structured : False
ScriptEnv :
```

```
DumpFileNamePrefix = ScrGtw_dump
```

Important

Environment variables in the Script Configuration file cannot be expanded. You cannot use environment variables to specify these fields as a result.

D 2 Script Configuration file DTD

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT AvailableScripts (Script*)>
<!ELEMENT Script (Description?, ScriptEnv*)>
<!ATTLIST Script
   name                    CDATA                                    #REQUIRED
   script.file             CDATA                                    #REQUIRED
   script.dir              CDATA                                    #IMPLIED
   sla.status.trigger      CDATA                                    #IMPLIED
   si.status.trigger       CDATA                                    #IMPLIED
   admin.status            (Enabled | Disabled)                     #REQUIRED
   script.type             (Synchronous | Asynchronous | Daemon)#REQUIRED
   daemon.restart          CDATA                                    #IMPLIED
   timeout                 CDATA                                    #IMPLIED
   debug.mode              (True | False)                           "False"
   file.mode               (True | False)                           "False"
   impacted_sla.structured (True | False)                           "False"
>
<!ELEMENT ScriptEnv EMPTY>
<!ATTLIST ScriptEnv
   name                    CDATA                                    #REQUIRED
   value                   CDATA                                    #REQUIRED
>
<!ELEMENT Description (#PCDATA)>
```

Appendix E

temip_sc_gtw_setup tool

The Script Gateway is delivered with the *temip_sc_gtw_setup* tool.

This tool, which is contained in the `$TEMIP_SC_HOME/Gateways/Script/v1_2/bin` directory, simplifies Script Gateway setup and configuration tasks.

Except where another user is specified, this tool is used only by a **sqmadm** user.

Tool summary

```
temip_sc_gtw_setup [-help] [-verbose] [-quiet] [-reset]
   [-platform <platform>] [-director <director>] [-application <application>]
   -display (repo|config)  |
   -gateway (create|start|stop|delete|dump|reload) |
   -script  ((create|show|edit|delete) <script> | uia [<scripts>])
```

E 1 General options

–help

The *-help*|*-h* option displays help information on the command.

–verbose

The *-verbose*|*-v* option displays messages that are less succinct.

–quiet

The *-quiet*|*-q* option displays no messages (except for error messages).

–reset

The *-reset*|*-r* option removes the *temip_sc_gtw_setup* tool's history file.

This history file is stored at the following location:

```
$TEMIP_SC_VAR_HOME/Gateways/Script/v1_2/temip_sc_gtw_setup.xml
```

–platform

The *-platform*|*-pla* option is used to specify the name of the Script Gateway platform.

If this is not specified, the tool asks the user to enter the platform name.

The name entered by the user is stored in the tool's history file.

–director

The *-director*|*-dir* option is used to specify the name of the Script Gateway director.

If this is not specified, the tool asks the user to enter the director name.

The name entered by the user is stored in the tool's history file.

–application

The *-application*|*-app* option is used to specify the name of the Script Gateway application.

If this is not specified, the tool asks the user to enter the application name.

The name entered by the user answer is stored in the tool's history file.

E 2 *-display* option

The *-display*|*-disp* display option can be used to display the Script Gateway configuration.

This option has two possible values: *repo* and *config*.

Both these values are described in detail below.

Note

This option can be used even if the gateway is not currently running.

–display repo

The *-display repo* option is used to display the Script Gateway's central repository configuration.

An example of the output produced when you select this option is shown below.

```
> temip_sc_gtw_setup -display repo
Please enter the platform [slmv12] :
Please enter the director [gateway] :
Please enter the application [ScriptGateway] : myGtw
Reading Central repository ... Done

* Display the central repository configuration
   ActionExecutorName = ScriptGateway
   XMLValidationMode = False
   ComplianceEnabled = false
   ServiceAlarmMapping/FanOut = true
   ServiceComplianceAlarmMapping/FanOut = true
   MessageMapping = Status
   ComplianceMapping = Threshold
   QuietMode = false
   ListenerProcessorThreadNb = 1
   NoFileGeneration = true
   NoImpactedSlaGeneration = true
   ObjectiveStatusFiltering = true
   EventQueue/EventQueueSize = 100
   EventQueue/MaxThreshold = 0.9
   EventQueue/MinThreshold = 0.8
   ServiceAlarmMapping/SeverityCode/ThresholdDefaultSeverity = Major
```

```
   ServiceAlarmMapping/SeverityCode/ObjectiveStatusCriticalSeverity = 0.2
   ServiceAlarmMapping/SeverityCode/ObjectiveStatusMajorSeverity = 0.4
   ServiceAlarmMapping/SeverityCode/ObjectiveStatusMinorSeverity = 0.6
   ServiceAlarmMapping/SeverityCode/ObjectiveStatusWarningSeverity = 0.8
   ServiceComplianceAlarmMapping/SeverityCode/ThresholdDefaultSeverity = Ma
jor
   ServiceComplianceAlarmMapping/SeverityCode/ObjectiveStatusCriticalSeveri
ty = 0.2
   ServiceComplianceAlarmMapping/SeverityCode/ObjectiveStatusMajorSeverity
= 0.4
   ServiceComplianceAlarmMapping/SeverityCode/ObjectiveStatusMinorSeverity
= 0.6
   ServiceComplianceAlarmMapping/SeverityCode/ObjectiveStatusWarningSeverit
y = 0.8
```

–display config

The *-display config* option is used to display the contents of the Script Gateway's Script Configuration file in formatted form.

An example of the output produced when you select this option is shown below.

```
> temip_sc_gtw_setup -display config
Please enter the platform [slmv12] :
Please enter the director [gateway] :
Please enter the application [myGtw] :
Reading file
'/var/opt/OV/SQM/slmv12/Gateways/Script/v1_2/config/slmv12_gateway_myGtw_sc
ripts.xml' ... Done

* Display the script configuration file
 script 'SendMail'
   Description :
      Send Email on SLA status change
   script.file        : SendMail
   script.dir         :
   admin.status       : Enabled
   script.type        : Synchronous
   timeout            : 0
   debug.mode         : False
   file.mode          : False
   sla.status.trigger : .*
   si.status.trigger  :
   impacted_sla.structured : False
   ScriptEnv :
      receiver = john.doe@hp.com

 script 'Dump'
   Description :
      Script that dumps HP.*SLA SQM events in file
   script.file        : dump.sh
   script.dir         :
   admin.status       : Enabled
   script.type        : Asynchronous
   debug.mode         : True
   file.mode          : False
   sla.status.trigger : HP.*SLA
   si.status.trigger  : .*
   impacted_sla.structured : False
   ScriptEnv :
      DumpFileNamePrefix = ScrGtw_dump
```

E 3 *-gateway* option

The *-gateway*|*-gate* gateway option can be used to manage a Script Gateway application.

This option has six possible values: *create*, *start*, *stop*, *delete*, *dump*, and *reload*.

Each of these values is described in detail below.

–gateway create

The *-gateway create* option is used to create a new Script Gateway application.

Important

You must be logged on as **root** user.
No other users (including **sqmadm**) are allowed to create new SQM applications.

An example of the output produced when you select this option value is shown below.

```
>temip_sc_gtw_setup –gateway create
Please enter the platform [slmv12] :
Please enter the director [gateway] :
Please enter the application [ScriptGateway] : myGtw
* Declare the application to SQM
The platform description file is a valid XML file!
TEMIP_SC_VAR_HOME=/var/opt/OV/SQM/slmv12
Application to setup (platform, director, application):
    slmv12, gateway, myGtw
The platform description file is a valid XML file!
Start the setup of the specified addOn(s)

Traces are redirected in
/var/opt/OV/SQM/slmv12/trace/temip_sc_setup_XXXXXX_XXXXXX.log


Directors were created successfully
Traces are redirected in
/var/opt/OV/SQM/slmv12/trace/temip_sc_setup_XXXXXX_XXXXXX.log


Applications were created successfully
Operation Success.
```

–gateway delete

The *-gateway delete* option is used to delete a Script Gateway application.

Important

You must be logged on as **root** user.
No other users (including **sqmadm**) are allowed to delete SQM applications.

An example of the output produced when you select this option value is shown below.

```
>temip_sc_gtw_setup –gateway delete
Please enter the platform [slmv12] :
Please enter the director [gateway] :
Please enter the application [myGtw] :
```

–gateway start

The *-gateway start* option is used to start a Script Gateway application.

This command is entered as shown below.

```
temip_sc_start_application -platform <platform> -director <dire
ctor> -application <application>
```

An example of the output produced when you select this option value is shown below.

```
>temip_sc_gtw_setup –gateway start
Please enter the platform [slmv12] :
Please enter the director [gateway] :
Please enter the application [myGtw] :
Processing
/tibco/private/adapter/ServiceCenter/PlatformDescription/slmv12
/platform ...
Application myGtw is starting. Check Alerts in Hawk Display to
get the startup status.
launch start of application myGtw on director gateway, platform
slmv12
Operation Success.
```

–gateway stop

The *-gateway stop* option is used to shut down a new Script Gateway application.

Enter this command as shown below.

```
temip_sc_stop_application -platform <platform> -director <direc
tor> -application <application>
```

An example of the output produced when you select this option value is shown below.

```
>temip_sc_gtw_setup –gateway stop
Please enter the platform [slmv12] :
Please enter the director [gateway] :
Please enter the application [myGtw] :
* Stop the application

Processing /tibco/private/adapter/ServiceCenter/PlatformDescrip
tion/slmv12/platform ...
Application myGtw stopped
Operation Success.
```

–gateway dump

The *-gateway dump* option is used to call the Script Gateway AMI to dump the gateway configuration.

An example of the output produced when you select this option value is shown below.

```
>temip_sc_gtw_setup –gateway dump
Please enter the platform [slmv12] :
Please enter the director [gateway] :
Please enter the application [myGtw] :
* Application dump configuration
   Application: myGtw
   ha: host.vbe.cpqcorp.net-HA
   hma: slmv12_gateway_myGtw_MA

** Result of the method invocation :
composite{dump=All /var/opt/OV/SQM/slmv12/trace/0_All_slmv12_gat
eway_myGtw.dump}

Operation Success.
```

–gateway reload

The *-gateway reload* option is used to call the Script Gateway's AMI to reload the application configuration.

An example of the output produced when you select this option value is shown below.

```
>temip_sc_gtw_setup –gateway reload
Please enter the platform [slmv12] :
Please enter the director [gateway] :
Please enter the application [myGtw] :
* Application reload configuration
   Application: myGtw
   ha: host.vbe.cpqcorp.net-HA
   hma: slmv12_gateway_myGtw_MA

Operation Success.
```

E 4 *-script* option

The *-script*|*-scr* script option can be used to manage a Script Gateway's Script Configuration file.

This option has five possible values: '*create*', '*show*', '*edit*', '*delete*' and '*uia*'.

Each of these values is described in detail below.

Note

This option can be used even if the gateway is not running.

Important

The gateway introduces changes in its Script Configuration file when it is started or its ReloadConfig AMI is run (using the `temip_sc_gtw_setup -gateway reload` command, for example).

–script create

The *-script create* <*script_name*> option is used to create the specified script configuration in the gateway's Script Configuration file.

Warning

A gateway cannot have two script configurations with the same name.
If you try to give a script configuration a *script_name* that already exists in the Script Configuration file, the tool exits with an error.

When this option is selected, the *temip_sc_gtw_setup* tool asks the user to enter all of the new script configuration attributes. These are detailed in appendix D, "Script Configuration XML file".

To make it simpler for you to set the attributes, the tool proposes default values for all attributes that are not mandatory.

The task of selecting an element of an option list is further simplified by enabling you to enter only the first letter of the choice (in upper or lower case). For example, the *admin.status* attribute has only two possible values: *Enabled* or *Disabled*. To choose *Enabled*, you can enter `Enabled`, `E`, or `e`.

An example of the output produced when you select this option is shown below.

```
> temip_sc_gtw_setup –script create Dump
Please enter the platform [slmv12] :
Please enter the director [gateway] :
Please enter the application [myGtw] :
Reading file
'/var/opt/OV/SQM/slmv12/Gateways/Script/v1_2/config/slmv12_gate
way_myGtw_scripts.xml' ... Done

* Create script 'Dump' to configuration file
   script.file : dump.sh
   Description : Script that dumps HP.*SLA SQM events in file
   script.dir :
   admin.status (Enabled,Disabled) : E
   script.type (Synchronous,Asynchonous,Daemon) : S
   timeout [0] :
   daemon.restart [-1] :
   debug.mode (True,False) [False] : T
   file.mode (True,False) [False] :
   sla.status.trigger : HP.*SLA
   si.status.trigger : .*
   impacted_sla.structured (True,False) [False] :
   ScriptEnv:
   Do you want to add a new ScriptEnv ? (Yes,No) [No] : Y
     ScriptEnv name : DumpFileNamePrefix
     ScriptEnv value : ScrGtw_dump
   Do you want to add a new ScriptEnv ? (Yes,No) [No] :

>>> Summary Dump
   Description :
      Script that dumps HP.*SLA SQM events in file
```

```
   script.file        : dump.sh
   script.dir         :
   admin.status       : Enabled
   script.type        : Synchronous
   timeout            : 0
   debug.mode         : True
   file.mode          : False
   sla.status.trigger : HP.*SLA
   si.status.trigger  : .*
   impacted_sla.structured : False
   ScriptEnv :
      DumpFileNamePrefix = ScrGtw_dump
Writing file
'/var/opt/OV/SQM/slmv12/Gateways/Script/v1_2/config/slmv12_gate
way_myGtw_scripts.xml' ... Done
```

–script show

The *-script show <script_name>* option displays the specified script configuration.

An example of the output produced when you select this option is shown below.

```
> temip_sc_gtw_setup -script show Dump
Please enter the platform [slmv12] :
Please enter the director [gateway] :
Please enter the application [myGtw] :
Reading file
'/var/opt/OV/SQM/slmv12/Gateways/Script/v1_2/config/slmv12_gate
way_myGtw_scripts.xml' ... Done

* Display the script configuration

Show: script 'Dump'
   Description :
      Script that dumps HP.*SLA SQM events in file
   script.file        : dump.sh
   script.dir         :
   admin.status       : Enabled
   script.type        : Synchronous
   timeout            : 0
   debug.mode         : True
   file.mode          : False
   sla.status.trigger : HP.*SLA
   si.status.trigger  : .*
   impacted_sla.structured : False
   ScriptEnv :
      DumpFileNamePrefix = ScrGtw_dump
```

–script edit

The *-script edit <script_name>* option is used to edit an existing script configuration.

When this option is selected, the *temip_sc_gtw_setup* tool asks to the user to confirm or modify each script configuration attribute in turn. These attributes are detailed in appendix D, “Script Configuration XML file”.

Each attribute’s present value is displayed as its proposed value. To leave the value unchanged, you simply press **Enter.**

As with the *-script create* option, you can select a value by typing the first letter of the value (in upper case or lower case).

An example of the output produced when you select this option is shown below.

```
> temip_sc_gtw_setup -script edit Dump
Please enter the platform [slmv12] :
Please enter the director [gateway] :
Please enter the application [myGtw] :
Reading file
'/var/opt/OV/SQM/slmv12/Gateways/Script/v1_2/config/slmv12_gatew
ay_myGtw_scripts.xml' ... Done

* Edit script 'Dump' of the configuration file
   script.file [dump.sh] :
   Description [Script that dumps HP.*SLA SQM events in file] :
Script that dumps all SLA SQM events in file
   script.dir :
   admin.status (Enabled,Disabled) [Enabled] :
   script.type (Synchronous,Asynchonous,Daemon) [Synchronous] : A
   debug.mode (True,False) [True] : F
   file.mode (True,False) [False] :
   sla.status.trigger is 'HP.*SLA' (Keep,Change,Delete) ? [Keep]
: C
     sla.status.trigger [HP.*SLA] : .*
   si.status.trigger is '.*' (Keep,Change,Delete) ? [Keep] : D
   impacted_sla.structured (True,False) [False] :
   ScriptEnv:
     DumpFileNamePrefix = 'ScrGtw_dump' (Keep,Change,Delete)
[Keep] : C
       DumpFileNamePrefix [ScrGtw_dump] : ScrGtw_dumpV2
   Do you want to add a new ScriptEnv ? (Yes,No) [No] :

>>> Summary Dump
   Description :
      Script that dumps all SLA SQM events in file
   script.file        : dump.sh
   script.dir         :
   admin.status       : Enabled
   script.type        : Asynchronous
   debug.mode         : False
   file.mode          : False
   sla.status.trigger : .*
   si.status.trigger  :
   impacted_sla.structured : False
   ScriptEnv :
      DumpFileNamePrefix = ScrGtw_dumpV2
Writing file
'/var/opt/OV/SQM/slmv12/Gateways/Script/v1_2/config/slmv12_gatew
ay_myGtw_scripts.xml' ... Done
```

–script delete

The `-script delete` <*script_name*> option is used to remove a script configuration from the gateway's Script Configuration file.

An example of the output produced when you select this option is shown below.

```
> temip_sc_gtw_setup -script delete Dump
Please enter the platform [slmv12] :
Please enter the director [gateway] :
Please enter the application [myGtw] :
Reading file
'/var/opt/OV/SQM/slmv12/Gateways/Script/v1_2/config/slmv12_gatew
ay_myGtw_scripts.xml' ... Done
```

```
* Delete script 'Dump' to configuration file
   Configuration deleted
Writing file
'/var/opt/OV/SQM/slmv12/Gateways/Script/v1_2/config/slmv12_gatew
ay_myGtw_scripts.xml' ... Done
```

–script uia

The *-script uia [<script_name_list>]* option is used to generate an SQM SLA Admin UI (or 'UI Admin') configuration Addon file for the Script Gateway.

The UI Admin uses this data to display a list of all available ActionExecutors with their available ActionNames, from which the user can choose.

In the generated UI Admin Addon file, the ActionExecutor is the one specified in the gateway's central repository; each script name is displayed as an element in the ActionNames list.

If no list is specified, an ActionName is added for each *Enabled* script in the gateway's Script Configuration file.

If a list is specified, an ActionName is added for each specified script name (regardless of whether it is *Enabled* or not).

Note

When a list is specified, each script name in the list must exist in the gateway's Script Configuration file.

The Addon file is generated at the following location:

```
$TEMIP_SC_VAR_HOME/Gateways/Script/config/UIAdmin_<day_in_year>_
<hh>_<mm>_<ss>.xml
```

An example of the output produced when you select this option is shown below.

```
> temip_sc_gtw_setup -script uia
Please enter the platform [slmv12] :
Please enter the director [gateway] :
Please enter the application [myGtw] :
Reading file
'/var/opt/OV/SQM/slmv12/Gateways/Script/v1_2/config/slmv12_gatew
ay_myGtw_scripts.xml' ... Done

* Generate UI Admin configuration addOn file
Reading Central repository ... Done
   - Action 'SendMail' added
   - Action 'Dump' added
Writing file
'/var/opt/OV/SQM/slmv12/Gateways/Script/v1_2/config/UIAdmin_addO
n_XXX_XX_XX_XX.xml
' ... Done
```

An example of the file generated when you select this option is shown below.

```
<?xml version="1.0" encoding="UTF-8"?><!DOCTYPE
AvailableScripts>
<AvailableScripts>
  <executor executor.name="ScriptGateway"
executor.label="ScriptGateway Script Gateway">
    <Desc>Script Gateway is a shell script executor. It maps SMS
messages and activates script execution to perform 'action'
towards third party</Desc>
    <action action.name="SendMail" action.label="Send Email on
SLA status change" />
    <action action.name="Dump" action.label="Script that dumps
all SLA SQM events in file" />
  </executor>
</AvailableScripts>
```

Appendix F

Summary of regular expression constructs

Characters

Construct	Matches
X	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 new line (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 (NOT)
[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, excluding 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
.	Any character (may or may not match line terminators)
\d	A digit: [0-9]
\D	A non-digit character: [^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}]

Construct	Matches
`\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*

Construct	Matches
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 look-ahead
(?!*X*)	*X*, via zero-width negative look-ahead
(?<=*X*)	*X*, via zero-width positive look-behind
(?<!*X*)	*X*, via zero-width negative look-behind
(?>*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, while \{ matches a left brace.

It is an error to use a backslash before 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 before 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. You must therefore 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 is 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 new line (line feed) character ("`\n`"),
- A carriage return character followed immediately by a new line 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.

Appendix G

SMS message example

An example of a SMS file generated when the `file.mode` script is enabled is shown below.

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE sc:SIServiceMonitoringStatus SYSTEM "DTD/tsc_ObjectiveMonitoringEvt.dtd">

<sc:SIServiceMonitoringStatus msg.id="64" scd.name="HostSystem" scd.label="UnixPlatf
orm" sci.name="BcorpUnix1" sci.label="Bcorp Unix1" rootComponent.flag="False" xmlns:
sc="http://www.compaq.com/TeMIP/ServiceCenter">
  <sc:TriggeredActions>
    <sc:TriggeredAction executor="ScriptGateway"/>
  </sc:TriggeredActions>
  <sc:CrossedParameters>
    <sc:CrossedParameter
parameter.name="PAGE_FAULT28" parameter.label="MEM_PAGE_FAULT_RATE" customerDepend.f
lag="False" acquisition.timeStamp="2004-09-21T10:00:00.000">
      <sc:ParameterValue parameter.name="PAGE_FAULT28" datatype="Float">534.0</sc Pa
rameterValue>
      <sc:Objectives>

<sc:Objective slo.name="PAGE_FAULT24187" slo.label="MEM_PAGE_FAULT_RATE" crossingTyp
e="Up" sd.name="WAPservice" sd.label="WAP service" sl.name="WAPcustomGold" sl.label=
"WAPcustomGold" scd.name="HostSystem" csl.name="HOSTSYSTEM2516" csl.label="UnixPlatf
orm" rootComponent.flag="False" QoS.status="Increasing" eventTypeDegradation="End"
eventTypeViolation="End">
          <sc:ObjectiveThresholds>
            <sc:ObjectiveThreshold ot.name="violation" ot.label="Violation" eventTyp
e="Violation" degradationFactor="1.0" action.executor="ScriptGateway" action.name="D
ump" clearLevel.flag="False">
              <sc:ThresholdValue>1000.0</sc:ThresholdValue>
            </sc:ObjectiveThreshold>
            <sc:ObjectiveThreshold ot.name="degradation_1" ot.label="Degradation 1"
eventType="Degradation" degradationFactor="0.1" action.executor="ScriptGateway" acti
on.name="Dump" clearLevel.flag="False">
              <sc:ThresholdValue>800.0</sc:ThresholdValue>
            </sc:ObjectiveThreshold>
          </sc:ObjectiveThresholds>
          <sc:ImpactedSLASummaryList>
            <sc:ImpactedSLASummary customer.name="CustGpA" customer.label="CustGpA"
sla.name="WAPgoldCustGpAM" sla.label="WAPgoldCustGpAMonthly" sd.name="WAPservice">
              <sc:SIs>
                <sc:SI si.name="bCorporate" si.label="B Corporate" sd.name="WAPservi
ce"/>
              </sc:SIs>
            </sc:ImpactedSLASummary>
            <sc:ImpactedSLASummary customer.name="CustGpA" customer.label="CustGpA"
sla.name="WAPgoldCustGpAQ" sla.label="WAPgoldCustGpAQuarterly" sd.name="WAPservice">
              <sc:SIs>
```

```
                <sc:SI si.name="bCorporate" si.label="B Corporate"
sd.name="WAPservice"/>
              </sc:SIs>
            </sc:ImpactedSLASummary>
          </sc:ImpactedSLASummaryList>
        </sc:Objective>
      </sc:Objectives>
    </sc:CrossedParameter>
  </sc:CrossedParameters>
  <sc:SLAObjectiveStatuses>
    <sc:SLAObjectiveStatus timeStamp="2004-09-21T10:00:00.000" customer.name="CustGp
A" customer.label="CustGpA" type="Customer" sla.name="WAPgoldCustGpAM" sla.label="WA
PgoldCustGpAMonthly" sd.name="WAPservice" sd.label="WAP service" sl.name="WAPcustomG
old" sl.label="WAPcustomGold" sla.objectiveStatus="0.0" sla.previous.objectiveStatus
="0.0">
      <sc:SCIObjectiveStatuses>
        <sc:SCIObjectiveStatus sci.name="BcorpUnix1" sci.label="Bcorp Unix1" scd.nam
e="HostSystem" scd.label="UnixPlatform" sci.rootComponent.flag="False" csl.name="HOS
TSYSTEM2516" csl.label="UnixPlatform" sci.objectiveStatus="1.0" sci.previous.objecti
veStatus="0.0">
          <sc:SINameList>
            <sc:SIName si.name="bCorporate" si.label="B Corporate" sd.name="WAPservi
ce" d.label="WAP service"/>
          </sc:SINameList>
          <sc:SLOStatusSummaryList>
            <sc:SLOStatusSummary slo.name="PAGE_FAULT24187" parameter.name="PAGE_FAU
LT28" timeStamp="2004-09-21T10:00:00.000" QoS.status="Increasing" eventTypeDegradati
on="End" eventTypeViolation="End" ot.name="degradation_1" objectiveStatus="1.0"/>
          </sc:SLOStatusSummaryList>
        </sc:SCIObjectiveStatus>
      </sc:SCIObjectiveStatuses>
      <sc:AssociatedRootObjectiveStatuses>
        <sc:AssociatedRootObjectiveStatus objectiveStatus="0.0" previous.objectiveSt
atus="0.0">
          <sc:SI si.name="bCorporate" si.label="B Corporate" sd.name="WAPservice" sd
.label="WAP service"/>
        </sc:AssociatedRootObjectiveStatus>
      </sc:AssociatedRootObjectiveStatuses>
    </sc:SLAObjectiveStatus>
    <sc:SLAObjectiveStatus timeStamp="2004-09-21T10:00:00.000" customer.name="CustGp
A" customer.label="CustGpA" type="Customer" sla.name="WAPgoldCustGpAQ" sla.label="WA
PgoldCustGpAQuarterly" sd.name="WAPservice" sd.label="WAP service" sl.name="WAPcusto
mGold" sl.label="WAPcustomGold" sla.objectiveStatus="0.0" sla.previous.objectiveStat
us="0.0">
      <sc:SCIObjectiveStatuses>
        <sc:SCIObjectiveStatus sci.name="BcorpUnix1" sci.label="Bcorp Unix1" scd.nam
e="HostSystem" scd.label="UnixPlatform" sci.rootComponent.flag="False" csl.name="HOS
TSYSTEM2516" csl.label="UnixPlatform" sci.objectiveStatus="1.0" sci.previous.objecti
veStatus="0.0">
          <sc:SINameList>
            <sc:SIName si.name="bCorporate" si.label="B Corporate" sd.name="WAPservi
ce" sd.label="WAP service"/>
          </sc:SINameList>
          <sc:SLOStatusSummaryList>
            <sc:SLOStatusSummary slo.name="PAGE_FAULT24187" parameter.name="PAGE_FAU
LT28" timeStamp="2004-09-21T10:00:00.000" QoS.status="Increasing" eventTypeDegradati
on="End" eventTypeViolation="End" ot.name="degradation_1" objectiveStatus="1.0"/>
          </sc:SLOStatusSummaryList>
        </sc:SCIObjectiveStatus>
      </sc:SCIObjectiveStatuses>
      <sc:AssociatedRootObjectiveStatuses>
        <sc:AssociatedRootObjectiveStatus objectiveStatus="0.0" previous.objectiveSt
atus="0.0">
```

```
          <sc:SI si.name="bCorporate" si.label="B Corporate" sd.name="WAPservice" sd
.label="WAP service"/>
        </sc:AssociatedRootObjectiveStatus>
      </sc:AssociatedRootObjectiveStatuses>
    </sc:SLAObjectiveStatus>
  </sc:SLAObjectiveStatuses>
</sc:SIServiceMonitoringStatus>
```

Glossary

The following table lists acronyms that are commonly used in this document.

Term	Description
OVSQM	OpenView Service Quality Manager
SLA	Service Level Agreement
SLO	Service Level Objective
SI	Service Instance
SCI	Service Component Instance
SMS	Service Monitoring Status SMS message are the SQM messages indicating the objectives threshold crossings and the services status changes
Third Party Product	Any product external to SQM


invent


Java™
COMPATIBLE