hp OpenView
Service Quality Manager

Administration Guide

Edition: 1.2

for the HP-UX and Microsoft Windows Operating Systems

April 2005

Legal Notices

Warranty

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

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

Restricted Rights Legend

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

Hewlett-Packard Company

United States of America

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

Copyright Notices

Trademark Notices

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

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

Java™ is a U.S. trademark of Sun Microsystems, Inc.

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

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

NMOS™ is a trademark of RiverSoft Technologies Limited.

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

PostScript® is a trademark of Adobe Systems Incorporated.

Riversoft™ is a trademark of RiverSoft Technologies Limited.

UNIX® is a registered trademark of The Open Group.

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

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

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

Contents

List of figures **7**
Preface 8

Chapter 1 ..... **11**

Introduction **11**
1.1 OpenView SQM Architecture Overview 12
1.2 OpenView SQM Buses: TIBCO Rendezvous and Corba 12
1.2.1 TIBCO Rendezvous 12
1.2.2 Corba 13
1.3 OpenView SQM Platform 14
1.3.1 Presentation Server 14
1.3.2 Acquisition 15
1.3.3 Service Level Monitoring 15
1.3.4 Service Level Reporting 15
1.3.5 SLA Administration UI 16
1.3.6 SLA Monitoring UI 16
1.3.7 Service Level Reporting UI 16
1.3.8 Service Designer UI 16
1.4 OpenView SQM Components 16
1.5 OpenView SQM Director 17
1.6 OpenView SQM Distribution 18
1.6.1 Typical Configuration 18
1.6.2 Distributed Configuration 18
1.7 Administrator User's Environment 21
1.7.1 sqmadm user 21
1.7.2 Other SQM users 21
1.7.3 TEMIP_SC_HOME & TEMIP_SC_VAR_HOME variables 21
1.7.4 Setting the environment 22
1.7.5 Directories structure 22

Chapter 2 ..... **25**

Administrating SQM **25**
2.1 Usual SQM administration tasks 25
2.1.1 Automatic Purge Configuration 25
2.1.2 SQM Automatic start during system Boot 26
2.1.3 Starting OpenView SQM 26
2.1.4 Stopping OpenView SQM 30
2.1.5 Retrieve status of SQM platform 33
2.1.6 Monitoring the SQM platform 35
2.1.7 Operations specific to SQM components 44
2.1.8 Backup/Restore 49

2.1.9 Licensing 51
2.2 Advanced Administration 54
2.2.1 SQM directors and firewalls 55
2.2.2 SLA Monitoring and SLA Administration GUIs behind a Firewall 56
2.2.3 Switching off the multicast IP used by TIBCO Rendezvous 56
2.2.4 RVD HTTP Service 57
2.2.5 RVRD configuration 62
2.2.6 Self Management command line utility 73
2.2.7 Advanced setup procedures 75
2.2.8 Advanced utilities to modify the structure of a SQM platform 78

Chapter 3 83

Configuring SQM components 83
3.1 How to Edit Central Repository 83
3.1.1 Using the TIBCO Designer 83
3.1.2 How to edit Global Variables 87
3.1.3 How to edit SQM Components configuration 88
3.2 Global Variables 89
3.3 SQM Components specific parameters 90
3.3.1 Common properties 90
3.3.2 Service Repository Manager 91
3.3.3 Service Performance Data Manager 92
3.3.4 Data Collector 94
3.3.5 Naming Service 95
3.3.6 Service Level Objective Manager 96
3.3.7 Logger 97
3.3.8 User Interface Server 97
3.3.9 Service Adapters 99
3.3.10 Gateways 104
3.3.11 SLA Monitoring UI 104
3.3.12 SLA Administration UI 108

Chapter 4 111

Command Line Utilities 111
4.1 Create utilities 111
4.1.1 temip_sc_create_cust 111
4.1.2 temip_sc_create_expression 111
4.1.3 temip_sc_create_sd 112
4.1.4 temip_sc_create_si 112
4.1.5 temip_sc_create_sig 112
4.1.6 temip_sc_create_sla 112
4.1.7 temip_sc_create_sl 112
4.1.8 temip_sc_create_dfd 113
4.1.9 temip_sc_declare_dfi 113
4.2 Get utilities 113
4.2.1 temip_sc_get_cust 113
4.2.2 temip_sc_get_dfd 113
4.2.3 temip_sc_get_dfdexpbind 114
4.2.4 temip_sc_get_dfi 114
4.2.5 temip_sc_get_expr 115

4.2.6 temip_sc_get_sc 115
4.2.7 temip_sc_get_sci 115
4.2.8 temip_sc_get_sd 116
4.2.9 temip_sc_get_si 116
4.2.10 temip_sc_get_sig 116
4.2.11 temip_sc_get_sla 117
4.2.12 temip_sc_get_sl 117
4.2.13 temip_sc_get_model 118
4.3 State Management utilities 118
4.3.1 temip_sc_unlock_sla 118
4.3.2 temip_sc_lock_sla 118
4.3.3 temip_sc_unlock_si 118
4.3.4 temip_sc_lock_si 119
4.3.5 temip_sc_unlock_dfi 119
4.3.6 temip_sc_lock_dfi 119
4.4 Delete utilities 119
4.4.1 temip_sc_delete_cust 119
4.4.2 temip_sc_delete_dfd 120
4.4.3 temip_sc_delete_dfi 120
4.4.4 temip_sc_delete_expr 120
4.4.5 temip_sc_delete_sd 121
4.4.6 temip_sc_delete_si 121
4.4.7 temip_sc_delete_sig 121
4.4.8 temip_sc_delete_sl 121
4.4.9 temip_sc_delete_sla 122
4.5 Subscriber/Customer mapping utility 122
4.6 Basic Messaging utilities 123
4.6.1 temip_sc_listen 124
4.6.2 temip_sc_publish 126
4.6.3 temip_sc_request 127

Chapter 5 130

Troubleshooting Guide 130

5.1 SQM platform troubleshooting 130
5.1.1 Alert logging 130
5.1.2 Traces logging 132
5.2 SQM components troubleshooting 135
5.2.1 Traces and logs 135
5.2.2 Dump facility 135
5.2.3 SQM components specific troubleshooting 136
5.3 How to detect if network support multicast or broadcast IP for TIBCO Rendezvous? 140
5.4 CORBA troubleshooting 141
5.5 How to expand memory allocation of Java processes? 142

Index of Commands 143

List of figures

Figure 1: Overall architecture of OpenView SQM 12
Figure 2: TIBCO Rendezvous basic components 13
Figure 3: OpenView SQM Components 14
Figure 4: OpenView SQM typical configuration 18
Figure 5: OpenView SQM distributed configuration 19
Figure 6: Two hosts Configuration 20
Figure 7: Three hosts Configuration 20
Figure 8: TIBCO Hawk Display console 36
Figure 9: TIBCO Hawk Display: Show Alerts sent by a TIBCO Hawk agent 38
Figure 10: TIBCO Hawk Display: Result of Show Alerts on platform agent 39
Figure 11: TIBCO Hawk Display: Alert Detail Window 40
Figure 12: TIBCO Hawk Display: Get MicroAgents on platform agent 41
Figure 13: TIBCO Hawk Display: TIBCO specific MicroAgent View 41
Figure 14: TIBCO Hawk Display: SQM component MicroAgent View 42
Figure 15: SQM and firewalls 55
Figure 16: RVD HTTP port 57
Figure 17: TIBCO Rendezvous daemon information page 58
Figure 18: TIBCO Rendezvous Services page 59
Figure 19: TIBCO Rendezvous Service information page 60
Figure 20: TIBCO Rendezvous Clients page 61
Figure 21: TIBCO Rendezvous Clients detailed information page 62
Figure 22: TIBCO Rendezvous buses used by a SQM Platform 63
Figure 23: RVRD ’s HTTP page. 65
Figure 24: Router page of RVRD. 68
Figure 25: XML Configuration page of RVRD 69
Figure 26: Opening TIBCO Designer 84
Figure 27: Connecting TIBCO Designer to the Repository Server 86
Figure 28: TIBCO Designer “Global Variables” Submenu 87
Figure 29: SRM_config display using the TIBCO Designer 88
Figure 30: TIBCO Hawk Display Alert Details Window 131
Figure 31: setTraceLogLevel method on a SQM component 133
Figure 32: getTraceLogLevel method on a SQM component 134
Figure 33: getTraceLogLevel output on a SQM component 135

Preface

This document is the Administration Guide of the OpenView Service Quality Manager platform. The OpenView SQM platform is a service level management product that automates the definition of Services and Service Level Agreements (SLAs), and the monitoring and reporting of operational and customer SLAs.

1/ The first chapter of this document describes how to administer the OpenView SQM platform. It contains:

- Specific details about the architecture, which are useful to understand the OpenView SQM administration tasks.
- The usual OpenView SQM platform administration tasks:
    - Start and stop the platform.
    - Monitor the platform.
    - Configure and tune the platform components.
    - Backup the platform.
- The advanced administration tasks, which may be required to:
    - Modify the distribution of the OpenView SQM platform.
    - Configure the TIBCO Rendezvous Routing Daemon.

2/ This document also contains the reference chapter that describes the command line utilities provided by SQM for the Service Management, also called Command Line User Interface (CLUI). For instance, it describes how to retrieve the Service Definitions held by the Service Repository Manager using the command line.

3/ The last chapter of this document is a troubleshooting guide.

Intended Audience

This document is intended for the following personnel:

- Administrators who manage the SQM platform.
- Operators who want to use the Command Line Utilities to manage the Services and SLA handled by SQM. In that case, user has to read only the chapter “Services Management Command Line Utilities”. It is not required to read chapters that describe architecture and administration of SQM platform.

Prerequisite Reading

This document assumes that you have read the OpenView SQM Overview, and are familiar with OpenView SQM terminology and components.

Supported Software

The supported software referred to in this document is as follows:

Product Version	Operating System
OpenView Service Quality Manager 1.2	HP-UX 11.11 Windows XP

The term UNIX is used as a generic reference to the operating system, unless otherwise specified

Typographical Conventions

`Courier` Font:

- Source code and examples of file contents.
- Commands that you enter on the screen.
- Path names
- Keyboard key names

Italic Text:

- Filenames, programs and parameters.
- The names of other documents referenced in this manual.

Bold Text:

- To introduce new terms and to emphasize important words.

Associated Documents

The OpenView SQM documentation set includes the following:

- *OpenView SQM SLA Monitoring UI User's Guide*
- *OpenView SQM Service Designer UI User's Guide*
- *OpenView SQM SLA Administration UI User's Guide*
- *OpenView SQM Overview*
- *OpenView SQM Getting Started Guide*
- *OpenView SQM Installation Guide*
- *OpenView SQM Datamart User's Guide*
- *OpenView SQM Reporting User's Guide*
- *OpenView SQM Information Modeling Reference Guide*
- *TIBCO Hawk Administrator's Guide*
- *TIBCO Designer User's Guide*
- *TIBCO Rendezvous Administration*

Support

eCare is the HP OpenView software customer support web portal. SQM technical support information is now available to customers directly through eCare, 24x7x365.

To access the main eCare portal page, go to http://support.openview.hp.com/.

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

2. Downloadable documentation
3. Troubleshooting information
4. Patches and updates
5. Problem reporting
6. Training information
7. Support program information

Chapter 1

Introduction

OpenView SQM is built around a powerful, distributed architecture and middleware, including TIBCO Rendezvous and Oracle RDBMS. This architecture guarantees full and smooth scalability of the solution, depending on the business need.

This chapter contains a short description of OpenView SQM multi-tier architecture. It contains the following information:

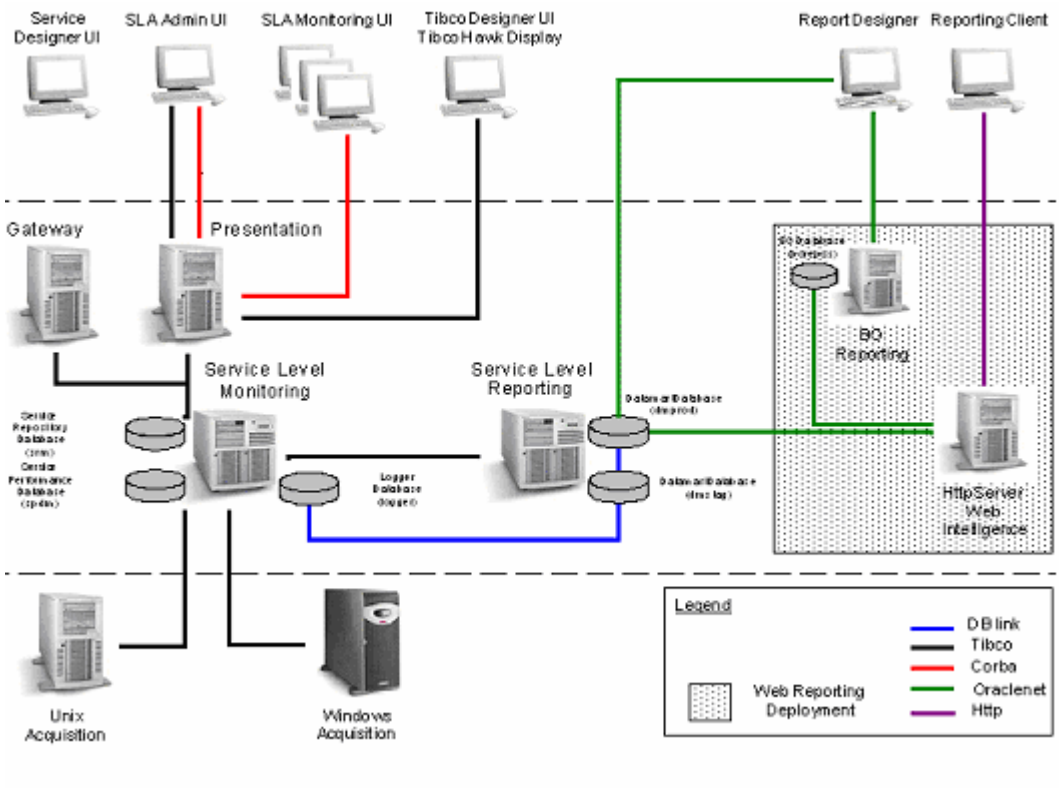
- Section 1.1 OpenView SQM Architecture Overview
- Section 1.2 OpenView SQM Buses: TIBCO Rendezvous and Corba
- Section 1.3 OpenView SQM Platform
- Section 1.4 OpenView SQM Components
- Section 1.5 OpenView SQM Director
- Section 1.6 OpenView SQM Distribution
- Section 1.7 Administrator User's Environment

1.1 OpenView SQM Architecture Overview

You should be familiar with the OpenView SQM overall architecture described in the document “OpenView SQM Planning Guide” to understand the management tasks described hereafter. This chapter adds required details about Framework, configuration and management components of the OpenView SQM platform.

The following picture recalls the overall architecture of OpenView SQM.

Figure 1: Overall architecture of OpenView SQM



1.2 OpenView SQM Buses: TIBCO Rendezvous and Corba

This section presents the two types of message busses used for communicating between the different components of OpenView SQM.

1.2.1 TIBCO Rendezvous

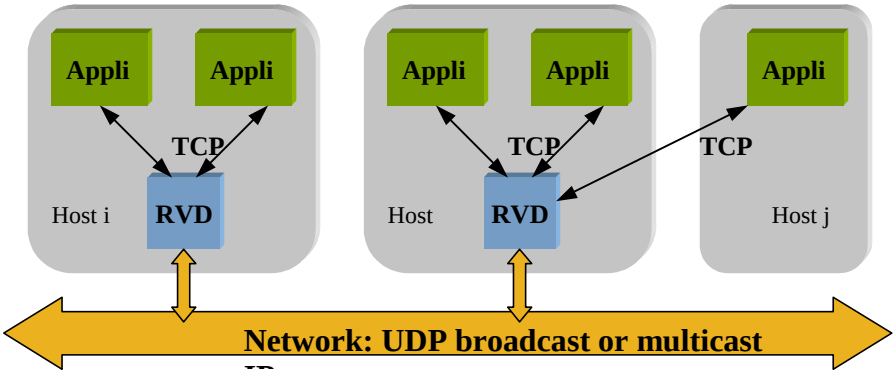
The OpenView SQM communication infrastructure is built around the TIBCO Rendezvous message bus. Refer to the document *TIBCO Rendezvous Administration* for more details about this middleware.

Several instances of the TIBCO Rendezvous message bus are present in the OpenView SQM architecture, each dedicated to a specific communication among modules.

Each module can register itself on several buses.

The TIBCO Rendezvous message bus provides full location transparency between components. Messages are conveyed to all suitable destinations as soon as the proper transport and subjects are used.

Figure 2: TIBCO Rendezvous basic components



The TIBCO Rendezvous message bus is based on:

- Rendezvous daemons (**RVD**) typically running on each host or/and
- Rendezvous routing daemon (**RVRD**) in some specific network configuration cases (see section 2.2.5.1).

RVD processes communicate which each other through UDP transports on well-identified UDP ports. There is one UDP port opened per TIBCO Rendezvous bus used by SQM.

By default, **RVD** sends broadcast UDP messages on the default IP network interface.

In addition, if IP multicast is supported by the physical network, an IP multicast group can be configured for both sending and receiving messages (IP multicast messages can traverse only IP multicast enabled routers).

Since IP multicast is not widely available, the usual approach consists of running a Rendezvous routing daemon **RVRD** on each sub-network in place of the basic **RVD** process. The routing daemon is configured to communicate with other routing daemons on other target IP sub-networks. See chapter RVRD administration.

Consequently, all functional component instances of OpenView SQM can theoretically be located on different machines with a **RVD** daemon over high-speed network (LAN type).

1.2.2 Corba

SQM uses Jacorb 2.2 as Implementation of CORBA (http://www.jacorb.org).

The SLA Monitoring UI and the SLA Administration UI communicate via CORBA with the Presentation Server, which provides a “restricted” access to OpenView SQM applications. In this version of OpenView SQM, security is simply based on UNIX security.

The SLA Monitoring UI uses CORBA to exchange data with the SQM platform.

The SLA Administration UI uses Tibco to exchange data with the SQM platform.

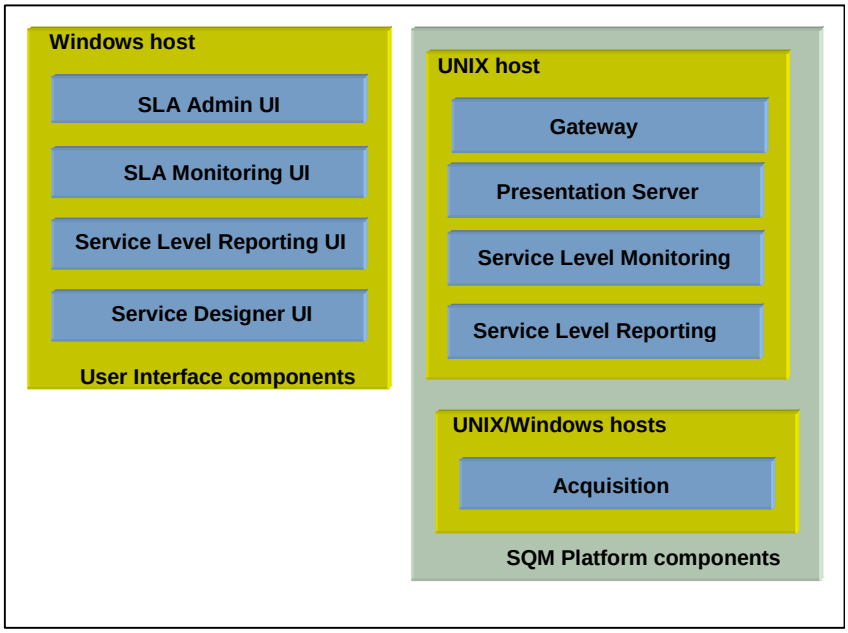
1.3 OpenView SQM Platform

An OpenView SQM platform is a set of hosts that run OpenView SQM. An OpenView SQM host always runs the SQM Kernel components. A host runs one or several directors (for information about directors see section 1.4). An OpenView SQM host requires only one instance of the SQM Kernel component.

The SQM kernel is composed of:

- TIBCO Hawk Agents
- The SQM Central Repository for the sole SQM Kernel running on the primary host
- TIBCO Rendezvous daemons (either RVD or RVRD)

Figure 3: OpenView SQM Components



OpenView SQM is composed of two types of components (Figure 3):

- Functional layers:
    - OpenView SQM Presentation Server
    - OpenView SQM Acquisition
    - OpenView SQM Service Level Monitoring
    - OpenView SQM Service Level Reporting
- User Interface components:
    - OpenView SQM SLA Administration UI
    - OpenView SQM SLA Monitoring UI
    - OpenView SQM Service Level Reporting UI
    - OpenView SQM Service Designer UI

The following sections give a short description of these layers.

1.3.1 Presentation Server

This layer acts as a gateway between Corba and TIBCO Rendezvous buses.

The SLA Monitoring UI uses the Presentation Server to connect to the Service Level Management components. The Presentation Server aggregates data from the platform. It also dispatches these data to the different SLA Monitoring UIs, allowing large numbers of users to connect simultaneously.

The Presentation Server also allows launching the SLA Monitoring UI outside the Demilitarized Zone by authorizing communication through a firewall. The protocol used between the SLA Monitoring UI and the Presentation Server is CORBA.

1.3.2 Acquisition

This layer is in charge of collecting data from the different sources of data, for example: Network, Traffic or System.

This is the southbound interface of OpenView SQM and it is composed of Service Adapters which act as connectors to various types of service information sources such as:

- End-user emulators and mobile agents (software agents embedded in end-users handsets)
- Network Information Qualifiers
- IT (service application platform)

1.3.3 Service Level Monitoring

This layer is responsible for comparing parameter values against predefined objectives. Every time a parameter value is collected, the corresponding objective defined in the scope of a SLA is validated.

The result of this validation (objective status) is stored by the Performance Data Manager.

When an objective is not met, a violation or degradation message is sent, according to the SLA type.

1.3.4 Service Level Reporting

The Service Level Reporting components archive historical service information for reporting purposes. The architecture of the Service Level Reporting components follows common rules for developing an efficient data warehouse.

The Datamart used by the Service Level Reporting components delivers aggregated views based on the following variables:

- Customer
- SLA
- Service definition
- Service instance
- Service component instance
- Time (aggregated measures, such as monthly, quarterly, and yearly)

By default, SQM is customized to use Business Object as reporting tool but other reporting tools can use the Datamart. You can also integrate the Datamart with corporate data warehouses for reporting purposes.

The Datamart consists of several databases:

- A staging database that OpenView SQM periodically populates with aggregated and summarized data from the logger database

- A production database that stores the history of all the aggregated and summarized data. This database provides information for the reporting tools.

1.3.5 SLA Administration UI

The SLA Administration UI is an application that instantiates service models, creates or modifies service levels, defines customer identities, and creates or modifies SLAs. It can work off-line when not connected to the OpenView SQM.

The SLA Administration UI first connects to OpenView SQM Presentation server to login (through CORBA). Then, it connects to OpenView SQM to update and read the Service Repository Manager database (using TIBCO Bus).

1.3.6 SLA Monitoring UI

The SLA Monitoring UI allows you to monitor Service Level Agreements and their associated services in real-time. For security, the SLA Monitoring UI uses CORBA to connect to the Service Level Management components. This connection is made through a Presentation Server that maps CORBA requests to OpenView SQM messages.

1.3.7 Service Level Reporting UI

The Service Level Reporting UI allows you to build and display reports about the quality of service delivered. You can customize the reports using the Business Objects product.

You can use the Service Level Reporting UI to:

- Consult reports
- Customize universes and reports
- Administer Business Object applications

1.3.8 Service Designer UI

The Service Designer UI allows the user to design, model, and import services in Unified Modeling Language (UML). The Service Designer UI is based on the Rational Rose Modeler.

The stand-alone Service Designer UI application runs on the Windows platform, and generates XML files. You can use the UI to design a service and check its model without interacting with the Service Level Management components.

1.4 OpenView SQM Components

The Figure 1 presents the OpenView SQM Architecture and its different components.

OpenView SQM is organized as a multi-tier architecture. The components that implement this multi-tier architecture can be organized as follows:

- Administration and Configuration
    - Message buses,
    - Process monitoring (i.e. watchdog),
    - Configuration manager (Repository Server accessible through TIBCO Designer and Administration GUI through TIBCO Hawk Display),
    - Central Logging,

    - Command Line User Interface (CLUI), a selection of dedicated command line tools for simple management of the services and instances loaded in the SQM repository.
- SQM Repository
    - This Repository stored under Oracle is managed by the Service Repository Manager (SRM).
- Data Acquisition
    - Data Acquisition is managed by one or several Service Adapters that are aimed at collecting data from different sources of data.
- Service Level Management
    - Performance Data Collector (DC) is aimed at collecting all measures from Service Adapters to map them in SQM Service Components measures,
    - Subscriber Naming Service (NS) is aimed at mapping subscribers into customers,
    - Service Performance Data Manager (SPDM) is aimed at calculating and aggregating measures,
    - SLO Monitoring (SLOM) is aimed at validating measures against objectives and generates violation or degradation events if needed,
    - Service Data Logger (Logger) is aimed at storing all SQM events for future processing by the Datamart.
- Service Level Reporting
    - Datamart,
    - Service Level Reporting,
    - Service Health SA.
- Interface functions
    - Service Designer UI,
    - Service Level Agreement Administration UI,
    - Service Level Agreement Monitoring UI,
    - User Interface Presentation Server,
    - Service Level Reporting UI.

1.5 OpenView SQM Director

Directors in OpenView SQM are autonomous systems that manage a set of OpenView SQM functionalities. A director is hosted on a single host.

The OpenView SQM packaging allows you to install the functional components independently on one or several machines.

In version 1.2 of the OpenView SQM product, a typical installation contains the following types of directors:

- Service Level Monitoring
- Acquisition (Service Adapters)
- Presentation Server
- Service Level Reporting

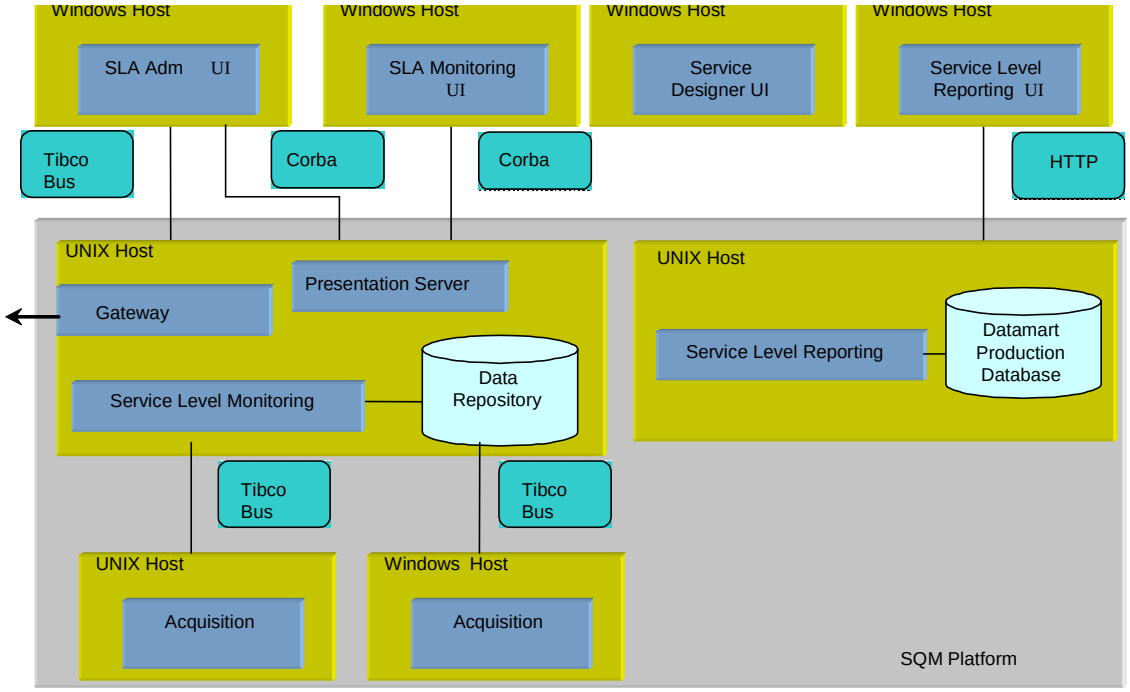
You can have several hosts for each type of director. For best performances, the Service Level Reporting director should run on a different host.

1.6 OpenView SQM Distribution

This chapter is intended to show briefly how the different OpenView SQM components can be organized in an OpenView SQM platform.

1.6.1 Typical Configuration

Figure 4: OpenView SQM typical configuration



In a typical configuration, all the core components are located on the same host. You can distribute certain components such as the Service Level Reporting director on another host for performance reasons (CPU load for example). Several instances of these components can be distributed on several hosts. Not all the components can be distributed. The following section describes possible distributed configurations.

1.6.2 Distributed Configuration

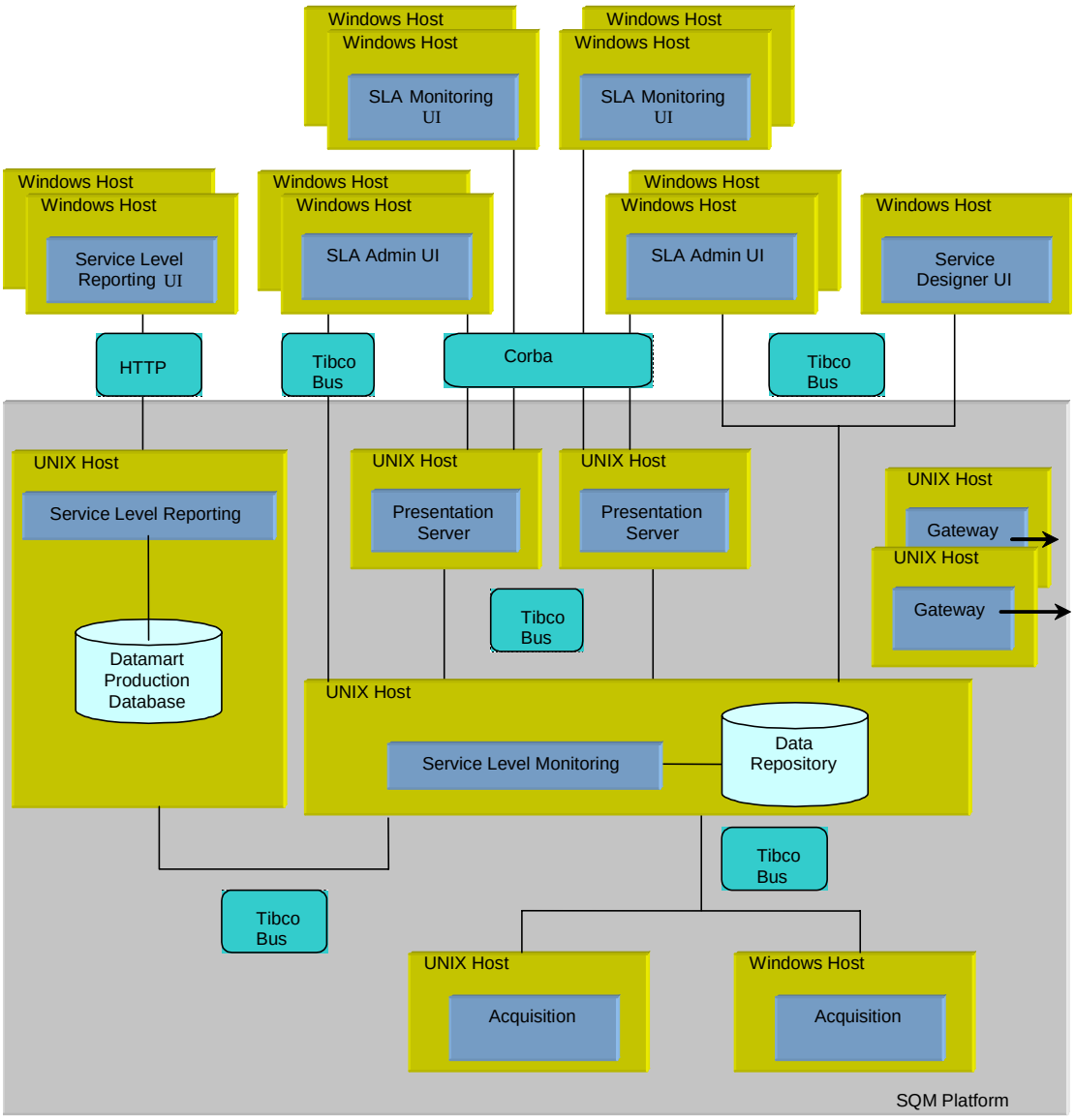
Depending on the complexity of your solution, you may need a large number of operators. For example, you might assign each operator to monitor a specific customer, instance group, or SLA.

As OpenView SQM architecture is distributed, the platform can be distributed on several machines, and several different directors can be hosted on the same machine.

In the version V1.2 of the OpenView SQM product, the following restriction applies: there can be only one Service Level Monitoring director on the platform.

OpenView SQM allows a large number of SLA Monitoring UIs to connect simultaneously. You can duplicate the Presentation Server as many times as you need. Figure 5 illustrates a configuration that supports multiple SLA Monitoring UIs running on multiple Presentation Servers.

Figure 5: OpenView SQM distributed configuration



For example, Figure 6 describes a configuration of two hosts with two directors each. Host1 contains the Service Level Monitoring director and Presentation Server director and host2 contains an Acquisition director: the OVSN SA and a Gateway director. Each host also includes the OpenView SQM Kernel.

Figure 6: Two hosts Configuration

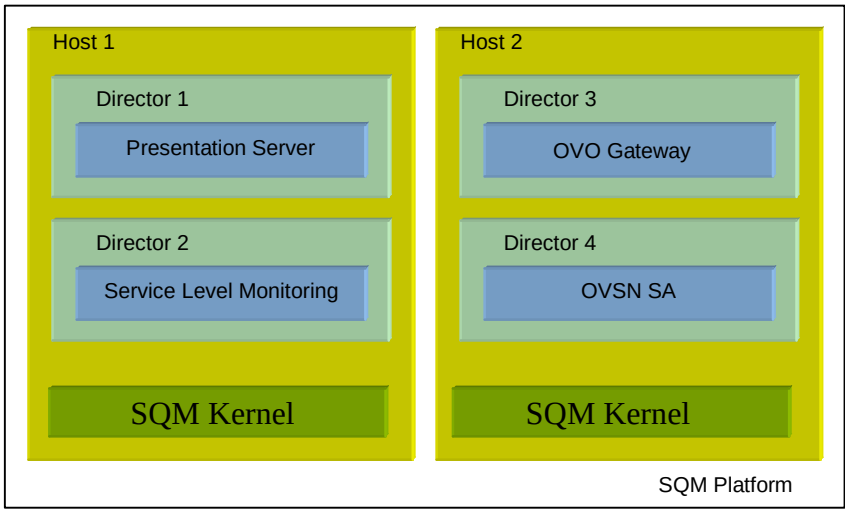
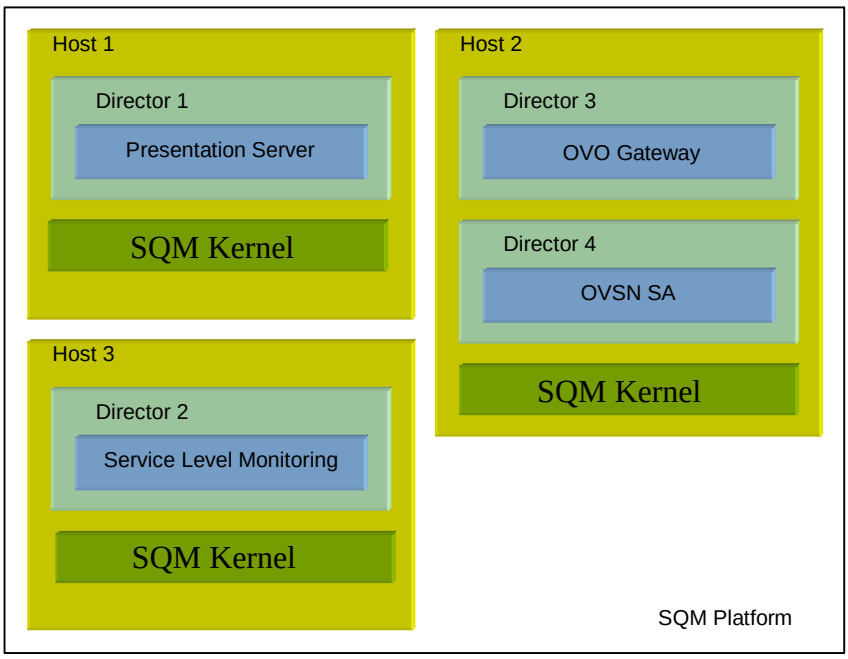


Figure 7: Three hosts Configuration



1.7 Administrator User's Environment

1.7.1 sqmadm user

On UNIX, all administrative commands described in this document have to be executed under the **sqmadm** user.

Please refer to the *OpenView SQM Installation Guide* to get more information on how to create this user.

1.7.2 Other SQM users

SQM users should note the following:

- Oracle administration must be done under the Oracle User.
- The default configuration of SQM specifies that:
  - To be able to log in the SLA Monitoring UI, the users should be declared as members of one of the following UNIX groups: users or sqmadm.
    You can define the list of Unix groups authorized to use the SLA Monitoring modifying the Presentation Server property "AuthorizedGroupSlaMonitoring" in the TIBCO Designer (node screpos/ServiceCenter/USerInterface/UIServer/V1_0/UIS_config).
    See sections 3.1 and 3.3.8 for details.
  - To be able to log in SLA Administration UI, the users should be declared as members of one of the following UNIX groups: users or sqmadm.
    You can define the list of Unix groups authorized to use the SLA Administration modifying the Presentation Server property "AuthorizedGroupSlaAdministration" in the TIBCO Designer (node screpos/ServiceCenter/USerInterface/UIServer/V1_0/UIS_config)

1.7.3 TEMIP_SC_HOME & TEMIP_SC_VAR_HOME variables

When your platform is installed, two main environment variables are set during the SQM setup:

- TEMIP_SC_HOME
- TEMIP_SC_VAR_HOME

TEMIP_SC_HOME points to the directory containing the installed components of the OpenView SQM software.

TEMIP_SC_VAR_HOME points to the directory containing data that varies during the life of your platform (logs, traces, etc).

By default, the value of these variables is:

On UNIX

```
TEMIP_SC_HOME: /opt/OV/SQM
TEMIP_SC_VAR_HOME: /var/opt/OV/SQM/slmv12
```

On Windows Acquisition Directors

```
TEMIP_SC_HOME: C:\Program Files\HP OpenView\SQM
TEMIP_SC_VAR_HOME: C:\Program Files\HP OpenView\SQM\slmv12
```

On Windows workstations where SLA Monitoring and SLA Administration kits are installed

```
TEMIP_SC_HOME: C:\Program Files\HP OpenView\SQM
```

Please refer to the *OpenView SQM Installation Guide* to get more information about the setting of these variables.

On Windows Web server or UNIX Web Server used to deploy the SLA Monitoring using Java Web Start technology

No environment variable is set by the installer when the SLA Monitoring is installed on the Web Server.

On Windows workstations where the SLA Monitoring has been deployed using Java Web Start technology

When the SLA Monitoring is deployed using Java Web Start, no environment variable is set on the Windows workstation.

1.7.4 Setting the environment

On UNIX

On UNIX, the script file that sets all the environment variables used by the SQM platform operations is:

$TEMIP_SC_VAR_HOME/temip_sc_env.sh

This script must have been executed in your environment before using any other SQM command line script, using the following command:

. temip_sc_env.sh

Note

It is recommended to add this script in the profile file of the sqmadm user.

On Windows Acquisition Directors

On Windows, the script file that sets all the environment variables used by the SQM platform operations is:

%TEMIP_SC_VAR_HOME%\temip_sc_env.bat

This bat file must have been executed before using any SQM management commands, using the following command:

call %TEMIP_SC_VAR_HOME%\temip_sc_env.bat

1.7.5 Directories structure

The main directories under **TEMIP_SC_HOME** are:

- DTD: contains the DTD of the XML messages exchanged between modules
- Definitions: contains the definitions provided with SQM:
  - Definitions/Services: example of services (Video Model for example)
  - Definitions/DFD: examples of DFD (Service Health for example)
  - Definitions/Expressions: contains the predefined expressions
- SLM: software that constitutes the Service Level Monitoring components
- ServiceAdapters: Service Adapter components software
- Gateways: Gateway components software
- UI: User Interface components software
- adapter: contains the scripts to launch the different modules
- bin: all tools (scripts)

- etc: contains shell scripts and configuration files
- fileset: contains the list of all SQM subsets installed
- java: Java software
- patches: patch directory
- setup: scripts used to setup a SQM platform (e.g. temip_sc_setup)

The main directories under **TEMIP_SC_VAR_HOME** are:

- Definitions: contains your definitions
- Gateways: Gateways components variable data
- SLM: variable data depending on the platform for the Service Level Monitoring components: property files, etc (similarly for other components)
- ServiceAdapters: Service Adapters components variable data
- UI: User Interface components variable data
- config: configuration files
- definition: contains your definitions
- hrb: contains the Hawk rulebases
- ledger: all ledger files
- log: logs (errors, warnings, …)
- oracle: Oracle database
- pid: list of running components
- dat: Central Repository file screpos.dat
- setupconfig: contains all elements related to setup
- trace: traces directory

Chapter 2

Administrating SQM

2.1 Usual SQM administration tasks

2.1.1 Automatic Purge Configuration

The service performance data manager and the logger databases must be purged on a regular basis. The following is an example of how to setup regular database purge in a `crontab`. See your system administrator for more information.

The purge data can be proposed to the user, to run in a `crontab`.

Here is the script that performs the purge:

```
#temip_sc_purge_data.sh
#!/bin/sh

. /var/opt/OV/SQM/slmv12/temip_sc_env.sh
export PLATFORM=slmv12
export DIRECTOR=slmonitoring
export SPDM_APPLI=SPDM
export LOGGER_APPLI=Logger

date >>  /tmp/temip_sc_purge_data.log

temip_sc_selfmgmt -u \
               $TEMIP_SC_VAR_HOME/config/CLUItibConfig \
               -ha `hostname`-HA -hma \
             ${PLATFORM}_${DIRECTOR}_${SPDM_APPLI}_MA \
             -m purgeDbMethod \
     -a "Purge Type"=Data  >>  /tmp/temip_sc_purge_data.log
echo >> /tmp/temip_sc_purge_data.log

temip_sc_selfmgmt -u \
               $TEMIP_SC_VAR_HOME/config/CLUItibConfig \
               -ha `hostname`-HA -hma \
             ${PLATFORM}_${DIRECTOR}_${LOGGER_APPLI}_MA \
             -m purge \
           -a Mode=Delete Days=0 datamart=True \
                        >> /tmp/temip_sc_purge_data.log
echo >> /tmp/temip_sc_purge_data.log
```

You can setup (as the root user) the following crontab, which will purge every day at midnight:

```
cp temip_sc_purge_data.sh /sbin

# Edit the sqmadm crontab:
crontab -e sqmadm

# and put
0 0 * * * /sbin/temip_sc_purge_data.sh
```

Warning

Data will be purged by comparison with GMT time. The start time of your crontab should take this information into account.

2.1.2 SQM Automatic start during system Boot

After the end of SQM setup, you may choose to install the boot scripts that will start the SQM platform and the required third products during the system boot. You have to invoke, under the root user id, the following commands:

Installing the OpenView SQM startup at UNIX system boot

On HP-UX, if you want to install the SQM startup as a system boot script, you have to invoke the following command as root:

```
$TEMIP_SC_HOME/setup/bin/temip_sc_install_SQMboot
    $TEMIP_SC_HOME $TEMIP_SC_VAR_HOME <platformName>
```

For instance, if you want the kernel of the slmv12 SQM platform to be started automatically at boot time, use the following command:

```
/opt/OV/SQM/setup/bin/temip_sc_install_SQMboot
       /opt/OV/SQM /var/opt/OV/SQM/slmv12 slmv12
```

This script will also install the ORBIX daemons startup and the Oracle database startup if required by the SQM startup at boot. See details in document “OpenView SQM Oracle Reference Guide (appendix D)”.

Installing the OpenView SQM startup at WINDOWS system boot

On Windows, if you want to install the SQM Kernel startup during the WINDOWS system boot script, you have to invoke the following command:

```
%TEMIP_SC_HOME%\setup\bin\temip_sc_install_SQMboot.bat
```

The SQM Directors and their applications will be started by the SQM Platform start executed on the SQM Primary host.

2.1.3 Starting OpenView SQM

On UNIX, the *temip_sc_env.sh* script has to be executed on each “session” that will run administrative command on the SQM platform in order to set the environment variables. On Windows, the environment variables are set by the temip_sc_env.bat script.

Then you need to start the SQM kernel.

Note that the SQM kernel must be restarted each time you reboot the host. However, this restart can be done automatically during the system boot (see section 2.1.2).

Starting manually OpenView SQM and required third party products

If you have not installed the SQM start-up at the UNIX boot process, you may have to enter the following commands in order to start correctly SQM on the primary host.

```
# Starting Oracle for SRM SPDM & Logger
# and eventually Datamart :
su – oracle
$ORACLE_HOME/bin/lsnrctl start
export ORA_TZFILE=$ORACLE_HOME/oracore/zoneinfo/timezlrg.dat
dbstart

# Starting SQM:
su - sqmadm
export TEMIP_SC_HOME=          # default install is
                               # /opt/OV/SQM
export TEMIP_SC_VAR_HOME=      # default install is
                               # /var/opt/OV/SQM/slmv12
. $TEMIP_SC_VAR_HOME/temip_sc_env.sh
temip_sc_kernel_start
temip_sc_start_platform -platform <platformName>
      # For instance<platformName> is slmv12 by default
```

2.1.3.1 Starting the OpenView SQM Kernel

An OpenView SQM Kernel is started on a given host, from its command line using the **temip_sc_kernel_start** command under the user id sqmadm.

An OpenView SQM Kernel must be started on each host of the distributed SQM platform to allow all OpenView SQM components to start and communicate in the same environment.

Command line syntax

```
temip_sc_kernel_start
```

Note

If you are in a distributed environment, you need to start the SQM kernel on each host.

2.1.3.2 Starting an OpenView SQM platform

An OpenView SQM platform is started from the command line using the **temip_sc_start_platform** command under the user id sqmadm.

This command starts all the directors and OpenView applications declared in the specified platform.

This command only works if the OpenView SQM kernel has been started on all the hosts where the OpenView SQM applications are deployed.

Command line syntax

```
temip_sc_start_platform –platform <platform>

Example: temip_sc_start_platform –platform slmv12
```

Parameters description

-platform <platform>	:	the platform to start
-help	:	displays the command line syntax

2.1.3.3 Starting an OpenView SQM director

An OpenView SQM director is started from the command line using the **temip_sc_start_director** command under the user id sqmadm.

This command starts all the OpenView applications declared in the specified director.

Command line syntax

```
temip_sc_start_director -platform <platform>
                        -director <director>

Example: temip_sc_start_director –platform slmv12 –director
slmonitoring
```

Parameters description

-platform <platform>	:	the platform to which the director belongs
-director <director>	:	the director to start
-help	:	displays the command line syntax

2.1.3.4 Starting an OpenView SQM application

An application is started from the command line by invoking the **temip_sc_start_application** command under the user id sqmadm.

Note

We recommend you to use t*emip_sc_start_platform* or possibly *temip_sc_start_director* rather than *temip_sc_start_application* (which is reserved for an advanced use).

Command line syntax

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

Parameters description

-platform <platform>	:	the platform to which the director belongs
-director <director>	:	the director of the application
-application < application >	:	the application to start
-help	:	displays the command line syntax

Example: `temip_sc_start_application –platform slmv12 –director slmonitoring -application SRM`

How to start SRM

On a typical SQM platform named smlv12, the Service Repository Manager runs on the *slmonitoring* director. To start the SRM, invoke the following command:

```
temip_sc_start_application –platform slmv12 –director
slmonitoring –application SRM
```

How to start SPDM

On a typical SQM platform named smlv12, the Service Performance Data Manager runs on the *slmonitoring* director. To start the SPDM, invoke the following command:

```
temip_sc_start_application –platform slmv12 –director
slmonitoring –application SPDM
```

How to start DC

On a typical SQM platform named smlv12, the Data Collector runs on the *slmonitoring* director. To start the DC, invoke the following command:

```
temip_sc_start_application –platform slmv12 –director
slmonitoring –application DC
```

How to start NS

On a typical SQM platform named smlv12, the Naming Service runs on the *slmonitoring* director. To start the NS, invoke the following command:

```
temip_sc_start_application –platform slmv12 –director
slmonitoring –application NS
```

How to start SLOM

On a typical SQM platform named smlv12, the Service Level Objective Manager runs on the *slmonitoring* director. To start the SLOM, invoke the following command:

```
temip_sc_start_application –platform slmv12 –director
slmonitoring –application SLOM
```

How to start Logger

On a typical SQM platform named smlv12, the Logger runs on the *slmonitoring* director. To start the Logger, invoke the following command:

```
temip_sc_start_application –platform slmv12 –director
slmonitoring –application Logger
```

How to start UI Server (Presentation server)

On a typical SQM platform named smlv12, the User Interface Server runs on the *presentation* director. To start the UIS, invoke the following command:

```
temip_sc_start_application –platform slmv12 –director
presentation –application UIS
```

How to start a SA

Directors for Service Adapters:

Generally, a Service Adapter application is created under the acquisition director. However, several Service Adapters can be installed on different hosts, in that case there should be several acquisition directors (created with different names) for each system boxes that will host a Service Adapter.

For instance: an OVIS Service Adapter is to be installed on PC1 and the OvSN Service Adapter is to be installed on HP2. In this case, two acquisition directors are created on the same SQM platform:

acquisition-PC1

acquisition-HP2

Each Service Adapter comes with its setup tool that creates the Service Adapter director and application. The setup prompts the user for both director name and application name to be created on the SQM Platform. Please refer to the appropriate Service Adapter Installation and Configuration Guide for more information about application creation.

On a typical SQM platform named smlv12, to start a SA named <*SAname*> belonging to a director `<acquisitionDirectorName>`, invoke the following command:

```
temip_sc_start_application –platform slmv12 -director
<acquisitionDirectorName> –application <SAname>
```

Depending on the Service Adapter, the application start command may be encapsulated in a menu provided at the Service Adapter installation. Please consult the Service Adapter documentation.

2.1.4 Stopping OpenView SQM

Stopping an OpenView SQM platform

An OpenView SQM platform is stopped from the command line using the **temip_sc_stop_platform** command under the user id sqmadm.

This command stops all of the SQM directors and the SQM applications declared under the specified SQM platform.

This only works if the OpenView SQM kernel has been started on all the hosts where the OpenView SQM applications are configured to run.

Command line syntax

```
temip_sc_stop_platform –platform <platform> [-force]

Example: temip_sc_stop_platform –platform slmv12
```

Parameters description

-platform <platform>	:	the platform to stop
-force	:	optional parameter to stop in force mode. With this mode the SQM applications interrupt their current tasks without to wait the end of the processing.
-help	:	displays the command line syntax

Stopping an OpenView SQM Director

A Director is stopped from the command line using the **temip_sc_stop_director** command under the user id sqmadm.
This command stops all the SQM applications declared in the specified Director.

Command line syntax

```
temip_sc_stop_director -platform <platform> -director
<director>

Example: temip_sc_stop_director –platform slmv12 –director
slmonitoring
```

Parameters description

-platform <platform>	:	the platform to which the director belongs
-director <director>	:	the director to be stopped
-force	:	Optional parameter to stop in force mode. With this mode the SQM applications interrupt their current tasks without to wait the end of the processing.
-help	:	displays the command line syntax

Stopping an OpenView SQM application

An SQM application is stopped from the command line using the **temip_sc_stop_application** command under the user id sqmadm.

Note

We recommend you to use of t*emip_sc_stop_platform* or possibly *temip_sc_stop_director* rather than *temip_sc_stop_application*, which is reserved for an advanced use.

Command line syntax

```
temip_sc_stop_application -platform <platform> -director
<director> -application <application> [-force]
```

Parameters description

-platform <platform> : the platform to which the director belongs

-director <director> : the director to which the application belongs

-application < application >: the application to be stopped

-force : optional parameter to stop in force mode. In this mode, the SQM application interupts its current running tasks without waiting for the end of processing.

-help : displays the command line syntax

```
Example:
temip_sc_stop_application –platform slmv12
–director slmonitoring –application SRM
```

How to stop SRM

On a typical SQM platform named smlv12, the Service Repository Manager runs on the *slmonitoring* director. To stop the SRM, invoke the following command:

```
temip_sc_stop_application –platform slmv12 -director
slmonitoring –application SRM
```

How to stop SPDM

On a typical SQM platform named smlv12, the Service Performance Data Manager runs on the *slmonitoring* director. To stop the SPDM, invoke the following command:

```
temip_sc_stop_application –platform slmv12 –director
slmonitoring –application SPDM
```

How to stop DC

On a typical SQM platform named smlv12, the Data Collector runs on the *slmonitoring* director. To stop the DC, invoke the following command:

```
temip_sc_stop_application –platform slmv12 –director
slmonitoring –application DC
```

How to stop NS

On a typical SQM platform named smlv12, the Naming Service runs on the *slmonitoring* director. To stop the NS, invoke the following command:

```
temip_sc_stop_application –platform slmv12 –director
slmonitoring –application NS
```

How to stop SLOM

On a typical SQM platform named smlv12, the Service Level Objective Manager runs on the *slmonitoring* director. To stop the SLOM, invoke the following command:

```
temip_sc_stop_application –platform slmv12 –director
slmonitoring –application SLOM
```

How to stop Logger

On a typical SQM platform named smlv12, the Logger runs on the *slmonitoring* director. To stop the Logger, invoke the following command:

```
temip_sc_stop_application –platform slmv12 –director
slmonitoring –application Logger
```

How to stop UI Server

On a typical SQM platform named smlv12, the User Interface Server runs on the *presentation* director. To stop the UIS, invoke the following command:

```
temip_sc_stop_application –platform slmv12 –director
presentation –application UIS
```

How to stop SA

Directors for Service Adapters:

Generally, a Service Adapter application is created under the acquisition director. However, several Service Adapters can be installed on different hosts, in that case there should be several acquisition directors (created with different names) for each system boxes that will host a Service Adapter.

For instance: an OVIS Service Adapter is to be installed on PC1 and the OvSN Service Adapter is to be installed on HP2. In this case, 2 acquisition directors are created on the same SQM platform:

acquisition-PC1

acquisition-HP2

On a typical SQM platform named smlv12, to stop a SA named <*SAname*> belonging to a director `<acquisitionDirectorName>`, invoke the following command:

```
temip_sc_stop_application –platform slmv12 -director
<acquisitionDirectorName> –application <SAname>
```

Stopping the OpenView SQM Kernel

An OpenView SQM Kernel is stopped on a given host, from its command line using the **temip_sc_kernel_stop** command under the user id sqmadm.

Command line syntax

```
temip_sc_kernel_stop [-killjacorb] [-killrvd]
```

Without option or with any of the options, the command stops the Hawk daemons and the Repository if any.

In addition, with the option –killjacorb, the Jacorb name server is also killed.

With the option –killrvd, the rvd daemon is killed along with all of the other kernel daemons.

Notes

- If you are in a distributed environment, you need to stop the SQM kernel on each host.

- The rvd daemon will be restarted as long as one of the SQM applications is running, even if you stop the kernel with option –killrvd. All of the SQM applications running on the local host must be stopped before to stop the kernel to actually stop the rvd daemon.

2.1.5 Retrieve status of SQM platform

Show the SQM platform

The **temip_sc_show_platform** command retrieves the list of all directors declared in the SQM platform and their applications along with the hostname where they have been configured to run on.

This command also displays the status (RUNNING or not) of the SQM platform, i.e. the status of all of its SQM directors and SQM applications.

If your platform is distributed on several hosts, this command will display all the applications. You can invoke the command from any host where the SQM Kernel is currently running.

Command line syntax

`temip_sc_show_platform –platform <platform> [-verbose]`

`Example: temip_sc_show_platform –platform slmv12`

Parameters description

-platform <platform>	:	name of the platform to be displayed. By default, the setup of SQM creates an SQM platform named *slmv12*.
-verbose	:	In verbose mode, gives full information about the platform definition.
-help	:	displays the command line syntax

Output without -verbose

```
Processing
  /tibco/private/adapter/ServiceCenter/PlatformDescription/slmv12/platform ...
Platform slmv12 :

Director slmonitoring :

Application SRM :
        applicationType  = Monitored
        host             = henry.vbe.cpqcorp.net
Application SRM is RUNNING

Application SPDM :
        applicationType  = Monitored
        host             = henry.vbe.cpqcorp.net
Application SPDM is RUNNING

#  Etc …
```

Output in verbose mode

```
Processing
  /tibco/private/adapter/ServiceCenter/PlatformDescription/slmv12/platform ...
Platform slmv12 :

        MonitoringRulePeriod     = 1
        MonitoringRuleRetryNb    = 3
```

```
Director slmonitoring :

        startid          = 10
        stopid           = 30

Application SRM :
        applicationType  = Monitored
        host             = henry.vbe.cpqcorp.net
        command          = srm_launch.sh
        startid          = 10
        stopid           = 60
        start_duration   = 1000
        stop_duration    = 50
        configpath       = /tibco/private/adapter/ServiceCenter/SLM/SRM/v1_0/SRM
Application SRM is RUNNING

Application SPDM :
        applicationType  = Monitored
        host             = henry.vbe.cpqcorp.net
        command          = spdm_launch.sh
        startid          = 20
        stopid           = 50
        start_duration   = 1000
        stop_duration    = 50
        configpath       =
                                /tibco/private/adapter/ServiceCenter/SLM/SPDM/v
                                1_0/SPDM
Application SPDM is RUNNING

#  Etc …
```

Show an SQM director

The status of an OpenView SQM director is shown using the **temip_sc_show_director** command. It shows the status of all the SQM applications declared in the given SQM director.

You can invoke the command from any host where the SQM Kernel is currently running.

Command line syntax

```
temip_sc_show_director –platform <platform>
                       -director <director>
```

Parameters description

-platform <platform>	:	the platform to which the director belongs
-director <director>	:	the director to which the application belongs
-verbose	:	in verbose mode, gives full information about the director definition.
-help	:	displays the command line syntax

The outputs of the **temip_sc_show_director**, in mode verbose or not verbose, are the same as the outputs of the temip_sc_show_platform but with the only information relative to the SQM director.

Show an SQM application

The status of an SQM application is shown using the t**emip_sc_show_application** command.

You can invoke the command from any host where the SQM Kernel is currently running.

The outputs of the **temip_sc_show_director** in mode verbose or not verbose are the same as the outputs of the temip_sc_show_platform but with the only information relative to the SQM application.

Command line syntax

```
temip_sc_show_application –platform <platform>
            -director <director> -application <application>
```

Parameters description

-platform <*platform*> : the platform to which the director belongs

-director <*director*> : the director to which the application belongs

-application < *application* >: the application to display

-verbose : In verbose mode, gives full information about the director definition.

-help : displays the command line syntax

In addition to this command, there is another command that only shows the list of all SQM applications currently running on the local host. This command is **temip_sc_show**, it can only be invoked on the local host**.** See section 2.1.6.3 Monitoring a host through the command line for more details.

2.1.6 Monitoring the SQM platform

2.1.6.1 Monitoring OpenView SQM using the Admin Console

Introduction

This section explains how the OpenView SQM platform can be managed with the TIBCO Hawk Display console, and in another way, directly from the command line. The most useful pieces of information managed are the platform and its component status and the alerts dispatched by applications that are running on the platform.

TIBCO Hawk sub-system of the SQM Kernel

The TIBCO Hawk Software is embedded as part of the OpenView SQM. It is used as a sub-system of the SQM Kernel to monitor and manage the distributed SQM Components.

For more details about TIBCO Hawk, see the “TIBCO Hawk *Administrator’s Guide”* document.

The TIBCO Hawk sub-system of the SQM Kernel is composed of TIBCO Hawk Agents running on each system boxes that hosts a part of the distributed SQM platform. A TIBCO Hawk Agent on a given host is responsible for:

- Discovering the SQM Components activated on this host,
- Invoking the management operations requests on the Hawk Micro Agent owned by the specified SQM Components running on this host.
- Collecting the Alerts issued by the SQM Components running on this host. These Alerts are then forwarded to the Central Hawk Service and to the currently opened TIBCO Hawk Displays.

To monitor a distributed application on the network using TIBCO Hawk Software, a TIBCO Hawk Agent must be running on each system box that host this distributed.

Graphical OpenView SQM Admin Console

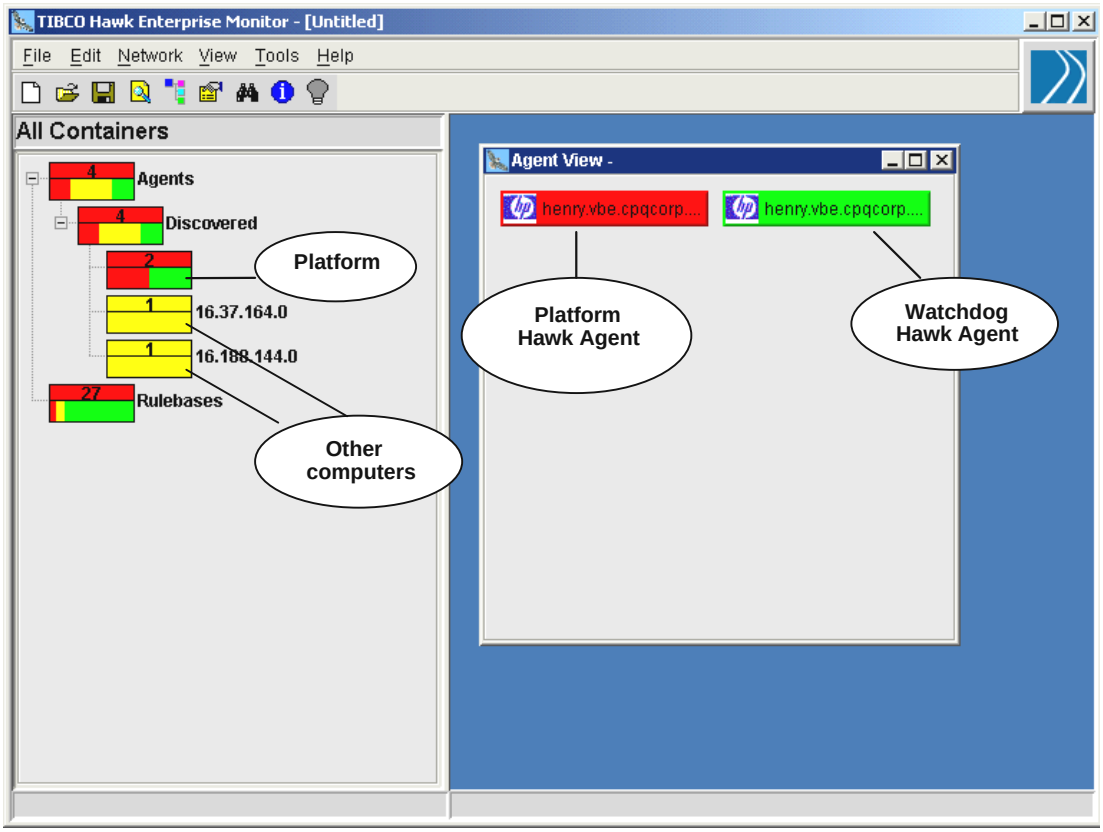
The OpenView SQM Admin Console is actually a TIBCO Hawk Display.

This Graphical User Interface can be used either on:

- A Windows box where it has been installed. To start an OpenView SQM Admin Console on Windows, users may use the shortcut created during the installation under the program group “HP OpenView -> SQM”.
- Or on a UNIX box where the SQM Kernel has been installed. To start an OpenView SQM Admin Console on UNIX, users have to invoke the command *`temip_sc_hawk_display_start`*.

The following figure shows the **TIBCO Hawk Display** console (Figure 8).

Figure 8: TIBCO Hawk Display console



TIBCO Hawk Display enables central monitoring of all OpenView SQM Applications.

In the Figure 8, all the Hawk Agents has been discovered. The discovery scope is determined by the TIBCO Rendezvous transport parameters used by the **TIBCO Hawk Display** and the TIBCO Hawk Agents. Note that in OpenView SQM, two TIBCO Hawk Agents are running on each host.

The agent is a process that monitors activity on a particular machine by processing loaded rule bases. It communicates with **TIBCO Hawk Display** using TIBCO Rendezvous messages. Rather than monitoring through one central console, agent activity is distributed across the entire network. Even though an agent communicates with instances of **TIBCO Hawk Display**, it operates independently of **TIBCO Hawk Display** and other agents.

In the “All Containers” panel, the rectangle (containing the number 2) represents the agent associated with the platform, and the others rectangles (containing the number 1) represent each host connected to the platform. When the agent associated with the platform is selected, two rectangles appear in the right panel named “Agent View”.

The first rectangle (red) represents the platform Hawk Agent named “hostname”-SCHA, associated to the platform, which gets all the information about the platform (such as Alerts or micro agents). The second rectangle (green) represents a watchdog Agent named “hostname”-SCHAW. This agent monitors modules. If it becomes red, it means that at least one of the SQM modules has abnormally stopped.

Monitoring the platform through TIBCO Hawk Display

The **TIBCO Hawk Display** tool can be used to monitor the OpenView SQM platform. It provides a graphical representation of the platform and its applications. In this representation, each container object is represented by an icon, the color of which depends on the highest alert found on discovered agents.

An icon can display up to six colors corresponding to the importance of the alarm, which are by default:

- Purple: The agent was discovered but currently no heartbeat message is being received. The lack of response could mean the agent is not running, the agent machine is down, or there is a network communication problem.
- Red: At least one high-level alert is active.
- Orange: At least one medium-level alert is active (but there are no high-level alerts).
- Yellow: At least one low-level alert is active (but there are no high-level or medium-level alerts).
- Blue: The agent is recovering. At least one alert was active within the last 30 minutes (default), but all active alerts were recently cleared or suspended.
- Green: No alerts are in effect.

Right clicking on the agent, as shown in the following Figure 9: TIBCO Hawk Display: Show Alerts sent by a TIBCO Hawk agent, opens two important submenus: Show Alerts and Get MicroAgents. The other submenus are not used in this context.

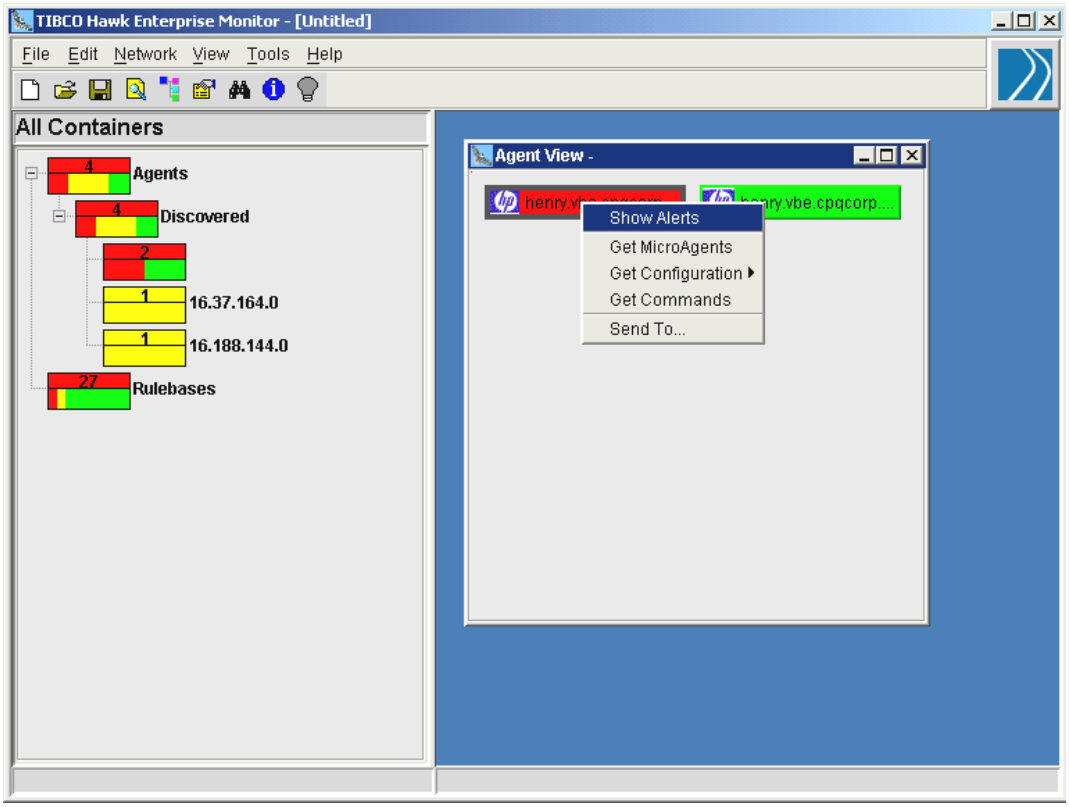
From the TIBCO Hawk Display it is possible to browse the lists of micro agents behind each TIBCO Hawk Agent. From here it is possible to invoke hawk methods on the micro agents.

Visualizing alerts of the OpenView SQM platform

The **TIBCO Hawk Display** provides an **Alert** Window, where the user can browse the current Alerts issued by the SQM components and sent by the Hawk Agents deployed on the distributed SQM platform

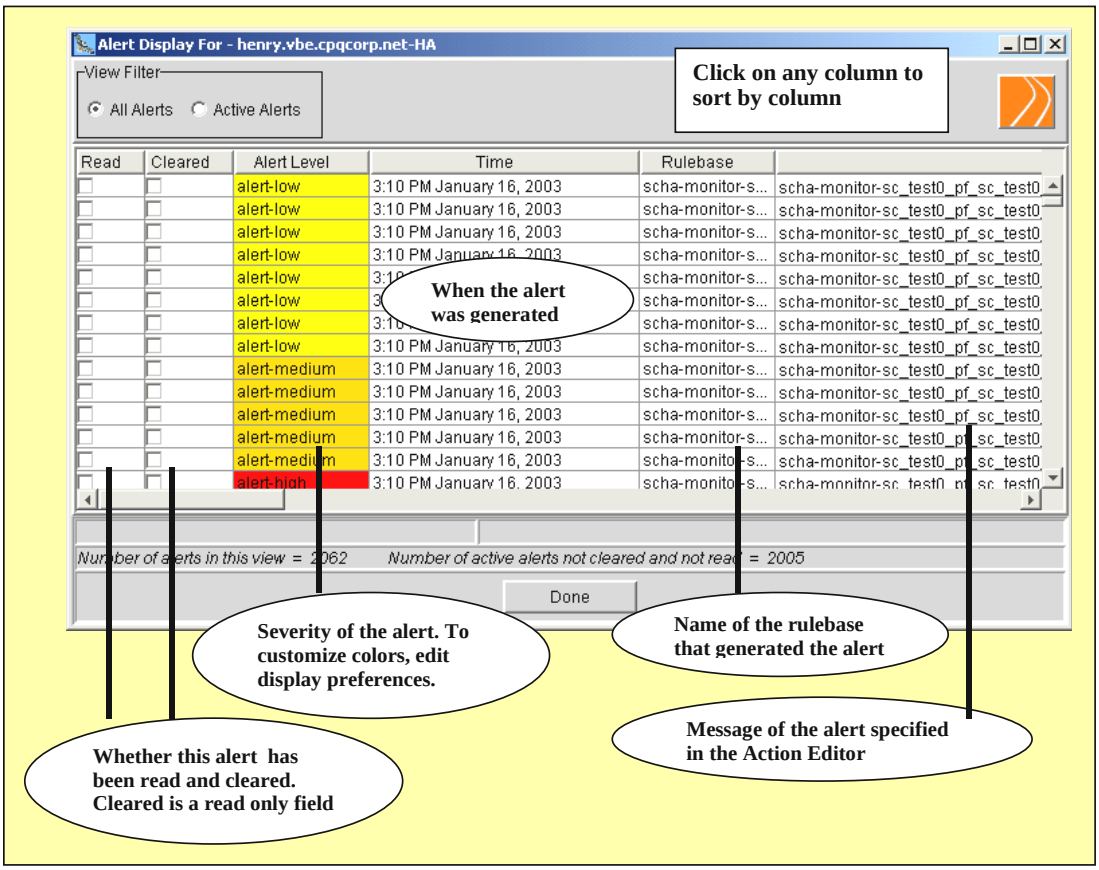
It is possible to browse these Alerts sent using the **Hawk Alert** Window.

Figure 9: TIBCO Hawk Display: Show Alerts sent by a TIBCO Hawk agent



Right-clicking on the platform agent and selecting the **Show Alerts** action (Figure 9) opens a **Display** window that displays a list of all alerts associated to the platform (Figure 10).

Figure 10: TIBCO Hawk Display: Result of Show Alerts on platform agent



Alerts are messages that an agent sends to the **TIBCO Hawk Display** when a specified condition occurs. TIBCO Hawk Rulebases that enforce your monitoring logic are used to generate alerts. In the **TIBCO Hawk Display**, the colors of each agent and container icon summarize alert levels, and the **Alert Display Window** shows alert details for a particular agent or all agents. In particular, you can use this window to visualize alerts sent by the OpenView SQM applications.

To see detailed information on an alert message, double-click on any field in the alert row: the following window appears (Figure 11).

Figure 11: TIBCO Hawk Display: Alert Detail Window

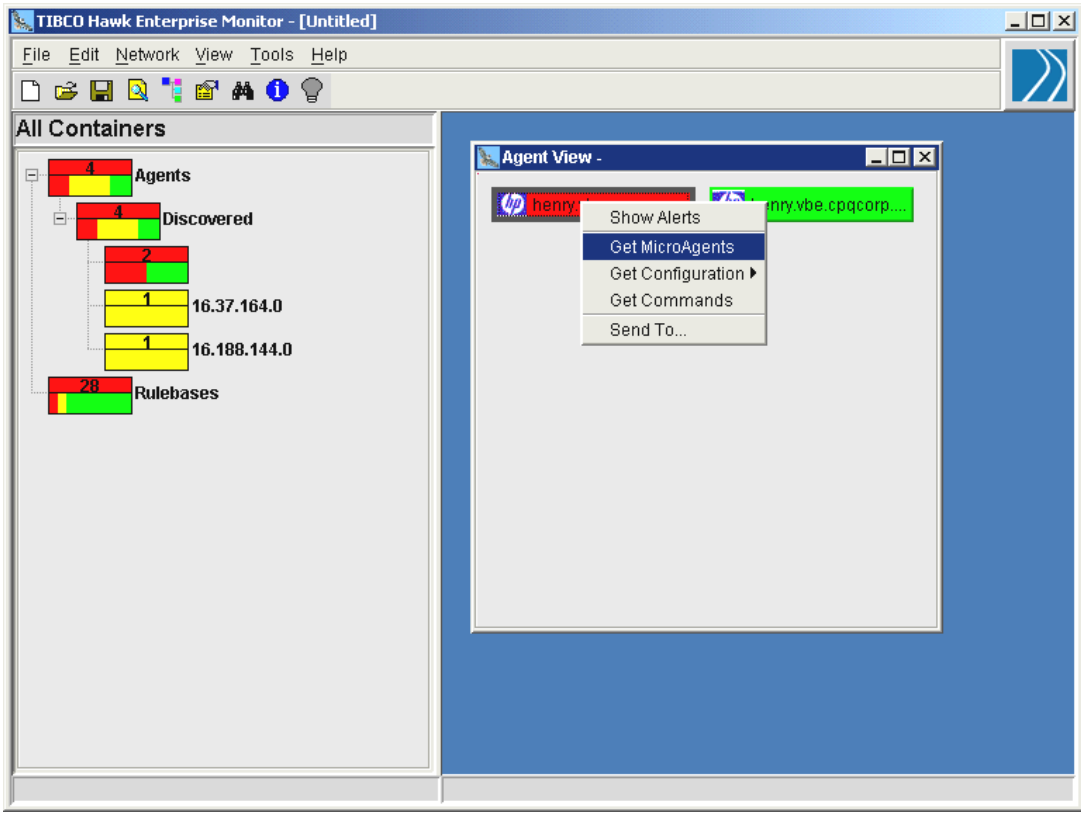
Understanding Alert contents

Alert contents and probable cause are described in the chapter SQM platform troubleshooting. See paragraph Alert logging.

Managing components

Each agent has a set of default micro agents, which are loaded when the agent is started. In the **TIBCO Hawk Display**, you can view micro agents and their methods for any discovered TIBCO Hawk Agent. The following example shows how to get a list of micro agents for the agent representing the platform (Figure 12).

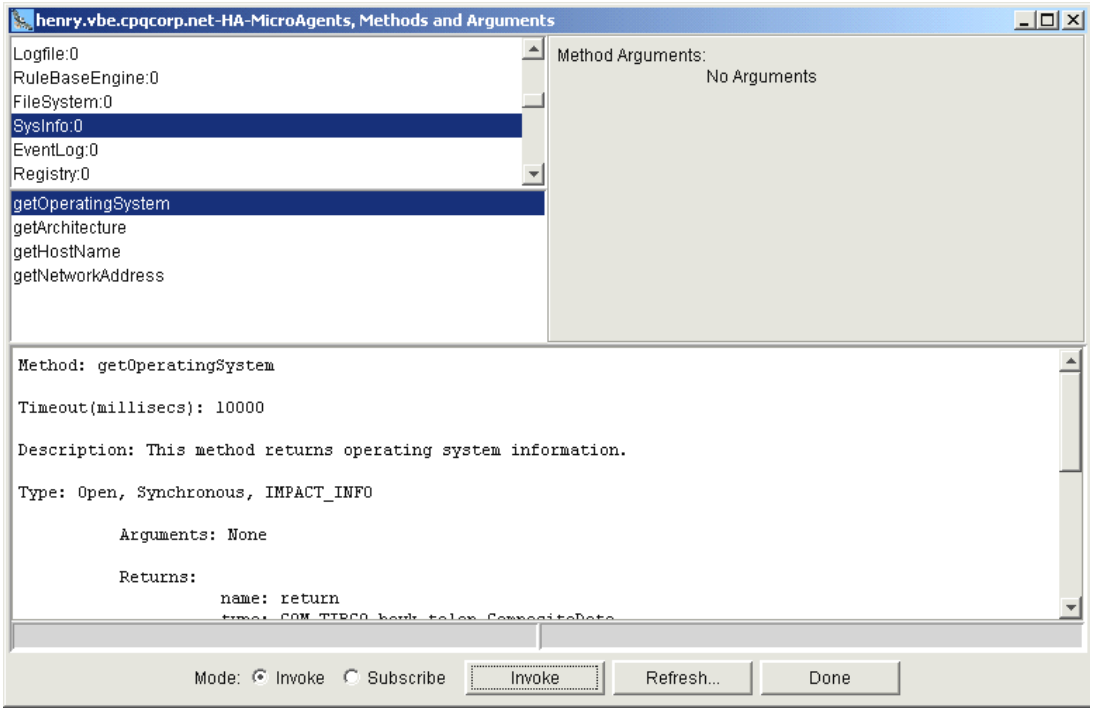
Figure 12: TIBCO Hawk Display: Get MicroAgents on platform agent



Two types of micro agents are displayed:

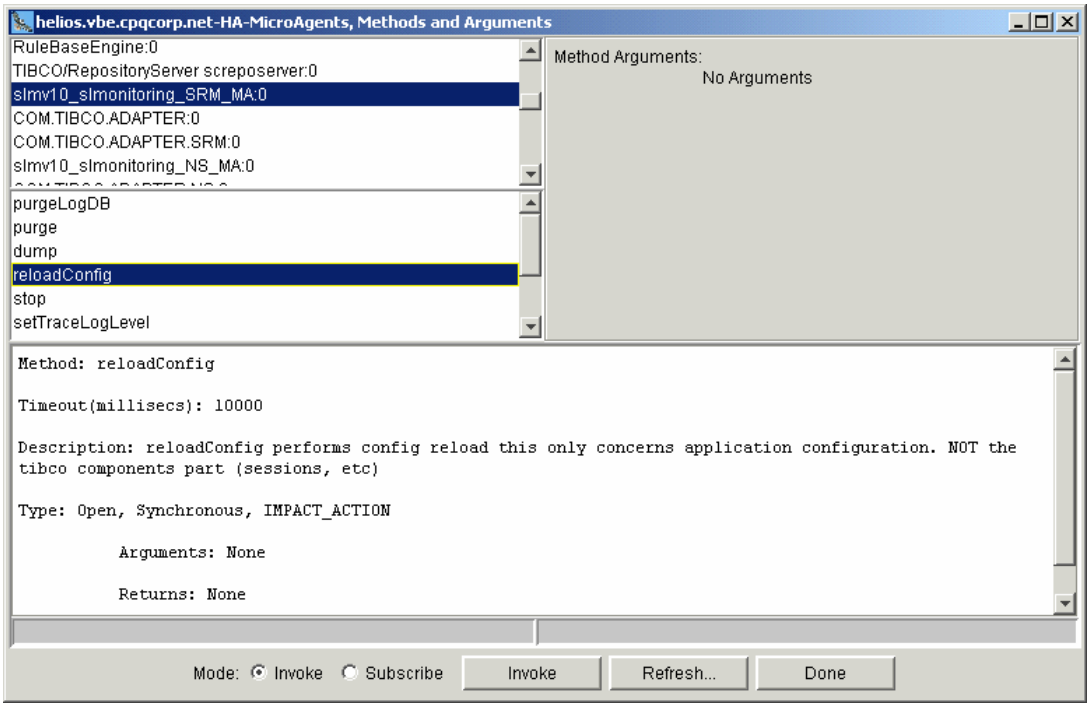
- TIBCO micro agents (Figure 13) that retrieve specific information about the system: For instance, CPU usage. To view the detailed information, click on the *Invoke* button.

Figure 13: TIBCO Hawk Display: TIBCO specific MicroAgent View



- Micro agents associated with the components running on the platform (Figure 14):

Figure 14: TIBCO Hawk Display: SQM component MicroAgent View



In this case, the **TIBCO Hawk Display** shows the applications running on the platform. In Figure 14, you can see the micro agent linked to the SRM application named:

- slmv12_slmonitoring_SRM_MA:0.

Convention

The convention for OpenView SQM Applications microagent names is:

<platform name>_<director name>_<application name>_MA:<Instance Id>.

For each micro agent, some AMI (Application Management Interface) functions are associated such as “dump”, “reloadConfig” or “setTraceLogLevel”.

Important

The *stop* AMI function is not supported through the **TIBCO Hawk Display.** This action must be done from the command line.

Monitoring status through the Watchdog Hawk agent (schwa)

Applications are monitored thanks to the monitoring rule handled by the main Hawk agent.

The **Show Alerts** submenu on the Watchdog Hawk agent does generally not show any errors. This agent is in charge of verifying the Hawk Agent status and restarting it in case of failure. Normally it does not show any alarm, because as soon as the main Hawk Agent is stopped, it is started again automatically.

2.1.6.2 Checking status of SQM Applications start-up

The SQM Components are started using the commands (e.g. `temip_sc_start_platform`) described in section Starting OpenView SQM. The start commands launch the initialization of the SQM Applications, but do not wait for the completion of their initialization, which can take some time depending on the applications tasks.

Starting banners

In order to allow checking of its actual startup when it has been completed, a starting banner is logged by each SQM Application. This banner indicates if the Application has successfully achieved its initialization, or has completely failed or has encountered some problem that will prevent to work properly. The banner will show keyword:

- successful, or
- failed, or
- degraded.

A successful NS starting banner is for instance:

> Application slmv12_slmonitoring_NS <*version*> started successfully

Retrieving starting banners

The starting banners are issued via low level Alerts, even the successful startup banners. These Alerts can be retrieved using the TIBCO Hawk Display console (see section Visualizing alerts of the OpenView SQM platform).

Users can also retrieve the starting banner of an SQM Application into its log, under the directory $TEMIP_SC_VAR_HOME/log, the log filename is:

> <*platformName*>_<*directorName*>_ <*applicationName*>.log

2.1.6.3 Monitoring the SQM platform through the command line

The SQM platform can be monitored from the command line by using the `temip_sc_show_platform –platform <platform>` command described in paragraph Show the SQM platform. This displays all components of the platform with their status (Running or not). With this command it is possible to see which SQM Application is running even if it is not on the local host.

Monitoring a host through the command line

The UNIX command **temip_sc_show** allows a quick display of the SQM processes running on the Host from where you launch the command. Those processes are: SQM Kernel UNIX processes and of all SQM applications running on the Host.

It does not take into account the Platform/Director configuration; therefore if you do not know all the applications that are supposed to run on this host, you will not notice if an application is not running. It is different from *temip_sc_show_platform* as it only displays components that are running on a host without displaying the ones that should be running but are not.

Command line syntax

`temip_sc_show`

Output

```
Applications
PID   UserId  Component  Platform  Director      Appli  Start   CPU%  Time   Size  State
5562  sqmadm  sqm_uis    slmv12    presentation  UIS    Nov 30  0.84  25:06  481   R
4692  sqmadm  sqm_dc     slmv12    slmonitoring  DC     Nov 30  1.77  10:20  537   R
4755  sqmadm  sqm_ns     slmv12    slmonitoring  NS     Nov 30  0.53  09:54  386   R
4011  sqmadm  sqm_slom   slmv12    slmonitoring  SLOM   Nov 30  1.17  08:22  608   R
4175  sqmadm  sqm_spdm   slmv12    slmonitoring  SPDM   Nov 30  1.19  38:05  836   R
7685  sqmadm  sqm_srm    slmv12    slmonitoring  SRM    Nov 30  0.74  30:44  495   R

Kernel
MINE  PID   UserId  Component                Platform  Start   CPU%  Time      Size  State
Yes   2738  sqmadm  rvd tcp:9900 -http 9799  slmv12    Nov 10  0.60  01:55:21  2242  R
Yes   7632  sqmadm  Repository               slmv12    Nov 17  0.51  27:18     505   R
Yes   2731  sqmadm  tibhawkhma               slmv12    Nov 10  0.18  13:52     355   R
Yes   3690  sqmadm  schawkevent              slmv12    Nov 10  0.29  34:39     685   R
```

```
Yes   2727  sqmadm  scha                    slmv12    Nov 10  1.51  01:21:01  2542  R
Yes   2729  sqmadm  schaw                   slmv12    Nov 10  1.07  51:10     1481  R
Yes   3692  sqmadm  Jacorb NS               slmv12    Nov 10  0.31  24:49     475   R
```

Monitoring SQM directors and applications through the command line

SQM directors and applications, even if they are not running on the local host, can be monitored through the command line using the following commands:

```
temip_sc_show_platform

temip_sc_show_director

temip_sc_show_application
```

For more details about these commands, see the paragraph Retrieve status of SQM platform.

Monitoring the SQM kernel

On UNIX, the *temip_sc_show_kernel* command displays the status of the SQM Kernel UNIX processes that are currently running on the local host:

Examples

On the **primary host** of the SQM platform, temip_sc_show_kernel displays the following processes and information. The SQM primary host is the system that hosts the Central Repository (identified as TIBRepository in the displayed status) and the SQM Kernel's Event Service (schawkevent).

```
MINE  PID    UID     Component      Status   Start Time   CPU Time
Yes   6426   sqmadm  rvd            R        Mar 5        105:42 tcp:11240 -http:9249
Yes   25220  sqmadm  tibhawkhma     R        Mar 10       1:13
Yes   25212  sqmadm  scha           R        Mar 10       13:33
Yes   25214  sqmadm  schaw          R        Mar 10       5:14
Yes   25227  sqmadm  TIBRepository  R        Mar 10       4:21
Yes   25241  sqmadm  schawkevent    R        Mar 10       8:39
Yes   25245  sqmadm  Jacorb NS      R        Mar 10       4:20
```

The SQM Kernels running on the **secondary hosts** do not contain the Central Repository or the Event Service. On a secondary host, temip_sc_show_kernel displays the following processes and information.

```
MINE  PID    UID     Component      Status   Start Time   CPU Time
Yes   6426   sqmadm  rvd            R        Mar 5        105:42 tcp:11240 -http:9249
Yes   25220  sqmadm  tibhawkhma     R        Mar 10       1:13
Yes   25212  sqmadm  scha           R        Mar 10       13:33
Yes   25214  sqmadm  schaw          R        Mar 10       5:14
```

Note: MINE indicates if the process is belonging to the user's SQM Kernel (Yes) or to another user's SQM Kernel (No). This is only useful if there are several SQM platforms running on the local host.

Monitoring TIBCO Rendezvous buses

TIBCO Rendezvous daemons provide an http console that can be used to get useful information about sessions and services. For more details about this console, please refer to section 2.2.4.

2.1.7 Operations specific to SQM components

In addition to the global administration commands shown in the previous paragraphs, SQM components may provide their own administration operations. This paragraph describes these additional operations for each of the SQM components. These operations are provided by methods called AMI (Application Management Interface). The **AMI** are invoked thanks to the

TIBCO Hawk sub-system (See paragraph Monitoring OpenView SQM using the Admin Console).

2.1.7.1 Service Repository Manager

The Service Repository Manager supports the following AMIs:

setTraceLogLevel, getTraceLogLevel, setMtLogLevel, getMtLogLevel

As for all other SQM components, the Service Repository Manager allows getting and setting of the trace level.

purge

Deletes physically the data marked as deleted in the Service Repository Manager.

purgeLogDB

Deletes the database log entries.

computeStatistics

Computes the statistics on database tables and indexes. After an important update of the model, the execution of this AMI will optimize (and speed up) the calls made to the database.

2.1.7.2 Service Performance Data Manager

The Service Performance Data Manager supports the following AMIs:

setTraceLogLevel, getTraceLogLevel, setMtLogLevel, getMtLogLevel like all other SQM components. And,

loadService

Argument: ServiceName

Loads the Calculation Engine for a given Service, or all Services if no service name is given in parameter.

purgeDbMethod

Argument: PugeType (Data or Model)

ServiceName

Depending on the chosen Purge type, this command purges the data or the models that have been previously deleted and are older than the retention delay. This can be done for a given service or for all services if no service name is given in the parameter.

In order to limit the size of the SPDM database, the purge AMI method of this component should be executed once a day.

Add an automatic purge in the crontab of the system. An example of script is given in the *OpenView SQM Installation Guide.*

Dump

As for the other SQM components, the Dump method creates a Dump file in the trace files directory:

Argument : Dump Mode, can be one of the following:

- Config: the current configuration loaded in the module.
- Memory: all the models and the current statuses
- Engines: gives the status for the Calculation Engines
- Topics: gives the topics to which the module is subscribing
- All: all of the above

2.1.7.3 Data Collector

The Data Collector supports the following AMIs:

setTraceLogLevel, getTraceLogLevel, setMtLogLevel, getMtLogLevel

As for all other SQM components.

Dump

As for the other SQM components, the Dump method creates a Dump file in the trace files directory:

Argument : Dump Mode, can be one of the following:

- Config: the current configuration loaded in the module.
- Memory: all the models and the current statuses
- Topics: the topics to which the module is subscribing
- All: all of the above (Config + Memory + Topics)

2.1.7.4 Naming Service

The Naming Service supports the following AMIs:

reloadConfig

As for the other OpenView SQM modules, the SQM can update the Naming Service configuration and ask the adapter instance to reload its updated configuration.

The Naming Service instance then commits the on-going transaction and applies the new configuration to the next transaction.

The outcome status of the reload command is sent at the end of the operation by the Naming Service Adapter as a TIBCO Hawk Alert message that indicates success or failure.

setTraceLogLevel, getTraceLogLevel, setMtLogLevel, getMtLogLevel

As for all other SQM components.

Dump

As for the other SQM components, the Dump method creates a Dump file in the trace files directory:

Argument : Dump Mode, can be one of the following:

- Config: the current configuration loaded in the module.
- Memory: all the models and the current statuses
- Topics: the topics to which the module is subscribing
- All: all of the above (Config + Memory + Topics)

2.1.7.5 Service Level Objective Manager

The Service Level Objective Manager supports the following AMIs:

setTraceLogLevel, getTraceLogLevel setMtLogLevel, getMtLogLevel

As for all other SQM components.

dump

As for the other SQM components, the Dump method creates a Dump file in the trace files directory:

Argument : Dump Mode, can be one of the following:

- Config: the current configuration loaded in the module.

- Memory: all the models and the current statuses
- Topics: the topics to which the module is subscribing
- All: all of the above (Config + Memory + Topics)

2.1.7.6 Logger

The Logger supports the following AMIs:

setTraceLogLevel, getTraceLogLevel, setMtLogLevel, getMtLogLevel

As for all other SQM components.

purge

Argument: Mode (Move or Delete)

Days

Datamart (True or False)

The purge directive allows you to purge data from the logger Database depending on the age of the Data (1 day, 2 days and so on). If called with the Datamart option, it also allows you to purge data that have been flagged as used by the Data Mart module.

dump

As for the other SQM components, the Dump method creates a Dump file in the trace files directory:

Argument : Dump Mode, can be one of the following:

- Config: the current configuration loaded in the module.
- Memory: all the models and the current statuses
- Topics: gives the topics to which the module is subscribing
- All: all of the above

reloadConfig

As for the other OpenView SQM modules, the SQM can update the Logger configuration and ask the adapter instance to reload its updated configuration.

The Logger instance then commits the on-going transaction and applies the new configuration to the next transaction.

2.1.7.7 User Interface Server

The User Interface Server supports the following AMIs:

setTraceLogLevel, getTraceLogLevel setMtLogLevel, getMtLogLevel

As for all other SQM components.

Dump

As for the other SQM components, the Dump method creates a Dump file in the trace files directory:

Argument : Dump Mode, can be one of the following:

- Config: the current configuration loaded in the module.
- Memory: all the models and the current statuses
- Topics: the topics to which the module is subscribing
- All: all of the above (Config + Memory + Topics)

2.1.7.8 SLA Administration UI

The SLA Administration UI does not provide any specific AMI.

2.1.7.9 Service Adapters

This chapter describes only AMI common to all SQM Service Adapters.

The following self-management commands are available:

setTraceLogLevel, getTraceLogLevel setMtLogLevel, getMtLogLevel

As for all other SQM components.

Dump

As for the other SQM components, the Dump method creates a Dump file in the trace files directory:

Argument : Dump Mode, can be one of the following:

- Config: the current configuration loaded in the module.
- Memory: all the models and the current statuses
- Topics: the topics to which the module is subscribing
- All: all of the above (Config + Memory + Topics)

quietMode: stops the service adapter instance from publishing performance messages on the collection bus.

reloadConfig: prompts the service adapter instance to reload its configuration. This directive stops all data collection and re-activates them with the latest configuration data. The following application parameters can be reloaded using this directive:

- pollingPeriod (minimum value is 0.5 which corresponds to 30 seconds)
- RequestRepliesNbRetry
- internalRequestRepliesTimeout

2.1.7.10 Gateways

This chapter describes only AMI common to most of the Gateways. If a Gateway implements a specific AMI (related to specific features of the application), please refer the appropriate Gateway Installation and Configuration Guide.

The following self-management commands are available:

setTraceLogLevel, getTraceLogLevel setMtLogLevel, getMtLogLevel

As for all other SQM components.

Dump

As for the other SQM components, the Dump method creates a Dump file in the trace files directory:

Argument : Dump Mode, can be one of the following:

- Config: the current configuration loaded in the module.
- Memory: all the models and the current statuses
- Topics: the topics to which the module is subscribing
- All: all the above data (Config + Memory + Topics)

quietMode: stops the Gateway instance from publishing performance messages on the collection bus.

reloadConfig: prompts the Gateway instance to reload its configuration. This directive will reload the Gateway application parameters. Please refer to the appropriate Gateway Installation and configuration Guide to have the list of available application parameters which can be reloaded after modification.

2.1.8 Backup/Restore

2.1.8.1 Oracle backup/restore

Oracle databases backup/restore must be done using the Oracle backup/restore policy, which is described in the *Oracle Configuration Guide*. Please refer to this documentation for further details.

2.1.8.2 Configurations backup/restore

What can be backed up

Configurations that can be backed up can be divided into different parts:

- Application configuration
- Tibco configuration
- Oracle configuration
- Jacorb Configuration

What is restored

The Restore procedure does not restore all the SQM productss. Only “dynamic” data will be restored. You need to reinstall SQM and deploy it before restoring the data.

Backup/restore tools

The **temip_sc_backup** command backs up or restores the OpenView SQM platform configuration files.

Command line syntax

```
temip_sc_backup
```

Parameters description

-g/-group <*groupname1 groupname2 ...*>: specify group name(s) to be backed up (Valid groups are - appli, tibco, oracle and jacorb).

-o/-output <*directory_name*>: Output directory to store tar files. The current directory is the default output directory.

-r/-restore : To restore the tar files from the output directory specified in -o option, to the directory specified in the environment variable TEMIP_SC_VAR_HOME. If no output directory is specified, the current directory is used to retrieve the tar files.

-q/-quiet : Do not print progress messages for normal successful status. Error messages will still be printed.

Examples

temip_sc_backup
-- by default all data directories will be backed up.

temip_sc_backup -g oracle
-- backs up the Oracle directory, saves backup files in the current directory.

temip_sc_backup -g tibco, appli -o /var/opt/backup
-- backs up the Tibco configuration and the application configuration and saves the backup files to the output path specified, /var/opt/backup

temip_sc_backup -restore
-- Restores files from the tar file of the current directory in the directory specified by TEMIP_SC_VAR_HOME

temip_sc_backup -restore -o /var/opt/backup
-- Restores files in the directory specified in the environment variable TEMIP_SC_VAR_HOME from the tar file located in the directory /var/opt/backup.

temip_sc_backup -restore -g tibco, appli -o /var/opt/backup
-- Restores files in the directory specified in the environment variable TEMIP_SC_VAR_HOME from the tar files corresponding to the groups tibco and appli located in the directory /var/opt/backup

Note

Backup operations must be executed using the root or sqmadm accounts only

2.1.8.3 Backup on Windows

No specific tool is provided to make backup on Windows. Windows backup tools can be found in the market.

The SQM GUIs have few files that need a backup:

- SLA Administration
  - Available Actions description file.
    `%TEMIP_SC_HOME%\UI\SLAClient\properties\AvailableActions.xml`
    This file has to be added to the backup procedure if it has been customized.
  - Customization files
    Files in `%USERPROFILE%\HP OpenView\SQM\UI\SLAClient\config` directory can be customized (by the SQM Administrator) in order to change some settings (timeouts, rendering formats, tracing configuration …).
    These files have to be added to the backup procedure if they have been customized.
- SLA Monitoring
  - Customization files
    Files in `%USERPROFILE%\HP OpenView\SQM\UI\SLMClient\config` directory can be customized (by the SQM Administrator) in order to change some settings (timeouts, rendering formats, tracing configuration …).
    These files have to be added to the backup procedure if they have been customized.
- SLA Administration & SLA Monitoring
  - Jacorb Configuration
    `%USERPROFILE%\HP OpenView\SQM\Jacorb\jacorb.properties` contains

the Jacorb settings. These files have to be added to the backup procedure if they have been customized.

- Last Login Information
  `%USERPROFILE%\HP OpenView\SQM\UI\config\TeSCUIServerLocation.properties` stores last login information (except password) entered in the login dialog box. This file can be included in the backup procedure but it is not mandatory. If no file is present, the login box will be empty; the operator will have to reenter his name and the name of the server.

2.1.9 Licensing

2.1.9.1 AutoPass product

SQM Licensing is based on the OpenView AutoPass product.

The SQM utility to manage the licenses is *`temip_sc_license`*.

Note

Any call to the *temip_sc_license* utility must be done under the **root** user and with all the **SQM environment variables** loaded as described in section 1.7.4).

Use of AutoPass GUI to get passwords

The installation of the permanent password can be done with the command # *`temip_sc_license -get`*. This command calls the Autopass GUI.

Licenses must be installed on hosts running Presentation, Acquisition and SLMonitoring directors.

There are two possible situations depending on whether your platform is connected to the internet or not:

Connection to the Internet available

Launch the AutoPass GUI (through # *`temip_sc_license -get`*) to connect to the Password Delivery Center at https://webware.hp.com/welcome.asp.

In the information requested, the reference of the Purchase Order is the key information to access the list of the LTU (License To Use) purchased, i.e. the Permanent Passwords available for each product.

You have to select a subset of LTU among the LTU available for this Purchase Order. You can later connect again to the Password Delivery Center with the same Purchase Order to get a new subset of LTU for a new server.

Example

You have purchased 1 SQM-CORE LTU, 1 SQM-SLA LTU and 2 SQM-USER LTU. Your platform is made of two HP-UX servers.

First, you connect to the Password Delivery Center from host A. You select 1 SQM-CORE LTU, 1 SQM-SLA LTU and 1 SQM-USER LTU giving the capacity on host A to:

- Manage 50 SLA simultaneously (SQM-SLA)
- Have 2 users connected to the Presentation director running locally

Then, you connect to the Password Delivery Center from host B and select the remaining SQM-USER LTU giving the capacity on host B to:

- Have 2 users connected to the Presentation director running locally

As Host B does not run the SLM director, it is not necessary to install a SQM-CORE password on Host B.

No Connection to the Internet

If you have no direct access to the internet from the SQM platform, you can contact the Password Delivery Center by phone, Fax, or e-mail. Your permanent passwords will be then returned to you as a Fax or as an attachment to an e-mail.

The AutoPass GUI (through *#temip_sc_license –get*) can be used to generate a license request form that can be sent directly to HP. This GUI also allows you to import directly the file attached to the e-mail.

If you receive your passwords by Fax, you have to enter the license keys manually. This last solution is not recommended because OVKey4 passwords are long and complex.

The distribution of the LTU among the different servers is done manually using:

```
# temip_sc_license –import/export.
```

For example, you can import a set of permanent license keys using:

```
# temip_sc_license -import
```

Through the GUI, to import a previously exported license:

- Select a flat license file using ‘browse’ window
- Once selected, click ‘Choose’ button
- Click ‘view file contents’ button
- Select the license
- Click ‘import’ button

Exporting the license in a file is the same procedure.

Note

You cannot distribute the same license on several hosts. Indeed, if you purchase a 50 SQM_SLA license, you can only install it on one host. If you plan to distribute on two hosts, purchase a 25 SQM_SLA license for host 1 and a 25 SQM_SLA license for host 2.

2.1.9.2 Licenses

SQM implements the following license keys.

SQM-CORE License

The SQM-CORE license is the main license for the core of the SQM platform.

You need to purchase only one SQM-CORE license per platform. This license has to be installed on all the HP-UX servers running the SLM director.

At platform installation, a temporary SQM-CORE license is installed. This temporary license allows a 90 days trial period. If AutoPass was not installed, the user will have to install it, then configure licensing by running *#temip_sc_license –setup.*

SQM-SLA License

The SQM-SLA license key allows the unlocking of a defined number of SLAs. You need to purchase one SQM-SLA license per 50 SLA.

At platform installation, a temporary SQM-SLA license is installed. This temporary license allows a 90 days trial period. If AutoPass was not installed, the user will have to install it, then configure licensing by running *#temip_sc_license –setup.*

The capacities associated to each SQM-SLA license are aggregated. For example, if you install 2 SQM-SLA licenses on the same server, you will get a 100 SLA capacity.

SQM-USER License

The SQM-USER license key allows the support of a defined number of concurrent users. You need to purchase one SQM-USER License per two concurrent users.

At platform installation, a temporary SQM-USER license is installed. This temporary license allows a 90 days trial period. If AutoPass was not installed, the user will have to install it, then configure licensing by running *#temip_sc_license –setup.*

The capacities associated to each SQM-USER LTU are aggregated: if you install 2 SQM-USER LTUs on the same server, you will get a 4 concurrent users capacity

SQM-SA-SIMPLE License

The SQM-SA-SIMPLE license key allows the installation and usage of a Service Adapter of type Simple. This type depends on the Service Adapter, so the user has to refer to the Service Adapter documentation to check for the appropriate license. You need to purchase one SQM-SA-SIMPLE license per Service Adapter of type Simple running on the SQM Platform.

At platform installation, a temporary SQM-SA-SIMPLE license is installed. This temporary license allows a 90 days trial period. If AutoPass was not installed, the user will have to install it, then configure licensing by running *#temip_sc_license –setup.*

SQM-SA-PREMIUM License

The SQM-SA-PREMIUM license key allows the installation and usage of a Service Adapters of type Premium. This type depends on the Service Adapter, so the user has to refer to the Service Adapter documentation to check for the appropriate license. You need to purchase one SQM-SA-PREMIUM license per Service Adapter of type Premium.

At platform installation, a temporary SQM-PREMIUM license is installed. This temporary license allows a 90 days trial period. If AutoPass was not installed, the user will have to install it, then configure licensing by running *#temip_sc_license –setup.*

SQM-SA-GENERATED-RT License

The SQM-SA-GENERATED-RT license key allows the installation and usage of a Runtime Service Adapter generated by a Toolkit. This type depends on the Service Adapter, so the user has to refer to the Service Adapter documentation to check for the appropriate license. You need to purchase one SQM-SA-GENERATED-RT license per Service Adapter of type Runtime.

At platform installation, a temporary SQM-SA-GENERATED-RT license is installed. This temporary license allows a 90 days trial period. If AutoPass was not installed, the user will have to install it, then configure licensing by running *#temip_sc_license –setup.*

SQM-SA-STANDARD License

The SQM-SA-STANDARD license key allows the installation and usage of a Service Adapters of type Standard. This type depends on the Service Adapter, so the user has to refer to the Service Adapter documentation to check for the appropriate license. You need to purchase one SQM-SA-STANDARD license per Service Adapter of type Standard.

At platform installation, a temporary SQM-STANDARD license is installed. This temporary license allows a 90 days trial period. If AutoPass was not installed, the user will have to install it, then configure licensing by running *#temip_sc_license –setup.*

SQM-SA-ADVANCED License

The SQM-SA-ADVANCED license key allows the installation and usage of a Service Adapters of type Advanced. This type depends on the Service Adapter, so the user has to refer to the Service Adapter documentation to check for the appropriate license. You need to purchase one SQM-SA-ADVANCED license per Service Adapter of type Advanced.

At platform installation, a temporary SQM-SA-ADVANCED license is installed. This temporary license allows a 90 days trial period. If AutoPass was not installed, the user will have to install it, then configure licensing by running *#temip_sc_license –setup.*

SQM-SQL-SA-TOOLKIT License

The SQM-SQL_SA_TOOLKIT license key allows the installation and usage of a SQL SA Toolkit.

At platform installation, a temporary SQM-SQL-SA-TOOLKIT license is installed. This temporary license allows a 90 days trial period. If AutoPass was not installed, the user will have to install it, then configure licensing by running *#temip_sc_license –setup.*

2.1.9.3 License management utility

The **temip_sc_license** allows managing licenses on SQM.

Command line syntax

```
temip_sc_license
```

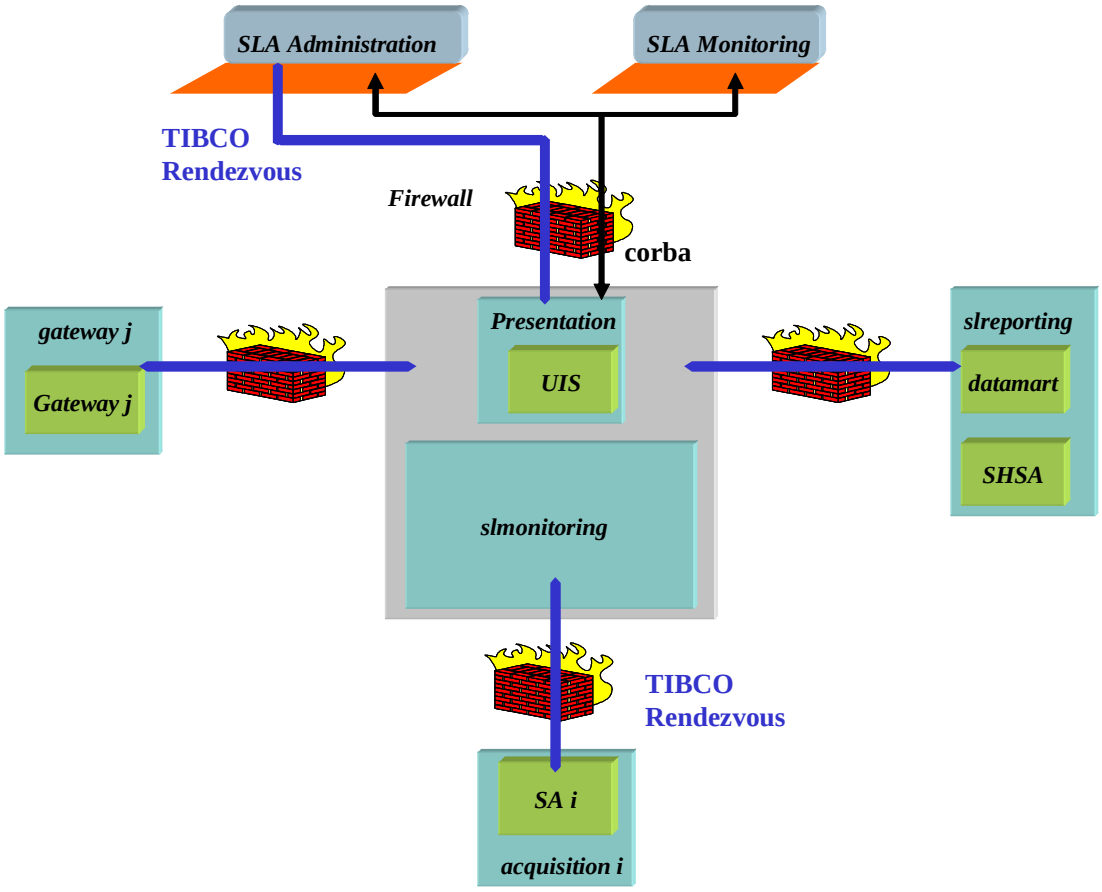
Parameters description

-setup	:	configure license environment and isntall trial licenses (90 days) (root privilege required)
-get	:	launch AutoPass GUI in order to retrieve licenses (root privilege required)
-export	:	launch AutoPass GUI in order to export a license to a text file (root privilege required)
-import	:	launch AutoPass GUI in order to import a license from a text file (root privilege required)
-check	:	check installed licenses

2.2 Advanced Administration

The next figure shows the SQM buses that may have to cross a firewall.

Figure 15: SQM and firewalls



2.2.1 SQM directors and firewalls

- The Corba channels are established between the Presentation Server and its connected SLA Monitoring and SLA Administration GUIs. Section 2.2.2 describes the TCP ports that the firewall must enable for these communications.
- Some of the TIBCO Rendezvous buses depending on the director deployed beyond the firewall:
    - For an Acquisition director (i.e. containing a Service Adaptor), the TIBCO Rendezvous buses that will cross the firewall are: `RepositorySession, SelfMgmt, ServiceRepManager, RawCollection` and `FaultTolerance`.
    - For the Service Level Reporting director, the TIBCO Rendezvous buses that will cross the firewall are: `RepositorySession, SelfMgmt and ServiceRepManager for the Datamart application and in addition RawCollection` and `FaultTolerance` for the SHSA application.
    - For a Gateway director the TIBCO Rendezvous buses that will cross the firewall are: `RepositorySession, SelfMgmt, ServiceRepManager SecondaryData` and `ObjectiveStatus`.

To cross a firewall, those TIBCO Rendezvous buses have to be routed by TIBCO Rendezvous router daemons, called RVRD, deployed on each side of the firewall. The RVRD is in fact a RVD that provides additional features to route broadcast/multicast Rendezvous buses onto a simple TCP connection. Such a route has to be configured for each TIBCO Rendezvous bus

that will cross a firewall and this firewall must authorize the TCP connection used by the route. Section 2.2.5 describes how to use and configure a RVRD.

2.2.2 SLA Monitoring and SLA Administration GUIs behind a Firewall

If the communications between the SLA Monitoring GUI (or the SLA Administration GUI) and its Presentation Server has to cross a firewall, you need to configure this firewall to authorize bidirectional communication on the TCP ports used by CORBA.

- One is the TCP port stored in the variable of environment TEMIP_SC_JACORB_NAME_SERVER_PORT (default value is 7171)
- One if defined in the property "ClientCommunicationCorbaPort" in the Presentation Server configuration of the TIBCO Repository (See Section 3.1 and 3.3.8)

If the communications between the SLA Administration GUI and the director "slmonitoring" have to cross a firewall, you need to configure this firewall to authorize bidirectional communication connections on the TCP ports used by TIBCO. Those are TCP ports identified by the following variables of environment (declared on the UNIX director "slmonitoring")

- `RepositorySession`
  The corresponding port is identified by the following variable environment (declared on the UNIX director "slmonitoring")
  $TEMIP_SC_REPOSITORY_SESSION_SERVICE
  (default value = Starting Range Port Number + 8)
- `ServiceRepManager`
  The corresponding port is identified by the following variable environment (declared on the UNIX director "slmonitoring")
  $TEMIP_SC_SDSI_RV_SERVICE
  (default value = Starting Range Port Number + 5)

2.2.3 Switching off the multicast IP used by TIBCO Rendezvous

Normal usage of TIBCO Rendezvous messaging bus is based on broadcast or multicast IP protocol. If your network administration requires you to switch the usage of such broadcast or multicast IP off, you can configure the TIBCO Rendezvous daemons to do not use broadcast/multicast IP.

- You may decide this option before the setup of SQM (see section 2.2.7). Edit the platform description file you plan to use for the SQM setup (e.g. $TEMIP_SC_HOME/tmp/platform_desc.cfg), and add the optional attribute ***`multicastFlag`*** of the configuration element ***`MessagingDaemons`***. Its values must be ***`False`***:

  ```
  <MessagingDaemons routerFlag="True" multicastFlag="False">
  ```

- Or you may decide to switch off the broadcast/multicast IP after the setup of SQM. Edit your environment profile (`temip_sc_env.sh` on UNIX & `temip_sc_env.bat` on WINDOWS) and set :

  TEMIP_SC_RV_MULTICAST_FLAG=**FALSE** # On UNIX

  set TEMIP_SC_RV_MULTICAST_FLAG=**FALSE** # On WINDOWS

  You need to restart the SQM Kernels to apply these environment changes.

Important

Be careful that, when you disable the broadcast/multicast IP, you cannot distribute the SQM Platform, or you have to use the **RVRD** (see section 2.2.5) rather than the usual TIBCO Rendezvous daemon **RVD**, even for a typical deployment. If no RVRD route is defined between two hosts, for a given SQM TIBCO Rendezvous bus, the messaging will not work properly between these two hosts.

2.2.4 RVD HTTP Service

Using your HTTP browser, you can access the administration console provided by the RVD daemon. It displays information about the TIBCO Rendezvous buses currently established by the SQM platform, the connected clients and their subscriptions. Such a bus is identified by a service port number. You may retrieve useful information about message transports on each session, e.g. inbound or outbound rates and, more importantly, about the **Alerts detected by the RVD daemon** that reflect an abnormal condition such as messages lost and retransmission due, for instance, to a physical network trouble.

Connecting

Define the TEMIP_RVD_RVRD_HTTP_PORT environment variable in your *temip_sc_env.sh* file:

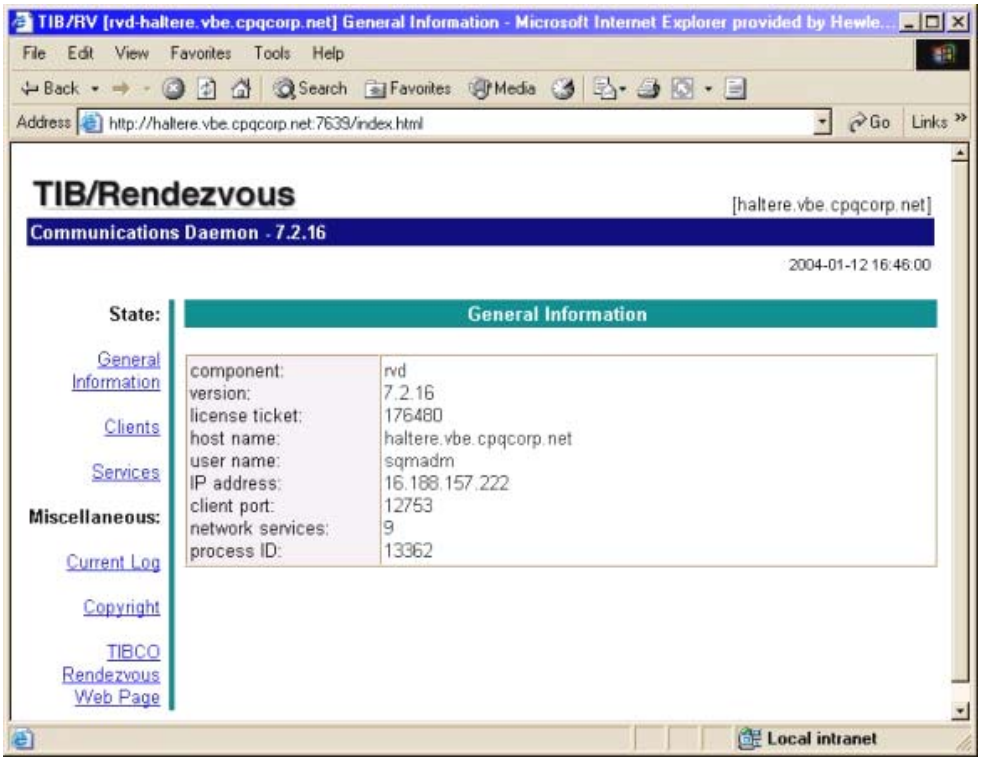
Figure 16: RVD HTTP port

```
#
# RVD (or RV routing daemon) administration http port
#
TEMIP_RVD_RVRD_HTTP_PORT=12509
export TEMIP_RVD_RVRD_HTTP_PORT
```

On your Web browser, type the following URL:
http://hostname:$TEMIP_RVD_RVRD_HTTP_PORT

Information Page

Figure 17: TIBCO Rendezvous daemon information page



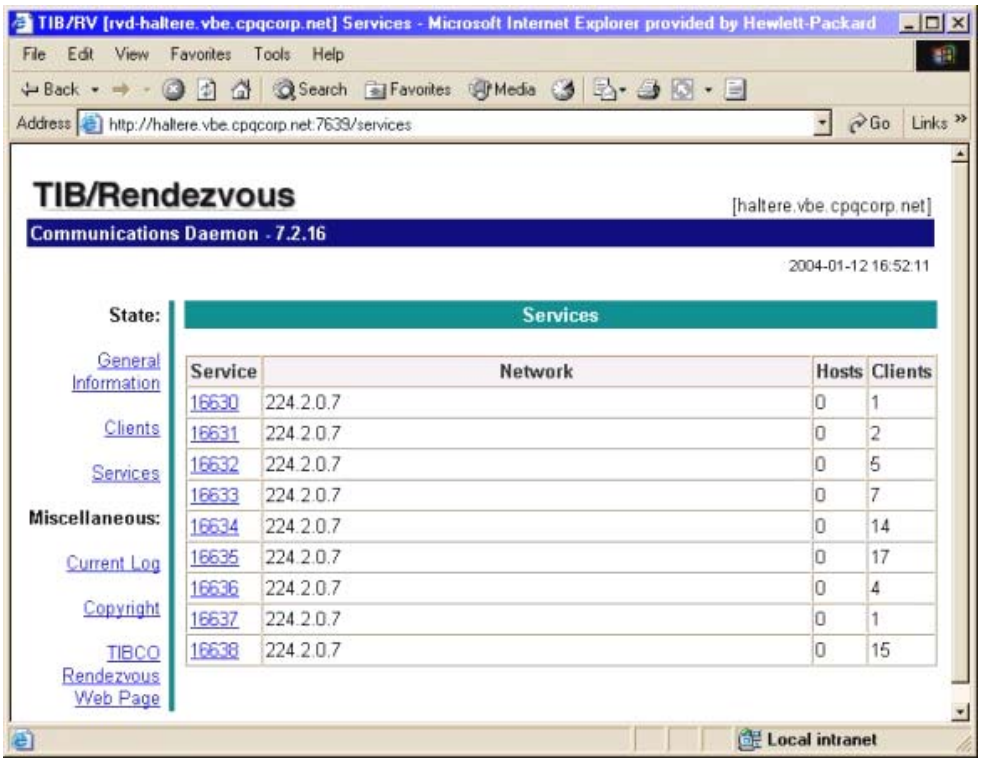
This information page contains important information such as:

- User name: this has to be `sqmadm`
- Client port: indicates the TCP port to which the RVD is connecting
- Network services: the number of currently enabled TIBCO logical buses

Services Page

Clicking on the **services** link displays the following page:

Figure 18: TIBCO Rendezvous Services page



This page lists all the services (i.e. logical buses) currently available on this system.

For each service, the following information is displayed:

- Hosts: The number of remote hosts connected to this logical bus
- Clients: The number of adapters with opened sessions on this logical bus

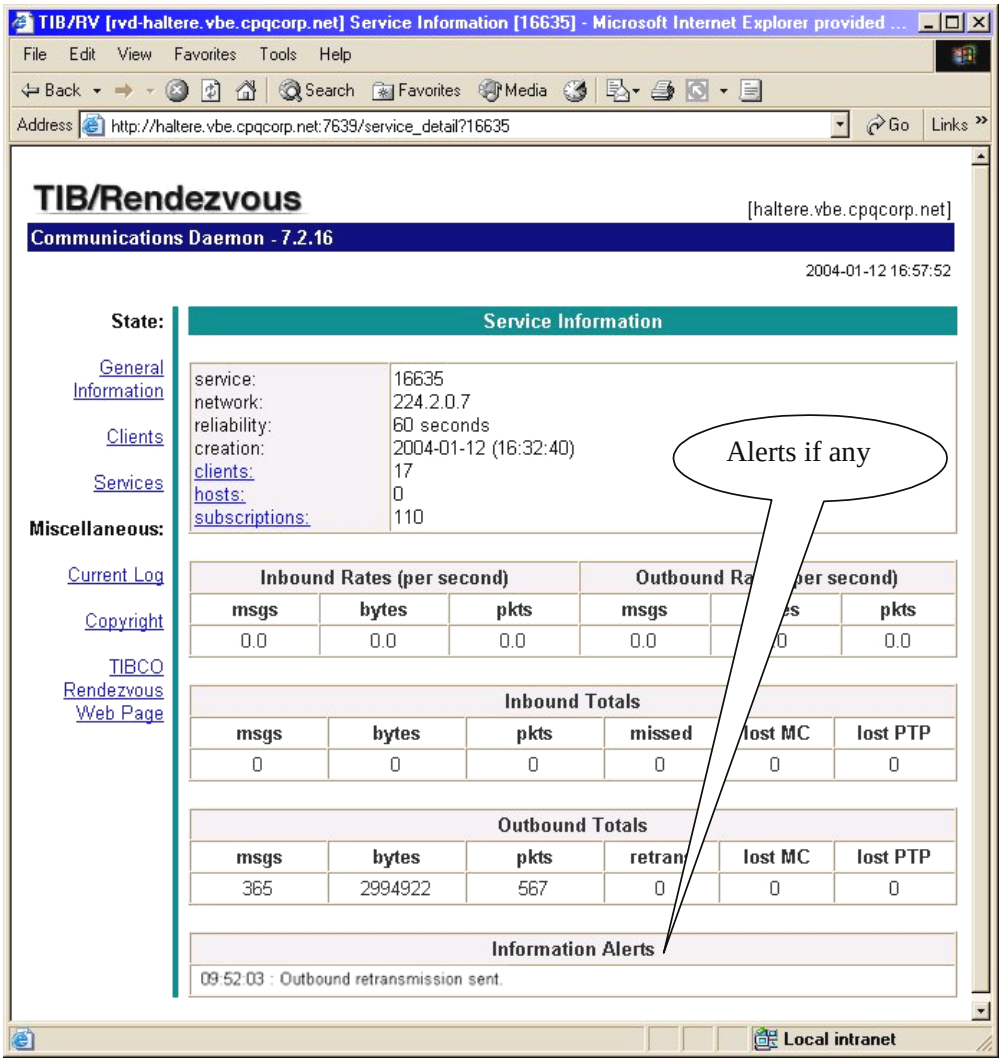
Below is a simple UNIX command that can be used to retrieve the TIBCO Rendezvous Service associated with each port listed in the Service List:

```
# env | grep SERVICE

TEMIP_SC_EVENT_SERVICE_FLAG=TRUE
TEMIP_SC_REPOSITORY_SESSION_SERVICE=16638
TEMIP_SC_RAW_COLL_RV_SERVICE=16630
TEMIP_SC_PRIM_DATA_RV_SERVICE=16631
TEMIP_SC_SECOND_DATA_RV_SERVICE=16632
TEMIP_SC_SDSI_RV_SERVICE=16635
TEMIP_SC_SVD_RV_SERVICE=16633
TEMIP_SC_NS_RV_SERVICE=16636
TEMIP_SC_FT_RV_SERVICE=16637
TEMIP_SC_SELF_MGMT_RV_SERVICE=16634
```

Service Detailed information and Alerts Page

Figure 19: TIBCO Rendezvous Service information page



This page contains detailed information about a selected service (in this case about the service running on port 16635). The Information Alerts section contains a list of alert messages that have been dispatched by the service.

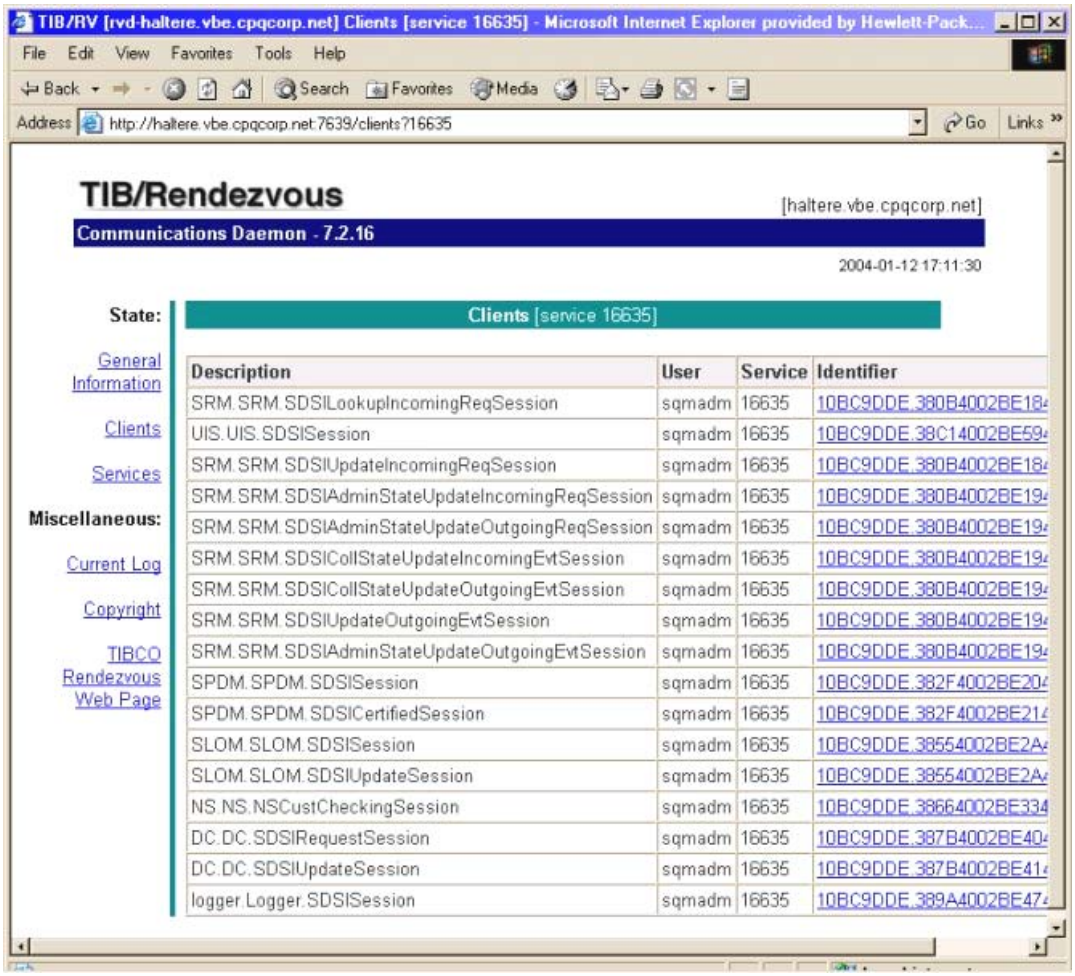
Under normal conditions, no Alert should be detected by a RVD daemon. An Alert may reflect a problem with the physical network usage or configuration, or with one of the connected hosts, for instance because the local RVD is occasionally starved of CPU.

Among the following (non exhaustive) Alert message examples, the last one is the worst:

- 17:49:59: Outbound retransmission sent.
- 17:45:49: Inbound data packets missed.
- 17:42:49: Inbound data loss has occurred.
- 17:45:49: Inbound data loss, sequence falls outside window.

Clients Page

Figure 20: TIBCO Rendezvous Clients page



This page lists the SQM Components that have opened sessions on the selected service.

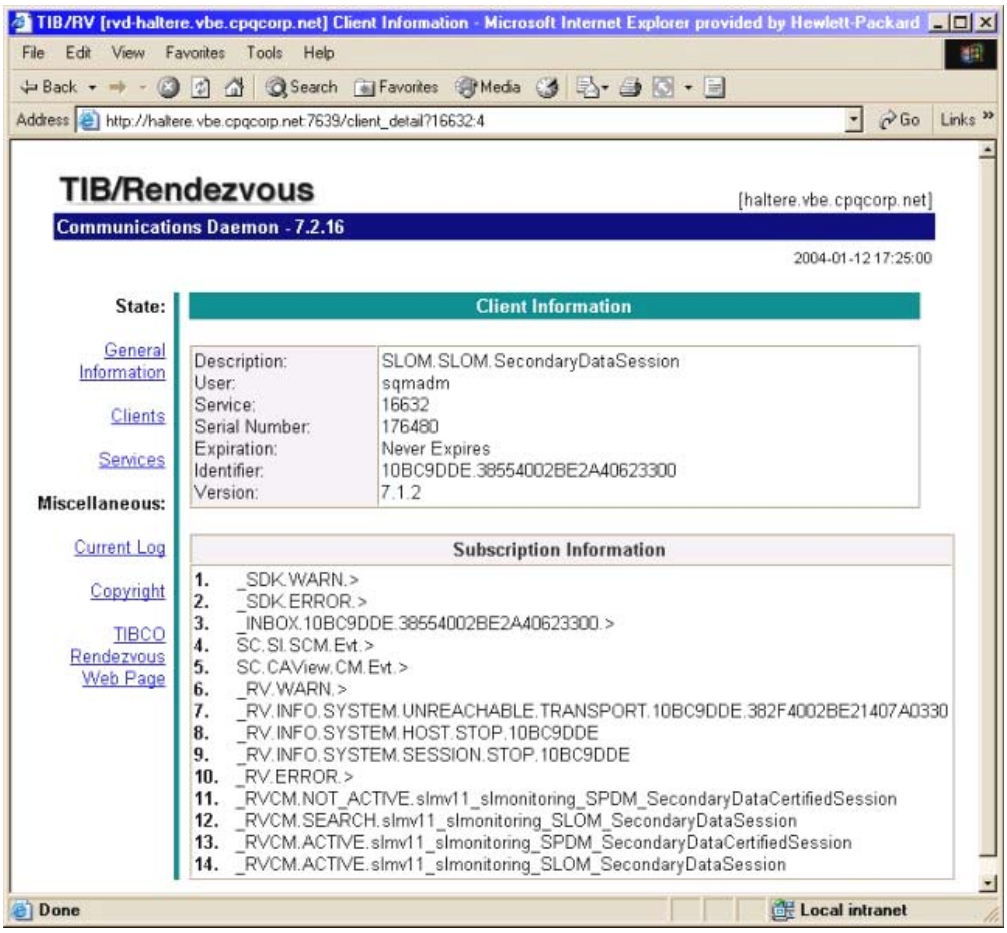
Note

The user must be sqmadm.

Client Detailed information

The following page shows information on the Sessions and Subscriptions (with topics) on this service.

Figure 21: TIBCO Rendezvous Clients detailed information page



This page contains information on a given session and on the topics listened to in this session.

2.2.5 RVRD configuration

2.2.5.1 Overview

The TIBCO Rendezvous daemon **RVRD** must be used at the place of **RVD** in the following situations:

- The physical networks that connect the distributed SQM Components do not support IP multicast or IP broadcast routing (see section 5.3).
- Participating networks are separated by a firewall.
- Participating networks lie in distant geographic areas.
- Messages must traverse expensive or slow WAN links.

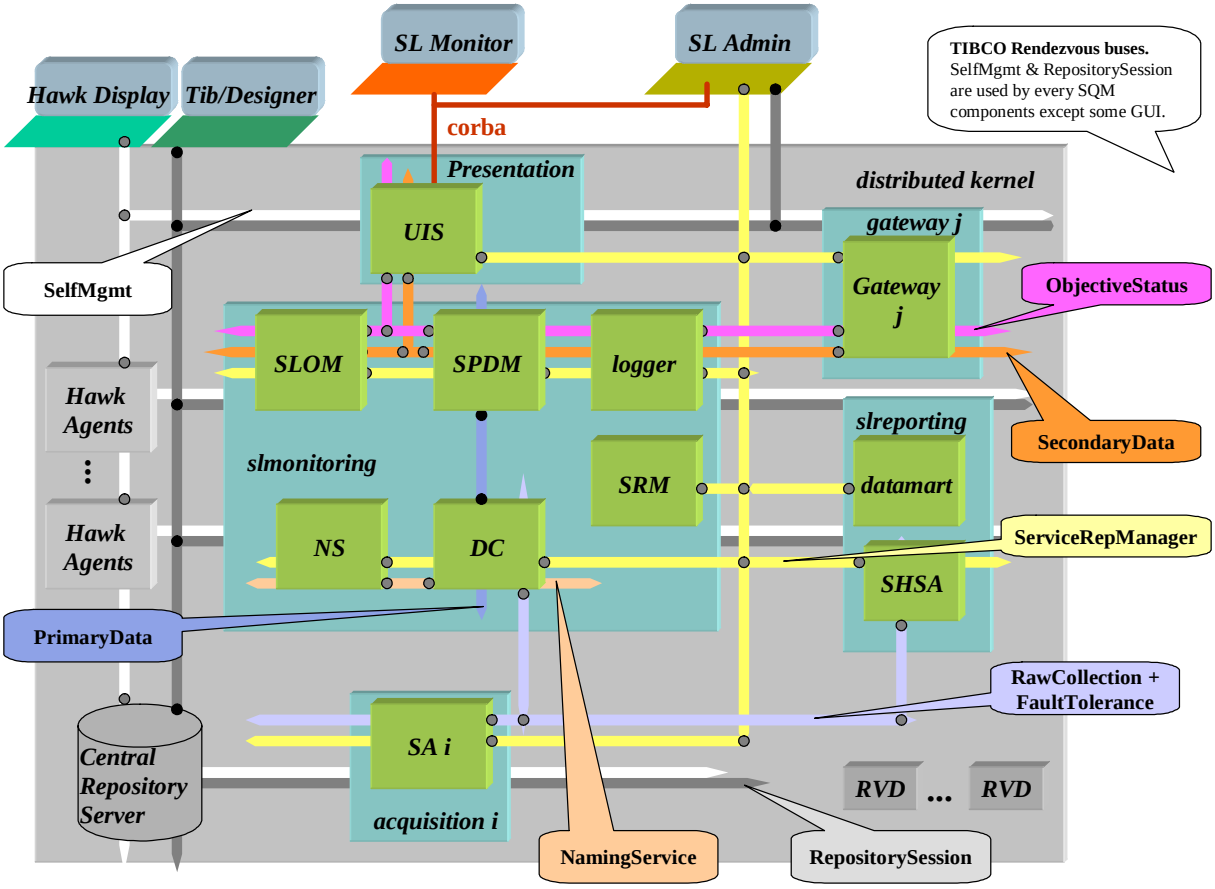
Refer to the TIBCO document "*TIBCO Rendezvous Administration*" for details about the TIBCO Rendezvous daemon **RVRD.**

This must be done on each of the system boxes that host your SQM platform and that must be connected via a **RVRD**. This may concern only some of the system boxes hosting your distributed SQM platform, depending on your network topology and the SQM TIBCO Rendezvous buses that will cross a physical network or sub-network that do not support the multicast or broadcast IP.Figure 22 shows the buses that may have to be configured in neighbors RVRD. In a typical SQM deployment, the sole SQM Directors that may have to run on remote secondary hosts are ***slreporting***, ***acquisition*** X, ***Presentation*** and ***Gateway*** Y.

Important

It is highly recommended to understand the TIBCO Rendezvous routing concepts described in the TIBCO document "*TIBCO Rendezvous Administration*" to avoid topology errors that may result in messages received several times. Except if no multicasting is used between the daemons, all of the RVD running on a distributed SQM platform have not to be systematically replaced by a RVRD.
For instance, in case of multicast (or broadcast), a couple of RVRD has to be configured between sub-networks that do not support multicast IP between them, but only one RVRD must be running on each sub-network. The others Rendezvous daemons on a sub-network will also receive the messages coming from the peer sub-network because they continue to be forwarded via multicast IP to the RVD connected to the same sub-network.

Figure 22: TIBCO Rendezvous buses used by a SQM Platform



2.2.5.2 How to start RVRD

In order to use **RVRD** at the place of **RVD**, your router flag must be true in your SQM environment. (i.e. TEMIP_SC_RVRD_FLAG=TRUE).

You can set this feature, during the setup or after.

During the setup of SQM

If you decide to use a **RVRD** during the setup, you must edit the `platform_desc.cfg` (see section Customizing the setup) before to proceed and set

```
<MessagingDaemons routerFlag="True">
```

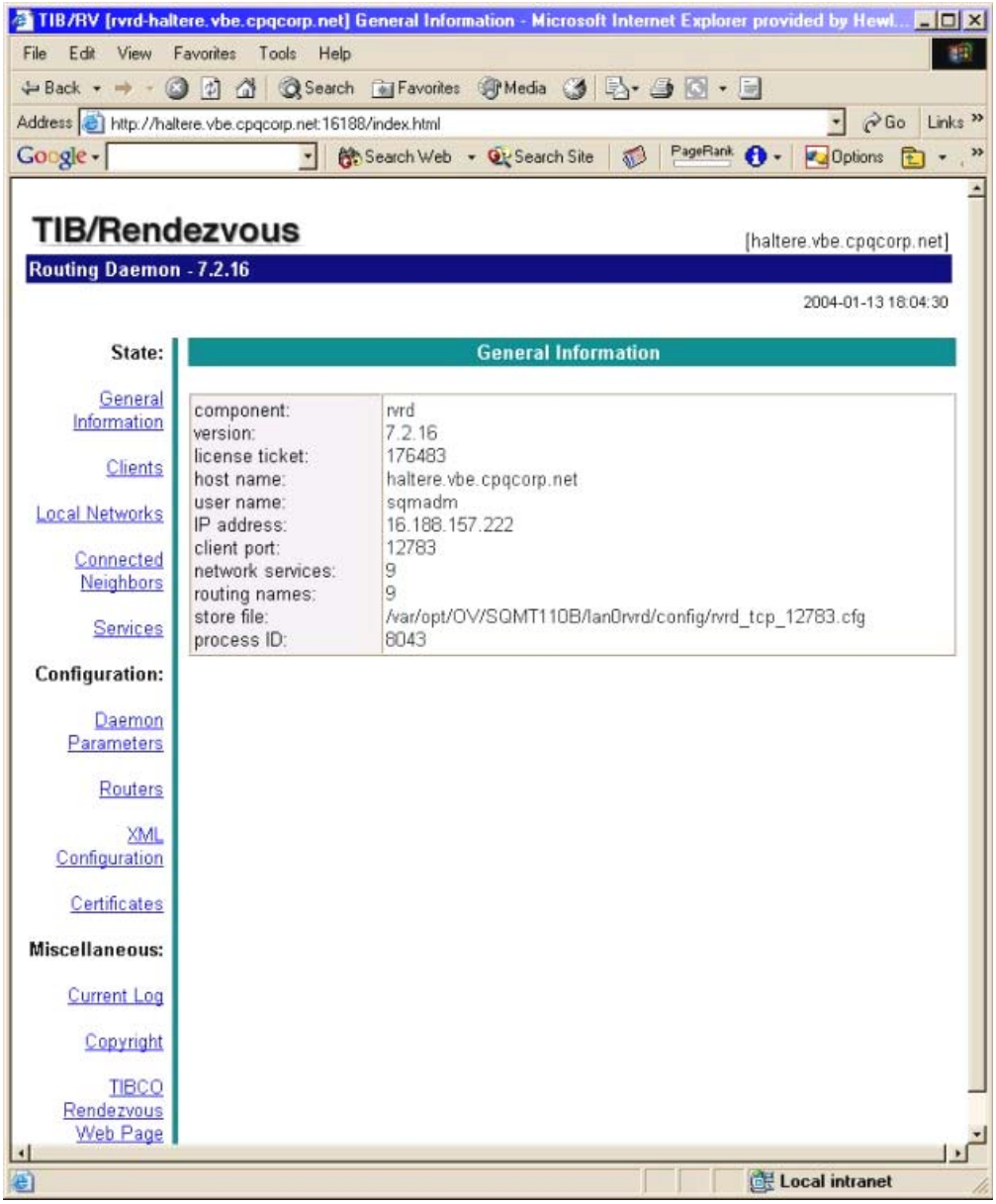
On an already setup SQM

You may also decide to replace the currently used **RVD** by a **RVRD** at any time after the setup. You will have to stop your SQM platform and Kernel (along with the **RVD**), and then you must change to ***TRUE*** the value of ***TEMIP_SC_RVRD_FLAG*** specified in your `$TEMIP_SC_VAR_HOME/temip_sc_env.sh` (or `.bat` on Windows). By default, this flag is ***FALSE*** meaning that the TIBCO Rendezvous daemon used by the SQM Kernel is **RVD**.
Once this flag is set in your environment, you will have to start the RVRD alone to provide time for the daemon to create its configuration file.
This is done by invoking the command *rvd*.
You can then start the SQM Kernel.

2.2.5.3 How to configure two neighbors RVRD

Once the **RVRD** started, a manual configuration has to be done to declare the routes between each couple of **RVRD.** This manual configuration is done through the **RVRD** 's HTTP console (use the same URL as described in section 2.2.4). When **RVRD** is running rather than **RVD**, the HTTP console contains additional information under the *"Configuration"* item. Particularly "*Routers*" and "*Configuration*" that must be used to declare the routes to the neighbor's **RVRD**. See Figure 23: RVRD 's HTTP page.

Figure 23: RVRD 's HTTP page.



Identifying the TIBCO Rendezvous buses

SQM uses several TIBCO Rendezvous buses as shown Figure 22. A route must be declared for each services (i.e. a TIBCO Rendezvous buses) used by SQM, if this service cross a physical network or sub-network that do not support multicast or broadcast IP.

A TIBCO Rendezvous bus is defined by:

- Its UDP port number, called SERVICE.
- And, its broadcast (i.e. ***";"***) or multicast (e.g. ***";224.2.0.7"***) IP address parameter, called NETWORK.

These parameters are specified, for each of the SQM buses, in the Central Repository and in SQM environment file under TEMIP_SC_VAR_HOME:

```
temip_sc_env.[ sh | bat ]
```

For instance, the SQM bus named `RepositorySession` used by every SQM Component to access the Central Repository is defined on UNIX by the environment variables:

$TEMIP_SC_REPOSITORY_SESSION_SERVICE
$TEMIP_SC_REPOSITORY_SESSION_NETWORK

The following table gives the correspondence between the identifier of a SQM TIBCO Rendezvous bus in the user's environment, its identifier displayed in the RVRD' s HTTP console and its name in the `platform_desc.cfg` file.

Global Identifier and name in `platform_desc.cfg`	**Identifier in TIBCO Rendezvous daemon's HTTP console and Central Repository**	**User's environment Variables** Where <*param*> is `NETWORK` and `SERVICE`
`FaultTolerance`	`MonitoringSession or FTsession`	`TEMIP_SC_FT_RV_`<*param*>
`NamingService`	`NS*Session`	`TEMIP_SC_NS_RV_`<*param*>
`PrimaryData`	`RawCollDataSession`	`TEMIP_SC_PRIM_DATA_RV_`<*param*>
`RawCollection`	`RawCollDataSession`	`TEMIP_SC_RAW_COLL_RV_`<*param*>
`RepositorySession`	`none`	`TEMIP_SC_REPOSITORY_`<*param*>
`ServiceRepManager`	`SDSI*Session*`	`TEMIP_SC_SDSI_RV_`<*param*>
`SecondaryData`	`SecondaryDataSession`	`TEMIP_SC_SECOND_DATA_RV_`<*param*>
`SelfMgmt`	`SelfMgmtSession`	`TEMIP_SC_SELF_MGMT_RV_`<*param*>
`ObjectiveStatus`	`SVD*Session`	`TEMIP_SC_SVD_RV_`<*param*>

General principle

On the primary host (i.e. the system that hosts the Central Repository):

Contact the HTTP console of the **RVRD** running on this host.

For each of the SQM TIBCO Rendezvous buses that must be routed, add a Neighbor link that **accepts any** remote connections on a chosen local port. A given route is used by a single SQM TIBCO Rendezvous bus.

At the end, on the primary host's RVRD, you will have configured a TCP port for at least the *`SelfMgmt`*, the *`RepositorySession`* and the *`ServiceRepManager`* buses, and possibly others buses depending on the distribution.

These TCP ports configured on the primary host's **RVRD** must be used by the secondary host's **RVRD** to connect to the primary host's **RVRD** and to transport messages through the given SQM TIBCO Rendezvous bus.

On a secondary host (i.e. a system that does not host the Central Repository):

For each SQM TIBCO Rendezvous buses that must be routed, add a Neighbor link to the corresponding TCP port configured on the primary host's **RVRD** (i.e. a remote TCP port from the secondary host's **RVRD** point of view). Since the primary **RVRD** is a passive listener, secondary **RVRD** must add the route as an **active** entity.

On each host (primary or secondary):

Each **RVRD** must declare the correspondence between a defined neighbor link and its local TIBCO Rendezvous bus (i.e. NETWORK & SERVICE), and declare the Subject that can potentially pass through this route. For SQM use the subject wildcard **">"** in any directions.

Notes

The **RVRD** administration information called "**Router Name**" refer to the routing table entry, it **must be a globally unique identifier**. This unique "Router Name" has to be used to identify a given neighbor.

The **RVRD** administration information called "**Local Network Name**" **must be a globally unique identifier**.

Important

Remember that each SQM Kernel that belongs to a distributed SQM Platform must have strictly the same TIBCO Rendezvous buses configuration. Those are the TIBCO Rendezvous Daemon TCP port, the TIBCO Rendezvous Service and the TIBCO Rendezvous Network parameters defined for the 9 SQM buses: `FaultTolerance, NamingService, PrimaryData, RawCollection, RepositorySession, ServiceRepManager, SecondaryData, SelfMgmt and ObjectiveStatus.`

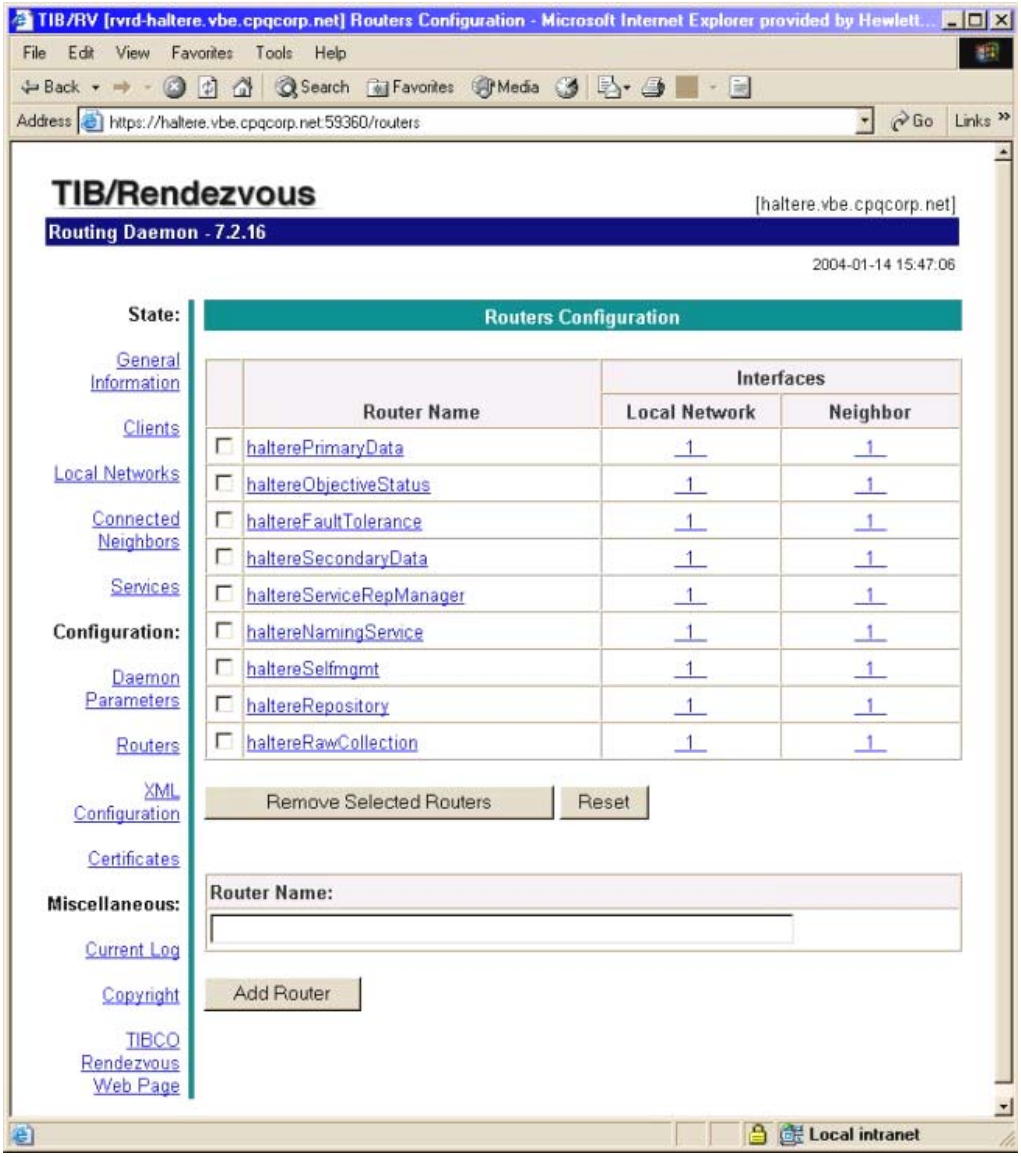
You must always keep the same definition on the primary host and on the secondary hosts for these parameters. This means that network and service defined for `local-network` of the RVRD routes must be the same on every RVRD of the SQM Platform, for a given SQM bus. See `<local-network name="xxx" network=";" service="1615x">` in section 2.2.5.4, the example shows that these parameters are the same on the primary and the secondary hosts. If you do not respect this constraint, the distributed SQM Platform will fail.

Configuring a RVRD routing table

There are two ways to configure the RVRD routing table using the RVRD 's HTTP console. We suggest that you to use the second alternative as it is simpler than the first one, at least at the beginning.

- **First method**: Via the page Figure 24 opened by the hyperlink ***Routers*** of the main page shown by Figure 23. Using this page and its sub-pages you will have to enter configuration route by route. Figure 24 shows the final state once the routes for the 9 SQM TIBCO Rendezvous buses have been configured. At the beginning, the rows are empty. In this example, the names of the routes have been built by concatenating the hostname and the bus name. Sub-pages that must be used are not shown in this document; they are opened by the hyperlink in columns ***Local Network*** and ***Neighbor***. See document "*TIBCO Rendezvous Administration*" for details on how to configure the route.

Figure 24: Router page of RVRD.

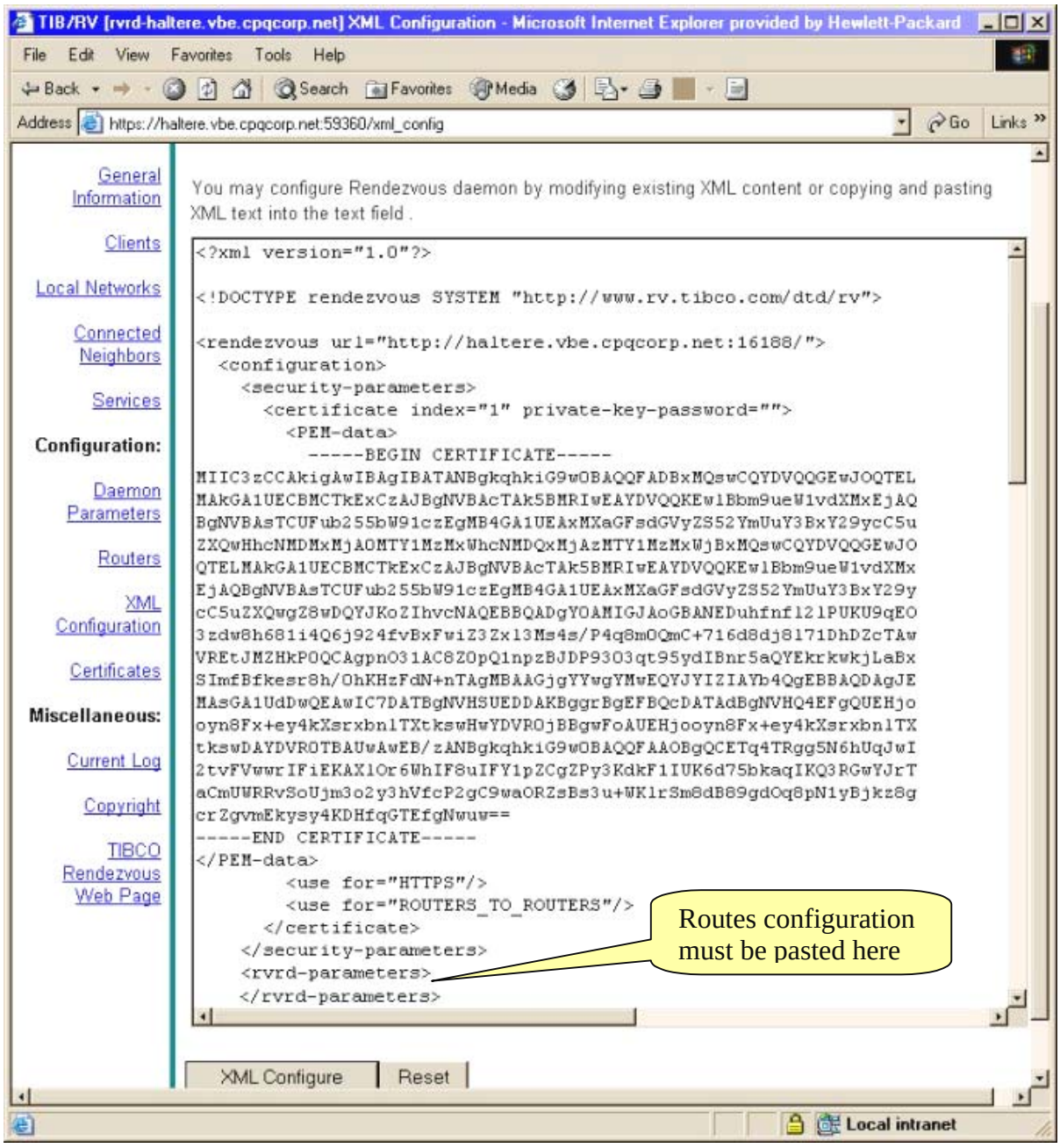


- **Second method**: Via the page Figure 25 opened by the hyperlink ***XML Configuration*** of the main page shown by Figure 23. Using this page, you can enter the XML that represents the configuration of the entire routing table of the RVRD in one step. First you retrieve the initial XML configuration, then you can paste your routes configuration between the XML tags `<rvrd-parameters>` `</rvrd-parameters>`. An example is given in the section Example of RVRD configuration, and a file containing the configuration for all of the 9 SQM TIBCO Rendezvous buses on the primary and on a secondary host is delivered by SQM (see `$TEMIP_SC_HOME/etc/rvrdCfg_example.txt`).

  Attributes which must be set to instantiate this example file to your own configuration are:

  - For each route defined on the primary host side: router name; neighbor local-port; local-network name; local-network network; local-network service;
  - For each route defined on a secondary host side: router name; neighbor local-port; neighbor neighbor-name; neighbor remote-host; neighbor remote-port; local-network name; local-network network; local-network service;

Figure 25: XML Configuration page of RVRD.



2.2.5.4 Example of RVRD configurations

This section shows the XML that must be pasted in the ***XML Configuration*** page of the RVRD 's HTTP consoles on a primary host and on a secondary host called **hanks**.

This example shows the routes that must be configured between the **RVRD** running on the primary host called **haltere** and the **RVRD** running on a secondary host called **hanks**. The secondary host in this example only hosts a **Service Adapter**, therefore the only SQM TIBCO Rendezvous buses that must be routed are `RepositorySession, SelfMgmt, ServiceRepManager, RawCollection` and `FaultTolerance.` Values in bold have to be substituted to instantiate this example to your own configuration.

Notes

- Remember that depending on your distributed architecture, some of the routes may be useless.
- You may use any TCP ports to connect the remote **RVRD**s. To simplify this example a **local Endpoint** uses the same TCP port as its **remote Endpoint** (i.e. 660x) but another choice is possible. Be sure that these TCP ports are not already used.
- The example applies the naming guideline: A route name is the concatenation of the hostname and the SQM bus name.

  This prevents the user to have conflict between the "**Router Name**" defined over the network used by the SQM Platform. Recall that this identifier **must be a globally unique identifier.**

TIBCO Rendezvous buses routed by this example

Configuration of the TIBCO Rendezvous buses routes by this example is shown by the following table. The same configuration is setup on all the system boxes that host the distributed SQM Platform:

Bus identifier	SERVICE (i.e. UDP Port)	NETWORK (bcast or mcast)
FaultTolerance	**16157**	**;**
NamingService	**16156**	**;**
PrimaryData	**16151**	**;**
RawCollection	**16150**	**;**
RepositorySession	**16158**	**;**
ServiceRepManager	**16155**	**;**
SecondaryData	**16152**	**;**
SelfMgmt	**16154**	**;**
ObjectiveStatus	**16153**	**;**

In this example we have choose to use the same TCP port on both host to enable the **Endpoints (remote & local)** which the routes will use (i.e. 660x):

Bus identifier	Endpoints (TCP)
FaultTolerance	**6607**
RawCollection	**6600**
RepositorySession	**6608**
ServiceRepManager	**6605**
SelfMgmt	**6604**

Configuring the RVRD on the primary host

Primary host name is **haltere.**

```
<?xml version="1.0"?>
<!DOCTYPE rendezvous SYSTEM "http://www.rv.tibco.com/dtd/rv">
<rendezvous url="http://haltere.vbe.cpqcorp.net:16188/">
  <configuration>
   <security-parameters>
    <certificate index="1" original-file="N/A" private-key-password="">
        <PEM-data>
          -----BEGIN CERTIFICATE-----
.........
          -----END CERTIFICATE-----
```

```
        </PEM-data>
        <use for="HTTPS"/>
        <use for="ROUTERS_TO_ROUTERS"/>
    </certificate>
   </security-parameters>

  <rvrd-parameters>
      <router name="haltereFaultTolerance" maximum-backlog="0">
        <neighbor type="ACCEPT-ANY" local-port="6607" cost="1"
                  compressed="no"/>
        <local-network name="FaultTolerance" network=";"
                  service="16157" cost="1">
          <import subject=">" weight="10"/>
          <export subject=">"/>
        </local-network>
      </router>
      <router name="haltereServiceRepManager" maximum-backlog="0">
        <neighbor type="ACCEPT-ANY" local-port="6605" cost="1"
        compressed="no"/>
        <local-network name="ServiceRepManager" network=";"
                  service="16155" cost="1">
          <import subject=">" weight="10"/>
          <export subject=">"/>
        </local-network>
      </router>
      <router name="haltereSelfmgmt" maximum-backlog="0">
        <neighbor type="ACCEPT-ANY" local-port="6604" cost="1"
                  compressed="no"/>
        <local-network name="selfmgmt" network=";" service="16154"
                  cost="1">
          <import subject=">" weight="10"/>
          <export subject=">"/>
        </local-network>
      </router>
      <router name="haltereRepository" maximum-backlog="0">
       <neighbor type="ACCEPT-ANY" local-port="6608" cost="1"
                 compressed="no"/>
        <local-network name="Repository" network=";" service="16158"
                  cost="1">
          <import subject=">" weight="10"/>
          <export subject=">"/>
        </local-network>
      </router>
      <router name="haltereRawCollection" maximum-backlog="0">
        <neighbor type="ACCEPT-ANY" local-port="6600" cost="1"
                  compressed="no"/>
        <local-network name="RawCollection" network=";" service="16150"
                  cost="1">
          <import subject=">" weight="10"/>
          <export subject=">"/>
        </local-network>
      </router>
</rvrd-parameters>

</configuration>
</rendezvous>
```

Configuring the RVRD on the secondary host

Secondary host name is **hanks.**

```
<?xml version="1.0"?>
<!DOCTYPE rendezvous SYSTEM "http://www.rv.tibco.com/dtd/rv">

<rendezvous url="http://hanks.vbe.cpqcorp.net:16188/">
  <configuration>
   <security-parameters>
     <certificate index="1" original-file="N/A" private-key-password="">
        <PEM-data>
          -----BEGIN CERTIFICATE-----
          .........
         -----END CERTIFICATE-----
       </PEM-data>
        <use for="HTTPS"/>
        <use for="ROUTERS_TO_ROUTERS"/>
     </certificate>
   </security-parameters>

   <rvrd-parameters>
      <router name="hanksRepository" maximum-backlog="0">
        <neighbor type="ACTIVE" local-port="6608"
              neighbor-name="haltereRepository"
              remote-host="haltere.vbe.cpqcorp.net"
              remote-port="6608" cost="1"
              compressed="no" encrypted="no"/>
        <local-network name="Repository1" network=";" service="16158"
              cost="1">
          <import subject=">" weight="10"/>
          <export subject=">"/>
        </local-network>
      </router>
      <router name="hanksSelfmgmt" maximum-backlog="0">
        <neighbor type="ACTIVE" local-port="6604"
              neighbor-name="haltereSelfmgmt"
              remote-host="haltere.vbe.cpqcorp.net"
              remote-port="6604" cost="1"
              compressed="no" encrypted="no"/>
        <local-network name="Selfmgmt1" network=";" service="16154"
              cost="1">
          <import subject=">" weight="10"/>
          <export subject=">"/>
        </local-network>
      </router>
      <router name="hanksRawCollection" maximum-backlog="0">
        <neighbor type="ACTIVE" local-port="6600"
              neighbor-name="RawCollection"
              remote-host="haltere.vbe.cpqcorp.net" remote-port="6600"
              cost="1"
              compressed="no" encrypted="no"/>
        <local-network name="RawCollection1" network=";" service="16150"
              cost="1">
          <import subject=">" weight="10"/>
          <export subject=">"/>
        </local-network>
      </router>
      <router name="hanksFaultTolerance" maximum-backlog="0">
        <neighbor type="ACTIVE" local-port="6607"
```

```
                  neighbor-name="haltereFaultTolerance"
                  remote-host="haltere.vbe.cpqcorp.net"
                  remote-port="6607"
                  cost="1" compressed="no" encrypted="no"/>
        <local-network name="FaultTolerance1" network=";"
                  service="16157" cost="1">
          <import subject=">" weight="10"/>
          <export subject=">"/>
        </local-network>
      </router>
      <router name="hanksServiceRepManager" maximum-backlog="0">
        <neighbor type="ACTIVE" local-port="6605"
                  neighbor-name="haltereServiceRepManager"
                  remote-host="haltere.vbe.cpqcorp.net"
                  remote-port="6605"
                  cost="1" compressed="no" encrypted="no"/>
        <local-network name="ServiceRepManager1" network=";"
                  service="16155"
                  cost="1">
          <import subject=">" weight="10"/>
          <export subject=">"/>
        </local-network>
      </router>
    </rvrd-parameters>

  </configuration>
</rendezvous>
```

Checking the RVRD configuration

Once the RVRD are configured, you may check that routes are correctly defined by testing if messages can be exchanged between primary and secondary hosts. For each of the concerned SQM buses, you can start a tibrvlisten on one side to verify that it receives messages sent by a tibrvsend on the other side. See chapter How to detect if network support multicast or broadcast IP for TIBCO Rendezvous that describes how to use these two commands.

2.2.6 Self Management command line utility

Openview SQM provides a command line utility called ***temip_sc_selfmgmt*** that can be used to invoke Self-Management methods on TIBCO Hawk micro agents. Recall that these methods can also be invoked using the SQM Admin console (see section Monitoring OpenView SQM using the Admin Console). ***temip_sc_selfmgmt*** can also list all micro-agents currently discovered by the given TIBCO Hawk agent (see option –list).

The command will exit with an exit code of 0 if no errors occurred during the execution of the command; otherwise, the exit code of the command is greater than zero.

Command line syntax

```
temip_sc_selfmgmt  [-help | -h]
                   {-hawk_agent|-ha} <agent_name >
                   {-hawk_microagent|-hma} <microAgentName>
                   [{-output|-o} <filename>
                   [{-list|-l}] [{listmethods | lm}]
                   [-timeout <millisecond>]
                   [-method <method_name>]
                   [-args <arg=val> ... ]
```

Options

-help Displays help information about the command.

-ha <*agent_name*>	Indicates the name of the TIBCO agent on which micro-agents are defined.
-hma <*microAgentName*>	Indicates the name of the TIBCO micro-agent on which methods are invoked. The value is generally <*hostname*>-HA.
-o <*filename*>	Indicates the output file where the result of the command will be written. Only the result of the command will be written to the file. Errors will be printed on the standard output. The file name must be an absolute path.
-list	With this option, temip_sc_selfmgmt returns the list all micro-agents currently discovered by the given TIBCO Hawk agent.
-timeout <*millisecond*>	Indicates the delay in milli-seconds used to discover agents and micro agent.
-method <method_name>	Indicates the name of the method that will be invoked on the given micro agent belonging to the given TIBCO Hawk agent.
-args <arg1=val1> ...<argI=valI>	Indicate arguments of the message to send. This option must be the last option of the command.
-listmethods	Displays the list of methods available under a specified TIBCO Hawk microagent
-describe <*methodName*>	Gives the description of the TIBCO Hawk micro agent method.

Examples

The following examples show how to use the tool with different combinations of options.

- temip_sc_selfmgmt -ha "haltere.vbe.cpqcorp.net-HA" –l

*** List of all Micro Agent linked to the Agent : haltere.vbe.cpqcorp.net-HA*
Micro Agent Name : COM.TIBCO.hawk.hma.FileSystem
Micro Agent Name : COM.TIBCO.hawk.microagent.SysInfo
Micro Agent Name : COM.TIBCO.hawk.microagent.Custom
Micro Agent Name : COM.TIBCO.hawk.microagent.Logfile
Micro Agent Name : COM.TIBCO.hawk.hma.FileStat
Micro Agent Name : COM.TIBCO.hawk.hma.Process
Micro Agent Name : COM.TIBCO.hawk.hma.System
Micro Agent Name : COM.TIBCO.hawk.hma.Network
Micro Agent Name : COM.TIBCO.hawk.microagent.RuleBaseEngine
Micro Agent Name : COM.TIBCO.hawk.hma.TibRendezvous
Micro Agent Name : COM.TIBCO.hawk.microagent.Self
Micro Agent Name : COM.TIBCO.REPOSITORYSERVER
Micro Agent Name : COM.TIBCO.hawk.microagent.HawkEventService
Micro Agent Name : COM.TIBCO.ADAPTER.SRM
Micro Agent Name : slmv12_slmonitoring_SRM_MA
Micro Agent Name : COM.TIBCO.ADAPTER.SPDM
Micro Agent Name : slmv12_slmonitoring_SPDM_MA
Micro Agent Name : COM.TIBCO.ADAPTER.SLOM
Micro Agent Name : slmv12_slmonitoring_SLOM_MA
Micro Agent Name : COM.TIBCO.ADAPTER.NS
Micro Agent Name : slmv12_slmonitoring_NS_MA
Micro Agent Name : COM.TIBCO.ADAPTER.DC
Micro Agent Name : slmv12_slmonitoring_DC_MA
Micro Agent Name : COM.TIBCO.ADAPTER
Micro Agent Name : COM.TIBCO.ADAPTER.logger
Micro Agent Name : slmv12_slmonitoring_Logger_MA
Micro Agent Name : COM.TIBCO.ADAPTER.UIS

Micro Agent Name : slmv12_presentation_UIS_MA

- temip_sc_selfmgmt -ha -hma slmv12_slmonitoring_DC_MA -m getTraceLogLevel

  *** Result of the method invocation:*
  composite {traceLevel=OFF}

- temip_sc_selfmgmt -ha agent1 -hma micro1 -m compute -a f1=2 f2=3 f3=4

2.2.7 Advanced setup procedures

This section describes SQM advanced setup procedures of various platforms that can be used to setup a non-typical SQM Platform.

Important

Normal setup of SQM must be done as described in the document *OpenView SQM Installation guide.* This chapter is reserved for advanced setup operations and should not normally be used.

The general principle that governs the advanced setup is to design the whole distributed SQM Platform into a file called platform description file.

This design describes:

- On one hand, the SQM Kernel parameters that must be used on each of the system boxes that will host a part of the distributed SQM Platform. These SQM Kernel parameters must be the same on all the hosts.
- On the other hand, the structure and the distribution of the SQM Directors and Applications which compose the SQM Platform to be setup. The distribution is defined host by host, given its real hostname or possibly using the keyword `localhost`, and for each host the SQM directors and applications that must be setup on it.

Once, the distributed SQM Platform has been completely described in the platform description file. It must be deployed and executed by the command `temip_sc_setup,` on each of the system boxes designed to host a part of the distributed SQM Platform. The deployment must always begin with the SQM primary host (i.e. the system that hosts the Central Repository), which must be a UNIX.

Note

OpenView SQM is delivered with a default platform description file that contains the description of the typical SQM Platform deployment. It has not to be manually edited for normal SQM setup.

Default SQM platform description file is
`$TEMIP_SC_HOME/etc/platform_desc.cfg`

2.2.7.1 SQM Setup Command Line Tool

The advanced setup procedures are based on a command `temip_sc_setup` and its main input file, generally named platform_desc.cfg, where the whole distributed SQM platform is described (see section 2.2.7.2).

Note: Location of `temip_sc_setup` is under $TEMIP_SC_HOME/setup/bin.

Command line syntax

```
temip_sc_setup [-NI]
                -all [-cfg <platformDescriptorFile>]
                | -update <platformDescriptorFile>
                        [-addDirector <directorName>]
                | -addOn [<addOnLabel> -appliName
                         <appliName> [-dirName <dirName>]]
```

```
[-force]
```

Options

The temip_sc_setup command is either a “temip_sc_setup –all” or a “temip_sc_setup –update” or a “temip_sc_setup –addOn” and has the following list of valid options:

-NI	non-interactive mode
-force	to overwrite the existing TEMIP_SC_VAR_HOME directories
-all	performs all tasks Kernel setup, kernel start, create platform, directors, applications mentioned in the platform descriptor file. By default the platform_desc.cfg that will be used by command temip_sc_setup –all is `$TEMIP_SC_HOME/tmp/platform_desc.cfg`
-update <*descFilePathname*>	to create applications designed to be hosted on the local host, in the platform descriptor. Option update do not setup the SQM Kernel and must only be used to enhanced an already setup SQM Kernel.
-addDirector <*directorName*>	to create on the local host, a director and its application given the director name. With the option addDirector the hostname matching rule is not applied to check if the director must be installed on the actual local host or not. The director will be setup anyway on the host where the setup –addDirector has been invoked. The only constraint is: The hostname declared in the platform descriptor for that director must be the keyword *ForFurtherUpdate*. Option –addDirector <*directorName*> requires to set also the option –update <*descFilePathname*>. This option may be used for example to create an additional instance of the SQM director *presentation*, possibly on a secondary host.
-addOn <*addOnFilename*>	to add an addOn compatible application (e.g. a Service Adaptor).
-appliName <*appliName*>	to specify the name of the application to be add-on
-dirName <*dirName*>	to specify the director name of the application to be add-on
	With the option -addOn the hostname matching rule is not applied to check if the director must be installed on the actual local host or not. The director will be setup anyway on the host where the setup –addOn has been invoked.

Note

All the platform description filenames must be specified with their full pathname.

Examples

temip_sc_setup -all

For interactive mode of kernel setup and non interactive mode of others components

temip_sc_setup -all -NI

Non interactive mode

temip_sc_setup -all -NI -force

To overwrite the existing TEMIP_SC_VAR_HOME

temip_sc_setup -update <platform_desc file pathname>

To create, on a running platform, additional directors and their applications, which have host value matching the local host name.

temip_sc_setup -update <*platformDescPathname*> -addDirector <*DirectorName*>

To create an additional director named <*DirectorName*> and its applications on a running SQM platform. The host value specified in the platform descriptor for this director must be “ForFurtherUpdate". Otherwise, creation is refused with ERROR: Local hostname doesn't match.

temip_sc_setup -addOn <*addOn filename*> -appliName <*ApplicationName*>

To create, on a running platform, an additional Service Adapter ort Gateway named <*ApplicationName*>.The director and application characteristics are described in the specified addOn file that should be located in the $TEMIP_SC_HOME/etc/addOn directory.

temip_sc_setup -addOn

To create interactively, a set of addOn compatible application. Note: an application is "addOn compatible" if a corresponding addOn file exists in the $TEMIP_SC_HOME/etc/addOn directory.

2.2.7.2 Customizing the setup

In advanced setup mode, you can edit manually the platform_desc.cfg file that will be used by the setup. By default the platform_desc.cfg that will be used by command temip_sc_setup is `$TEMIP_SC_HOME/tmp/platform_desc.cfg`.

The platform description file is actually an XML file. Refer to its DTD (`$TEMIP_SC_HOME/etc/platform_desc.dtd`) to see data that can be edited in this file before the deployment.

Example: the Multicast IP Address used by the SQM Kernel

To change the broadcast IP network to multicast IP network parameters, change all the network session parameters to “;*224.2.xx.yy*”.

Note

These changes must be carefully done. Refer to section How to detect if network support multicast or broadcast IP for TIBCO Rendezvous before to proceed.

The platform description file must contain the following data for the definition of the TIBCO Rendezvous buses used by your SQM Platform:

```
<ConfigServerRepository>
  <RepositorySession network=";224.2.0.1" service="6608"/>
  <RepositoryInstance name="screpos"/>
</ConfigServerRepository>

<SLMMessageBuses>
  <SelfMgmt network=";224.2.0.1" service="6604"/>
  <RawCollection network=";224.2.0.1" service="6600"/>
  <PrimaryData network=";224.2.0.1" service="6601"/>
  <SecondaryData network=";224.2.0.1" service="6602"/>
  <ServiceRepManager network=";224.2.0.1" service="6603"/>
  <ObjectiveStatus network=";224.2.0.1" service="6605"/>
  <NamingService network=";224.2.0.1" service="6606"/>
  <FaultTolerance network=";224.2.0.1" service="6607"/>
</SLMMessageBuses>
```

2.2.8 Advanced utilities to modify the structure of a SQM platform

Important

SQM Administrators normally have not to modify the structure and distribution of the SQM platform. This operation must be done very carefully. If one of the mandatory directors is not well redeployed, run of SQM will fail.

Require a running SQM Kernel

All the commands and operations described in this chapter require that a SQM Kernel is running on each host concerned by the operation. If you need to add a new host in the distributed SQM platform, you have first to setup the SQM Kernel, on this new host, with the same SQM platform parameters than the already installed SQM Kernels.

Refer to paragraph Advanced setup procedures and document *OpenView SQM Installation Guide* to see how to setup a new host. Recall that the SQM platform parameters are registered into the `platform_desc.cfg` file used to setup SQM, on a given host. You must copy the primary `platform_desc.cfg` under `$TEMIP_SC_HOME/tmp` on the new host you want to setup. See also the section Advanced setup procedures.

2.2.8.1 Redeploying SQM directors

Adding a new Director

The name of the new director must be different from the other existing directors on the platform in order to avoid conflicts.

On an existing host

To create a director on an existing host, invoke the command temip_sc_setup -update (see section 2.2.7), it creates the director and its applications.

On a new host

This is the same procedure as for adding a Director on an existing host, but you need to install the SQM Kernel before adding a director on the new host.

Connecting the new Director to the existing platform

This is done by using the `temip_sc_start_director` command.

Monitoring the new Director

You can monitor the application of the new director with the **TIBCO Hawk Display**, as described in section Monitoring OpenView SQM using the Admin Console.

Note

A delay used by the agent discovery is necessary before being able to view the new director with the **TIBCO Hawk Display** and you need to do a *Refresh*.

Moving a Director

The name of the new director must be different from the other existing directors on the platform in order to avoid conflicts.

To an existing host

To move a director to an existing host, you need to use the following scripts and perform the following actions:

- temip_sc_setup –update (see section 2.2.7), to create the director and its applications
- *temip_sc_stop_director* on the source host (see section Stopping an OpenView SQM Director)
- Copy ledgers files from source host to destination host and rename the files
- *temip_sc_start_director* on the destination host (see section Starting an OpenView SQM director)
- *temip_sc_delete_director* on the source host (see section Delete an SQM director)

To a new host

Follow the same steps as for moving a Director to an existing host, but you need to install the kernel before moving the director to the new host.

Connecting the new Director to the existing platform

This is done by using the *temip_sc_start_director* command (see section Starting an OpenView SQM director).

Monitoring the moved Director

You can monitor the application of the new director with the **TIBCO Hawk Display**, as described in section Monitoring OpenView SQM using the Admin Console.

Note

A few minutes are necessary before being able to view the new director with the **TIBCO Hawk Display** and you need to do a *Refresh*.

Deleting a Director

To delete a director, use *temip_sc_delete_director* command (see section Delete an SQM director). All the applications included in this director will also be deleted.

2.2.8.2 Platform and components instantiation commands

This chapter is intended to list and describe the different advanced administration utilities that are provided with OpenView SQM to manage the deployment of the SQM platform. These utilities are located in the $TEMIP_SC_HOME/bin directory.

They have not to be invoked during normal administration of a SQM platform.

Important

SQM Administrators normally have not to modify the structure and distribution of the SQM platform. In any case, this operation must be done very carefully. If one of the mandatory directors is not well redeployed, run of SQM will fail.

Create an SQM platform

An SQM Platform is normally created by the command temip_sc_setup and the relevant platform description file (see section 2.2.7). This temip_sc_setup invoke the basic command **temip_sc_create_platform** that creates the described OpenView SQM platform.

Delete an SQM platform

The **temip_sc_delete_platform** command deletes an SQM Platform from the SQM Central Repository.

Command line syntax

```
temip_sc_delete_platform –platform <platform> [-force]
```

Parameters description

-force	:	forces the creation of the platform even if it exists and is running
-platform <platform>	:	the platform to create
-help	:	displays the command line syntax

Create an SQM director

An OpenView SQM director is normally created by the command temip_sc_setup and the relevant platform description file (see section 2.2.7). temip_sc_setup invokes the basic command **temip_sc_create_director** that creates the SQM Directors declared in the described SQM platform.

Delete an SQM director

The **temip_sc_delete_director** tool deletes an SQM director from the SQM Central Repository in the given OpenView SQM platform.

Command line syntax

```
temip_sc_delete_director -platform <platform> -director
<director> [-force]
```

Parameters description

-force	:	force the deletion of the director even if it exists and is running
-platform <platform>	:	the platform to which the director belongs
-director <director>	:	the director to delete
-help	:	displays the command line syntax

Create an SQM application

An SQM application is normally created by the command temip_sc_setup and the relevant platform description file (see section 2.2.7). This temip_sc_setup invoke the basic command **temip_sc_create_application** that creates the SQM applications declared in the described SQM platform.

Delete an SQM application

The **temip_sc_delete_application** tool deletes an SQM application from the SQM Central Repository in the given OpenView SQM director and platform.

Command line syntax

```
temip_sc_delete_aplication -platform <platform> -director
<director> -application <application> [-force]
```

Parameters description

-force	:	forces the deletion of the application even if it exists and is running
-platform <platform>	:	the platform to which the application belongs
-director <director>	:	the director to which the application belongs
-application < application >	:	application that must be deleted

-help : displays the command line syntax

Chapter 3

Configuring SQM components

The SQM components store and retrieve their configuration into the Central Repository provided by the SQM Kernel. This chapter shows how to edit this configuration using the TIBCO Designer user interface. It also describes the configuration items defined for each SQM components. Elements of configuration common to all SQM components are declared in the Global Variables handled by the Central Repository. They are described in paragraph Global Variables.

3.1 How to Edit Central Repository

The Central Repository is a unique repository, located on only one host (called primary host) of the SQM platform. It contains all information about the SQM platform configuration and the configuration of each of the SQM Components.

The SQM configuration can be edited using the **TIBCO Designer** GUI (for more information about TIBCO Designer, see *"TIBCO Designer User's Guide"* document).

3.1.1 Using the TIBCO Designer

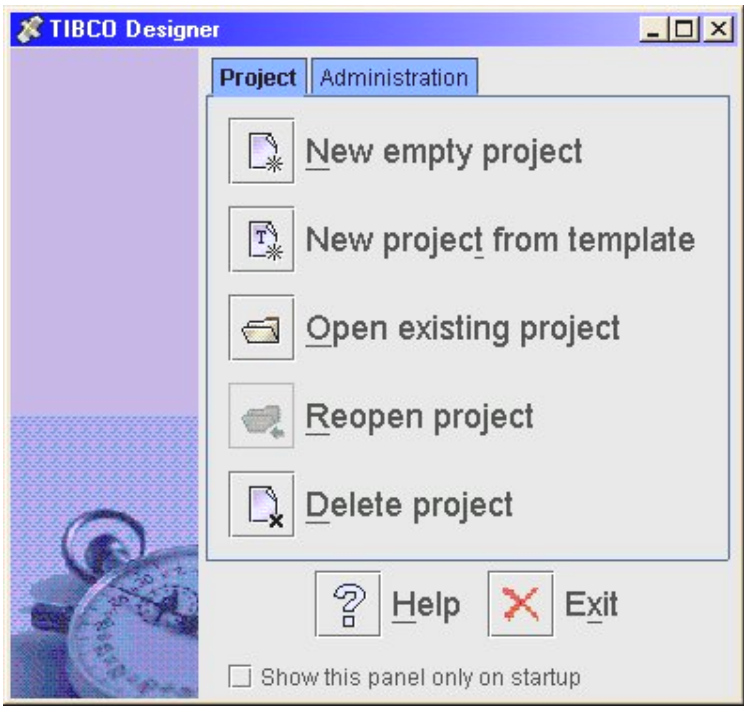
This Graphical User Interface can be used either on:

- A Windows where it has been installed. To start a TIBCO Designer on Windows, users may use the shortcut created during the SQM GUI installation, under the program group "HP OpenView -> SQM".
- Or on a UNIX system where the SQM Kernel has been installed. To start an OpenView SQM Admin Console on UNIX, users have to invoke the command **`designer`**.

The Figure 1 of chapter 1.4 illustrates the position of the Central Repository within the OpenView SQM architecture.

When you start the **TIBCO Designer**, the following startup window appears:

Figure 26: Opening TIBCO Designer



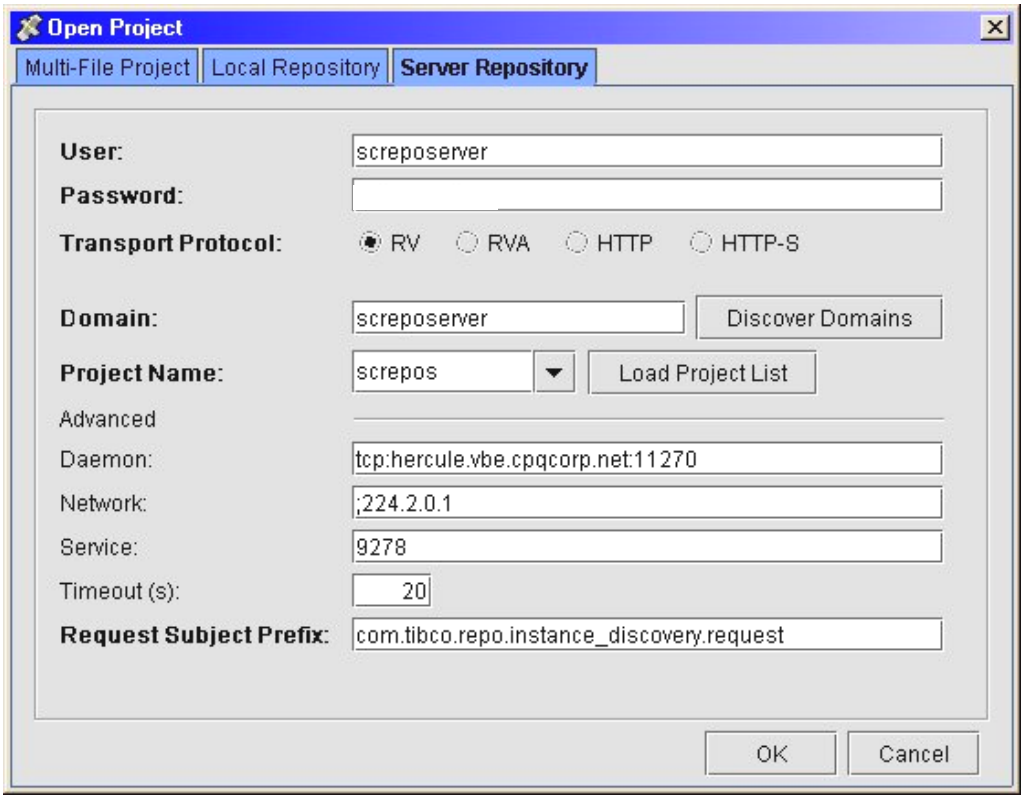
Use the **Open existing project** button to connect the **TIBCO Designer** to the **Central Repository**.

The parameters used to connect the **TIBCO Designer** to the **Central Repository** can be found into the *TEMIP_SC_VAR_HOME/temip_sc_env.sh* file (see following example).

Example: Variables defined in the temip_sc_env.sh file and used by the TIBCO Designer

```
#
# TIBCO repository instance location
#
TEMIP_SC_REPOSITORY_LOCATION='tibcr@screpos:subject=com.tibco
.repo.instance_discovery.request:service=9278:network=;224.2.
0.1:daemon=tcp:11270:userName=scRepoAdmin'
export TEMIP_SC_REPOSITORY_LOCATION
#
# UDP port used for the repository bus
#
TEMIP_SC_REPOSITORY_SESSION_SERVICE='9278'
export TEMIP_SC_REPOSITORY_SESSION_SERVICE
#
# network parameter for the repository bus
#
TEMIP_SC_REPOSITORY_SESSION_NETWORK=';224.2.0.1'
export TEMIP_SC_REPOSITORY_SESSION_NETWORK
#
# TCP port used by Tibco RVD daemon
#
TEMIP_SC_REPOSITORY_SESSION_DAEMON='tcp:hercule.vbe.cpqcorp.n
et:11270'
export TEMIP_SC_REPOSITORY_SESSION_DAEMON
#
# name of the repository instance
#
TEMIP_SC_REPOSITORY_INSTANCE=screpos
export TEMIP SC REPOSITORY INSTANCE
```

Figure 27: Connecting TIBCO Designer to the Repository Server



When you open a project, you can configure parameters to connect to the Repository Server. To get a window such as the one in Figure 27, you have to follow these steps:

- Click on Server Repository tab
- Enter your parameters Daemon, Network and Service, to connect to the Repository Server. Transport protocol is RV.
- Click on the Discover Domains button
- Select the screposerver in the Domain combo box
- Click on the Load Project List button
- Select the screpos project in the Project Name combo box
- Click OK

Daemon, **Network** and **Service** values define a Repository Session.

The **Central Repository Server** is launched at kernel startup. It loads the *screpos.dat* file in memory. If this file is modified while the platform is running, the modifications are only taken into account at the next platform and **Central Repository** restart.

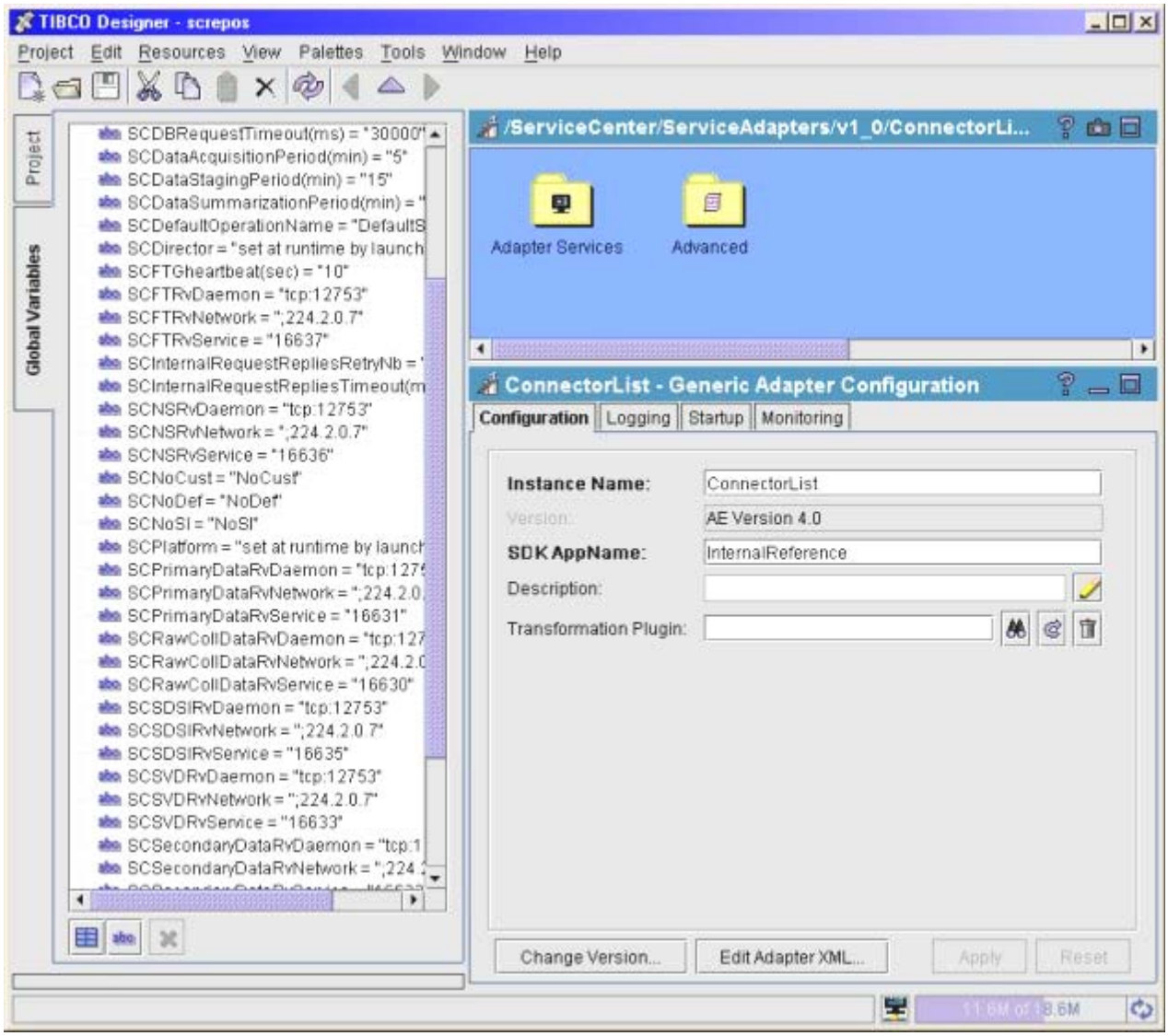
Each application is connected to the **Central Repository** at startup, and thus gets the information contained in the *screpos.dat* file. The **TIBCO Designer** also makes a local copy of the *screpos.dat* file when it connects to the **Central Repository**.

Modifications to parameter values with the TIBCO Designer will be taken into account by an application in case of "**reloadConfig**" or at the next application restart.

3.1.2 How to edit Global Variables

TIBCO Designer can be used to visualize and modify Global Variables. Click on tab “Global Variables” on left side of the main window (see in Figure 28).

Figure 28: TIBCO Designer “Global Variables” Submenu



Note

You can use this window to visualize the Global Variables. To modify the value of a Global Variable click with the right mouse button on the variable you want to modify and select “**Edit Selected Item**”. The modification of a Global Variable will be taken into account by all the components using this Global Variable (value set as %%globalvariablename%%).

Important

To avoid unpredictable results, do not use the *Insert* or *Delete* to create/delete Global Variables.

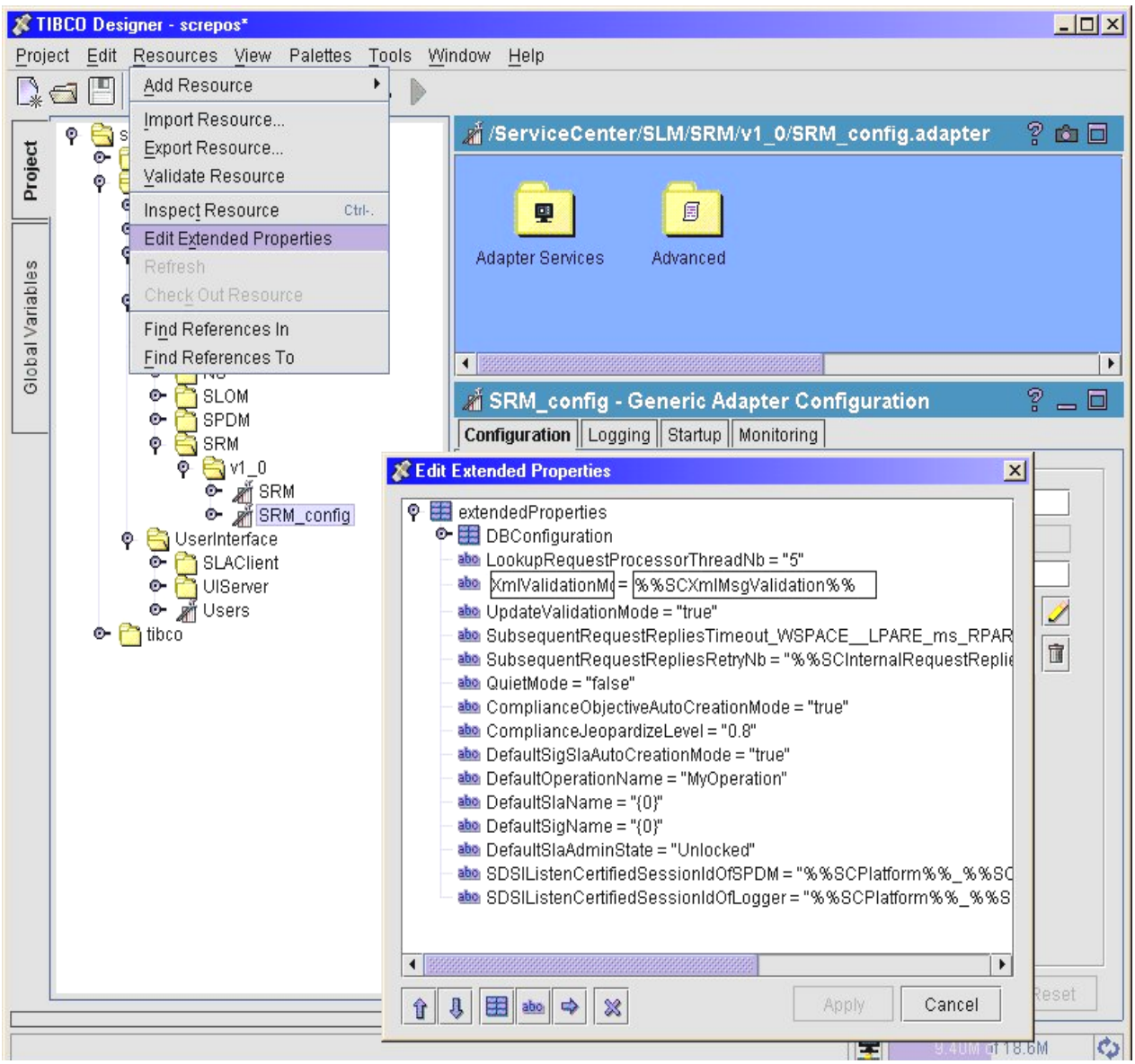
3.1.3 How to edit SQM Components configuration

All Components of the SQM platform are displayed in the **TIBCO Designer**. The **TIBCO Designer** allows visualization and modification of parameters of the SQM Component in the **`<ComponenName>_config`** part. Parameters are called "Extended Properties" and are edited in an "*Edit Extended Properties*" windows (see an example in Figure 29):

- Select tab "*Project*" on the left side of the main windows of TIBCO Designer.
- Select the **`<ComponenName>_config`** node under *screpos*. For instance *SRM_config* is under *screpos -> ServiceCenter -> SLM -> SRM -> v1_0*
- Select menu "***Resources***" -> "***Edit Extended Properties***"

These "***Extended Properties***" represent parameters that are specific to each of the SQM component. For example, the parameters for *SRM* are shown in Figure 29.

Figure 29: SRM_config display using the TIBCO Designer



3.2 Global Variables

Global variables are configuration variables shared by SQM components. They have been defined in order to avoid modifying the same kind of information for each component.

Table 1 lists all the global variables that are used by SQM components. Other global variables visible in the TIBCO Designer tool (see section How to edit Global Variables) are useless and should not be changed.

Table 1: List of Global Variables

Variable Name	Default	Description
SCDBRequestTimeout(ms)	30000	Timeout for all requests towards Database
SCDataAcquisitionPeriod(min)	5	Acquisition polling period for Service Adapters to get data from the product/hardware they are connected to
SCDataStagingPeriod(min)	15	Staging period for Datamart
SCDataSummarizationPeriod(min)	60	Summarization period for Datamart
SCDefaultOperationName	DefaultSLA	Name of the the default operation name that is automatically created when creating a service definition
SCFTGheartbeat(sec)	10	Heartbeat to detect the existence of an application inside a Fault Tolerance Group
SCFTRvDaemon	Value depends on the SQM setup	RV Daemon for Tolerance Group. For instance: `tcp:11240`
SCFTRvNetwork	Value depends on the SQM setup	Network for Tolerance Group. For instance: `;`
SCFTRvService	9247	Service for Tolerance Group. For instance: `9247`
SCInternalRequestRepliesRetryNb	5	Number of retries when an internal requests times out
SCInternalRequestRepliesTimeout(ms)	60000	Internal requests timeout (Internal requests are requests between SQM components)
SCNSRvDaemon	Value depends on the SQM setup	Naming Service bus RV daemon. For instance: `tcp:11240`
SCNSRvNetwork	Value depends on the SQM setup	Naming Service session network. For instance: `;`
SCNSRvService	Value depends on the SQM setup	Naming Service session service. For instance: `9246`
SCPrimaryDataRvDaemon	Value depends on the SQM setup	Primary bus RV daemon. For instance: `tcp:11240`
SCPrimaryDataRvNetwork	Value depends on the SQM setup	Primary bus network. For instance: `;`
SCPrimaryDataRvService	Value depends on the SQM setup	Primary bus service. For instance: `9241`
SCRawCollDataRvDaemon	Value depends on the SQM setup	Raw Collected Data bus RV daemon. For instance: `tcp:11240`
SCRawCollDataRvNetwork	Value depends on the SQM setup	Raw Collected Data bus Network. For instance: `;`
SCRawCollDataRvService	Value depends on the SQM setup	Raw Collected Data bus service. For instance: `9240`
SCRepoRvDaemon	Value depends on the SQM setup	TIBCO Repository bus RV daemon. For instance: `tcp:11240`
SCRepoRvNetwork	Value depends on the SQM setup	TIBCO Repository bus Network. For instance: `;`
SCRepoRvService	Value depends on the SQM setup	TIBCO Repository bus service. For instance: `9248`
SCSDSIRvDaemon	Value depends on the SQM setup	SDSI bus RV daemon. For instance: `tcp:11240`
SCSDSIRvNetwork	Value depends on the SQM setup	SDSI bus network. For instance: `;`
SCSDSIRvService	Value depends on the SQM setup	SDSI bus service. For instance: `9245`

SCSVDRvDaemon	Value depends on the SQM setup	SVD bus RV daemon. For instance: `tcp:11240`
SCSVDRvNetwork	Value depends on the SQM setup	SVD bus network. For instance: `;`
SCSVDRvService	Value depends on the SQM setup	SVD bus service. For instance: `9243`
SCSecondaryDataRvDaemon	Value depends on the SQM setup	Secondary Data bus RV daemon. For instance: `tcp:11240`
SCSecondaryDataRvNetwork	Value depends on the SQM setup	Secondary Data bus Network. For instance: `;`
SCSecondaryDataRvService	Value depends on the SQM setup	Secondary Data bus Service. For instance: `9242`
SCSecondarySDSIRvDaemon	Value depends on the SQM setup	Secondary SDSI bus RV daemon. For instance: `tcp:11240`
SCSecondarSDSIRvNetwork	Value depends on the SQM setup	Secondary SDSI bus network. For instance: `;`
SCSecondarSDSIRvService	Value depends on the SQM setup	Secondary SDSI bus service. For instance: `9251`
SCSelfMgmtRvDaemon	Value depends on the SQM setup	Self Management bus RV daemon. For instance: `tcp:11240`
SCSelfMgmtRvNetwork	Value depends on the SQM setup	Self Management bus Network. For instance: `;`
SCSelfMgmtRvService	Value depends on the SQM setup	Self Management bus service. For instance: `9244`
SCStateDegradationLevel	0.8	Default value for State Degradation level
SCStateViolationLevel	0.2	Default value for State Violation level
SCUnknownValuePolicy	optimistic	Policy applied for unknown service parameter value.
SCXmlMsgValidation	false	If TRUE, all XML messages are validated on reception

Important

The value of the SCDataAcquisitionPeriod is defined during the setup of the SQM platform. It is mandatory to not modify this variable.

If you need to modify the value of SQM Component parameters using this global variable (such as the CalculationPeriod for the SPDM component see 3.3.3), you need to set the new value at the component parameters level.

3.3 SQM Components specific parameters

TIBCO Designer allows visualization and modification of SQM Component parameters accessible through the *Component* part as described in section ” How to edit SQM Components configuration”.

Important

This chapter only describes parameters that can be used to customize the SQM Components. Other parameters that exist in the Central Repository but not described hereafter should not be modified by the user, except if required by the OpenView SQM support.

To avoid unpredictable results, do not delete or insert SQM Component parameters.

3.3.1 Common properties

Each of the SQM components defines at least the following common properties in its own configuration registered into the SQM Central Repository. They are used to configure the log and trace files sink size limit. The Column “Cfg” indicates that this value can be modified in the `$TEMIP_SC_HOME/etc/SRM_setup.cfg` used by the setup when deploying the SQM platform.

Table 2: Common properties defined by each application (<*Appli*>_ instance)

Variable Name	Default	Description	Cfg
Traces			
→ FileCount	10	Maximum number of trace files	Yes
→ FileLimit	10000000	Maximum size for trace files	Yes
→ AppendMode	true	Appends on restart	Yes
Logs			
→ FileCount	10	Maximum number of log files	Yes
→ FileLimit	1000000	Maximum size for log files	Yes
→ AppendMode	true	Appends on restart	Yes

3.3.2 Service Repository Manager

The table below presents all the variables that can be set using the TIBCO Designer to configure the Service Repository Manager application.

Table 3: Service Repository Manager variables (SRM_config instance)

Variable Name	Default	Description
DBConfiguration		
→ DBHost	localhost	Name of the machine hosting the database
→ DBSID	srm	SRM database name
→ DriverType	thin	Driver used for the database connection (OCI/thin)
→ ListenerPort	1521	Port used by the Oracle listener
→ UserName	srm	SRM Database Schema name
→ Password	********[1]	SRM Database Schema Password
→ NetServiceName	…	Database Connection String
→ MaxConnectionNb	5	Max Number of Simultaneous database connections dedicated to the SRM
→ DBTraceMask	5	Internal database trace mask
LookupRequestProcessorThreadNb	5	Number of threads dedicated to the lookup (Get, Find) requests
XmlValidationMode	%%SCXmlValidationMode%%	Global variable. If true, validate the incoming messages against the DTD.
SubsequentRequestRepliesTimeout	%%SCInternalRequestRepliesTimeout%%	Global variable. Timeout value for requests emitted by the SRM
SubsequentRequestRepliesRetryNb	%%SCInternalRequestRepliesRetryNb%%	Global variable. Number of retries for requests emitted by the SRM
UpdateValidationMode	true	If false, the semantic validation items are not executed for update purpose: this might be useful during design phase
QuietMode	false	If true, there is no message emission

[1] Passwords registered by the Central Repository are encrypted. Use the SQM command temip_sc_dbpasswd -passwd <*passwd*> to encrypt your password. If you want to change value of this property in the Central Repository, enter the result of the temip_sc_dbpasswd.

Compliance Objective AutoCreationMode	true	The UI Service Designer allows you to mark a service definition parameter for compliance. Compliance is automatically computed for the objectives defined on this parameter. The compliance values are available on each marked parameter, for a given reference period and a max allowed time, and this as soon as a service level agreement and a service level objectives are associated to them. Setting the ComplianceObjectiveAutoCreationMode flag to true activates the creation of additional objectiveson the computed compliance values. When the ComplianceObjectiveAutoCreationMode flag is false, it is up to the user to define manually (using the SLA Administration) the additional objectives on the computed compliance values.
ComplianceJeopardizeLevel	0.8	Ratio of violation time (over the reference period) bringing a degradation event.
DefaultSigSlaAutoCreationMode	true	If true , this flag activates the automatic creation of Default SIG and Default SLA at Service Def creation.
DefaultOperationName	MyOperation	Name of the default operation created at the first start of the SRM.
DefaultSlaName	SD Name	Name of the default SLA (if created automatically)
DefaultSigName	SD Name	Name of the default SIG (if created automatically)
DefaultSlaAdminState	Unlocked	Value of the Admin state of the automatically created SLAs (one per SD)
SDSIListenCertifiedSessionIdOfSPDM	%%SCPlatform%%_%%SCDirector%%_SPDM_SDSICertifiedSession	This variable allows the creation, in the publisher's ledger, of entries on new published topics even if the pre-registered listener is not connected.
SDSIListenCertifiedSessionIdOfLogger	%%SCPlatform%%_%%SCDirector%%_Logger_SDSICertifiedSession	This variable allows the creation, in the publisher's ledger, of entries on new published topics even if the pre-registered listener is not connected.

3.3.3 Service Performance Data Manager

The table below presents all the variables that can be set in the TIBCO Designer to configure the Service Performance Data Manager application.

Table 4: Service Performance Data Manager variables (SPDM_config instance)

Variable Name	Default	Description
DBConfiguration		
→ DBHost	localhost	Name of the host hosting the database.
→ DBSID	spdm	The SID of the database.
→ DriverType	thin	Driver used to connect database.
→ ListenerPort	1521	DB listener port. Used to create service name.

→ UserName	spdm	DB user name
→ Password	******** [2]	DB password
→ NetServiceName	…	Used to connect to database.
→ OracleHome	/usr/oracle/u01/app/oracle/product/9.2.0[3]	Oracle home
→ DBTraceMask	0	Trace mask enabling tracing in DB.
Misc		
→ XmlValidationMode	%%SCXmlValidationMode%%	
→ QuietMode	false	Process all the Incoming messages, engines are running but no messages are published if the value is set up to true
→ synchronizationMode	false	At the startup, if this variable is set to true, the SPDM considers its data as obsolete. In this case, a complete setup is done. The SPDM requests to the SRM all SQM data needed to work properly
Service		The following thread numbers depend directly on the number of Oracle connections defined in SPDM properties. It is not useful to increase it if the number of Oracle connections is not increased.
→ StorageThreadNumber	3	Max Number of threads that computes PM received from DC
→ LookupThreadNumber	1	Max Number of threads that computes Msg 44 requests
→ CalculationThreadNumber	2	Max number of threads that can run the Calculation Engines simultaneously.
→ Engine		
→ CalculationPeriod(min)	%%SCdataAcquisitionPeriod	Period between each engine run.
→ PropagateUnavailableFlag	false	On Secondary parameter calculation propagate the NoValue of the input parameter.
→ AutomaticModelUpdate	true	Automatically reload new generation of ServiceDefinition updates. If false, the LoadService AMI has to be called manually
→ DataRetentionDelay(days)	1	Number of days of Data kept when doing a Purge Data AMI call.
Status		
→ LookupThreadNumber	3	Max Number of threads that computes incoming Get SOS requests
→ Engine		
→ CalculationPeriod(min)	%%SCdataAcquisitionPeriod	Period between each Compliance engine run.
→ UnkownValuePolicy	%%SCUnknownValuePolicy	When a status is Unknown, define how it impacts the Compliance Calculation. Optimistic : does not impact, as OK Pessimistic: Impacts as KO Realistic: Calculation is not done.
→ DataRetentionDelay(days)	7	Number of days of Statuses kept when doing a Purge Data AMI call

[2] Passwords registered by the Central Repository are encrypted. Use the SQM command temip_sc_dbpasswd -passwd <*passwd*> to encrypt your password. If you want to change value of this property in the Central Repository, enter the result of the temip_sc_dbpasswd.

[3] Actual value of OracleHome is set during the SQM setup.

→ ResetComplianceOnUnlock	false	if true, unlocking a SLA resets all the compliance values under it. Unlocking a SI/SCI resets all its compliance values and all its SLO compliance values. If false, previous compliance values are kept
→ ResetComplianceOnSlaUpdate	true	if true, changing the SIG or SL of the SLA resets the compliance of the SLA and all its sub-components. Note that modifying the SIG or the SL themselves does to trigger the reset. If false, nothing is done (this can lead to strange behaviors).
OperationView		
→ LookupThreadNumber	3	Max Number of threads that computes incoming CAView requests
SRM		
RequestRepliesTimeout(ms)	%%SCInternalRequestRepliesTimeout%%	
RequestRepliesRetryNb	%%SCInternalRequestRepliesRetryNb%%	
NbMaxResults	1000	Maximum number of records that a reply can contain in the filtered requests to the SRM. If the number of records to retrieve is greater than the fixed limit, subsequent requests are sent to the SRM in a "handle more" mode till all SRM objects are returned.

3.3.4 Data Collector

The table below presents all the variables that can be set in the TIBCO Designer to configure the Data Collector application.

The Column "Cfg" indicates that this value can be modified in the $TEMIP_SC_HOME/etc/DC_Setup.cfg to be taken into account when deploying the whole SQM platform.

Table 5: Data Collector variables (DC_config instance)

Variable Name	Default	Description	Cfg
XmlValidationMode	%%SCXmlValidationMode%%		
NSRequestRepliesTimeout	%%SCInternalRequestRepliesTimeout%%		
NSRequestRepliesRetryNb	%%SCInternalRequestRepliesRetryNb%%		
SRMRequestRepliesTimeout	%%SCInternalRequestRepliesTimeout%%		
SRMRequestRepliesRetryNb	%%SCInternalRequestRepliesRetryNb%%		
NbMaxResults	1000	Maximum number of records that a reply can contain in the filtered requests to the SRM. If the number of records to retrieve is greater than the fixed limit, subsequent requests are sent to the SRM in a "handle more" mode till all SRM objects are returned.	
CacheSize	100	Nb of Cache entries (Least Recently Used) of Subscriber /Customer couple	Yes

DFIValidationMode	false	If true, the DC validates that the parameters in the Perf Measure (Msg 52) are matching their SRM declaration regarding their datatype	Yes
NoValuePropagation	true	When DC cannot publish a value if this variable is true, it publishes a “no value” else it discards it.	Yes
SAPassiveMonitoring	true	If true, when DC detects that a SA is down, it sends a Msg 67 to SRM.	Yes
SAHeartBeat	%%SCFTGheartbeat(sec)%%	Must be equal to Heartbeat variable of the SA.	
DataBurstPolicy			
→ ActivationFlag	false	If true, the DC can discard some messages in case of burst of Msg 52	Yes
→ LimitRateMsgPerSecondPerDFI	10	If activation Flag is true, the DC won’t publish more messages than set in this variable period. Other messages will be discarded.	Yes
QuietMode	false	If True, The DC computes the Msg 52 but does not publish anything to the SPDM	Yes

3.3.5 Naming Service

The table below presents all the variables that can be set in the TIBCO Designer to configure the Naming Service application.

Table 6: Naming Service variables (NS_config instance)

Variable Name	Default	Description
XmlValidationMode	%%SCXmlValidationMode%%	
CustomerValidationMode	false	If set to true, verify the validity of the Customer while loading the Naming Schema. When the "CustomerValidationMode" is enabled, during the customer validation, the NS will check if more than one Customer match a given subscriber/domain couple and will rise an error if this is the case. Note that activating the "CustomerValidationMode" has a huge cost on the NS performance.
QuietMode	false	
CustomersResolution		
→ Number of Thread	5	Number of active threads to validate a subscriber/domain
→ CacheSize	100	Number of subscribers in the cache
SRMRequestRepliesTimeout	%%SCInternalRequestRepliesTimeout%%	

SRMRequestRepliesRetryNb	%%SCInternalRequestRepliesRetryNb%%	

3.3.6 Service Level Objective Manager

The table below presents all the variables that can be set in the TIBCO Designer to configure the Service Level Objective Manager application.

Table 7: Service Level Objective Manager variables (SLOM_config instance)

Variable Name	Default	Description
XmlValidationMode	%%SCXmlValidationMode%%	
SRMRequestRepliesTimeout	%%SCInternalRequestRepliesTimeout%%	
SRMRequestRepliesRetryNb	%%SCInternalRequestRepliesRetryNb%%	
SRMReplyNbMaxResults	1000	Maximum number of records that a reply can contain in the filtered requests to the SRM. If the number of records to retrieve is greater than the fixed limit, subsequent requests are sent to the SRM in a "handle more" mode till all SRM objects are returned.
SPDMRequestRepliesTimeout	%%SCInternalRequestRepliesTimeout%%	
SPDMRequestRepliesRetryNb	%%SCInternalRequestRepliesRetryNb%%	
GetHistoricalValue	true	If true, at startup (or update) the SLOM re-computes the unknown statuses from the last parameter values stored in the SPDM.
GetHistoricalStatus	true	If true, at startup the SLOM initializes all the statuses with the latest statuses stored in SPDM.
ValidateHistoricalStatus	true	The components statuses will be recomputed from the SLO historical statuses (Warning: not from the historical values).
QuietMode	false	The SLOM computes the statuses but does not publish SMS messages
GlobalStatusPropagation	Worst	Global Status computation policy : **Worst**:. Parent node takes the worst status. **Average** : Parent node status is the average of child nodes statuses
SpdmDBConfiguration		
→ SpdmHostName	localhost	Name of the machine hosting the SPDM database
→ SpdmInstanceName	spdm	The SID of the SPDM database
→ SpdmDriverType	thin	Driver used to connect to the SPDM database
→ SpdmPortNumber	1521	SPDM DB listener port. Used to create service name
→ SpdmUserName	spdm	SPDM DB user name
→ SpdmPassword	******** [4]	SPDM DB password
→ SpdmServiceName	…	Used to connect to the SPDM database

[4] Passwords registered by the Central Repository are encrypted. Use the SQM command temip_sc_dbpasswd -passwd <*passwd*> to encrypt your password. If you want to change the value of this property in the Central Repository, enter the result of the temip_sc_dbpasswd.

3.3.7 Logger

Repository Configuration

The table below presents all the variables that can be set in the TIBCO Designer to configure the Logger application.

The Column “Cfg” indicates that this value can be modified in the $TEMIP_SC_HOME/etc/Logger_setup.cfg to be taken into account when deploying the whole SQM platform.

Table 8: Logger variables (Logger_config instance)

Variable Name	Default	Description	Cfg
DBConfiguration			
→ DBDriverType	thin	Driver used to connect database.	Yes
→ DBTargetListenerPort	1521	Logger DB listener port. Used to create service name.	Yes
→ DBTargetHost	localhost	Name of the host hosting the Logger database.	Yes
→ DBTargetSID	logger	The SID of the Logger database.	Yes
→ DBTargetNetServiceName	…	Used to connect to Logger database.	Yes
→ DBTargetUserName	logger	Logger DB user name	Yes
→ DBTargetPassword	*********[5]	Logger DB password	Yes
→DBMaxNbMessagesPerTransaction	10	max number of messages that are inserted into the Logger DB during one transaction	Yes
→DBMaxNbSecondsbetweenTransactions	10	max interval of time (in seconds) between two transactions	Yes
→ DBPurgeListenerPort	1521	Purge DB listener port. Used to create service name.	Yes
→ DBPurgeHost	localhost	Name of the host hosting the Purge database.	Yes
→ DBPurgeSID	arch	The SID of the Purge database.	Yes
→ DBPurgeNetServiceName	…	Used to connect to Purge database.	Yes
→ DBPurgeUserName	purger	Purge DB user name	Yes
→ DBPurgePassword	********[5]	Purge DB password	Yes
→DBMaxTimeInFault	60	max interval of time (in minutes) during which the Logger tries to open a database connection in case of errors	Yes

3.3.8 User Interface Server

The table below presents all the variables that can be set in the TIBCO Designer to configure the Service User Interface Server application.

[5] Passwords registered by the Central Repository are encrypted. Use the SQM command temip_sc_dbpasswd -passwd <*passwd*> to encrypt your password. If you want to change value of this property in the Central Repository, enter the result of the temip_sc_dbpasswd.

Table 9: User Interface Server variables (screpos/ServiceCenter/UserInterface/UIServer/v1_0/UIS_config instance)

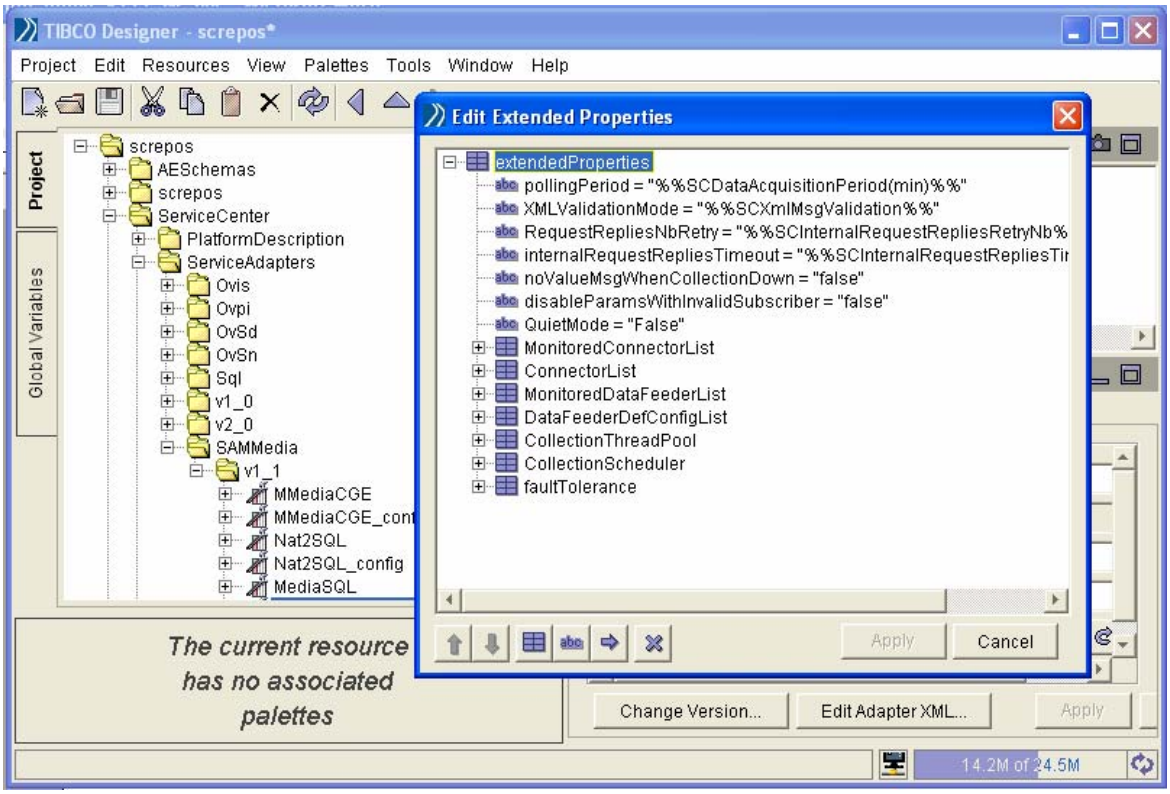
Variable Name	Default	Description
OrbNamingServiceHost	This value is initialized during the setup of the platform	Name of the host of the CORBA Naming Service. This is the host of the presentation director.
ClientCommunicationCorbaPort	7172	TCP Port used by the Presentation Server to communicate with the SLA Monitoring and the SLA Administration UI
JacorbConfigDir	%%DIR_SCVar%%/UI/UIServer/	Root folder that contains the Jacorb configuration file (<JacorbConfigDir>/config/jacorb.properties)
RepoUrlAdmin	tibcr@screpos:subject=com.tibco.repo.instance_discovery.request:service=%%SCRepoRvService%%:network=%%SCRepoRvNetwork%%:daemon=%%SCRepoRvDaemon%%:scRepoAdmin	Pattern used to build the Tibco Repository that the SLA Administration will use to contact the SQM Core Servers (once the user has successfully logged in).
UserInterfaceServerName	defaultServer	"CORBA Name" of the Presentation Server.
UserAuthentication	true	If true the password entered by the SLA Monitoring user is checked in the /etc/passwd of the machine.
AuthorizedGroupCommon	sqmadm	SQM Reserved for future use.
AuthorizedGroupSlaAdministration	users;sqmadm	List of UNIX groups allowed to open a SLA Administration session. To be able to open a SLA Administration session, the user must be part of one of these groups. The separator is '**;**'.
AuthorizedGroupSlaMonitoring	users;sqmadm	List of UNIX groups allowed to open a SLA Monitoring session. To be able to open a SLA Monitoring session, the user must be part of one of these groups. The separator is '**;**'.
UsersUrl	%%DirSCVar%%/UI/UIServer/users/config/	Directory where users' configuration files and launch definition files are stored.
UserSessionGarbagePeriod(Min)	10	The period used to garbage the unused User Sessions.
NbThreadsOfSDSISession	3	Number of threads dedicated to the processing of replies and events sent by the SRM.
NbThreadsOfSVDSessionReqRep	4	Number of threads dedicated to the processing of the statuses lookup replies.
NbThreadsOfSecondaryDataSession	4	Number of threads dedicated to the processing of the parameters ' values replies and events sent by the SPDM.
SlaInFilteredLookupMaxNumber	50	Maximum number of SLAs that the SRM can send when the User Interface Server requests a filtered list of SLAs.
HistoricalValuesMaxNumber	1000	Maximum number of data the SPDM can send when the User Interface Server requests archived parameters values.
HistoricalSOSMaxNumber	1000	Maximum number of data the SPDM can send when the User Interface Server requests archived parameters statuses.
MaximumNumberOfDataPerRetentionQueue	50000	Number max of data (status / value) the UIS can store for a SLA Monitoring session between 2 SLA Monitoring polls.
Others	Mapped on a Global variable	Cf description of the associated Global variable.

3.3.9 Service Adapters

This chapter describes configuration variables common to all SQM Service Adapters.

Details are described in the OpenView SQM document “SQM Service Adapters User’s Guide” or also available in the appropriate Service Adapter Installation and Configuration Guide.

3.3.9.1 Service Adapter Repository configuration



The Service Adapter application configuration is generally at the following location in the TIBCO Repository:

/ screpos / ServiceCenter / ServiceAdapters /
<ServiceAdapterName>/<ServiceAdapterVersion>/<ApplicationName>_config

Once the application configuration object is selected, it is possible to edit and modify the parameters by selecting menu “***Resources***” -> “***Edit Extended Properties***”.

The table below presents all the variables that can be set in the TIBCO Designer to configure a Service Adapter application itself.

Table 10: Service Adapters variables (Service Adapter config instance)

Variable Name	Default	Description
pollingPeriod	%%SCDataAcquisitionPeriod(min)%%. (Global variable)	This polling period applies to all the data collection by all of the data feeder instances managed by this service adapter instance. Each time this period is reached, each data feeder instance gets the parameter values coming from the third party product. This does not mean that all data collections are done at the same time, but that their frequency is equal to the pollingPeriod. This variable is applicable for Service Adapters that are using polling mechanism to collect data. The minimum pollingPeriod value is 0.5 (corresponds to 30 seconds)
QuietMode	false	When the quiet mode is set to "True", the application continues to work exactly like before, but it no longer publishes performance data. This mode is useful to keep Service Adapters from publishing data on the bus, allowing you to investigate problems or to migrate from a version of this service adapter to a more recent one.
XMLValidationMode	%%SCXmlMsgValidation%% (Global variable)	When the XMLValidationMode variable is active, all the messages sent or received are checked against their respective DTD to detect errors. This mode can be useful when troubleshooting the SAI.
InternalRequestRepliesTimeout (optional)	%%SCInternalRequestRepliesTimeout%% (Global variable)	The internalRequestRepliesTimeout variable provides the maximum time the service adapter instance is authorized to wait for the reply when sending requests to another SQM component (SRM for example).
RequestRepliesNbRetry	%% SCInternalRequestRepliesRetryNb%% (Global variable)	Number of retries when an internal requests times out
DisableParamsWithInvalidSubscribers (optional)	false	The variable is an internal flag used to avoid emitting performance data messages for customer dependent parameters when the subscriber information could not be retrieved properly. If this flag is set to "true", these parameters are filtered by the service adapter instance and are not sent to the Service Level Monitoring layer (the error is logged).
NoValueMessageWhenCollectionDown (optional)	false	This variable is an internal flag set to false to avoid emitting "Not Available" messages when there is an error getting performance data. This variable can be enabled to force the emission of a performance data message (with all the parameters set to NoValue and their suspect flags set to true), even when data collection is down. Normally, the Service Adapter application does not emit a performance data message (to avoid creating useless traffic). Instead, it sends an update about the collection status of the data feeder instance to the OpenView SQM repository layer.
NbMaxResults	1000	Maximum number of records that a reply can contain in the filtered requests to the SRM. If the number of records to retrieve is greater than the fixed limit, subsequent requests are sent to the SRM in a "handle more" mode till all SRM objects are returned.
MonitoredConnectorList		The MonitoredConnectorList variable contains a list of the connector's logical names that are used by this service adapter instance.
C If the number to retrive is greater than the fixed limit multiple replies will be sent in a handle more mode.onnectorList		The ConnectorList variable contains all the connection data. All the monitored connectors should be found inside the configuration list.

MonitoredDataFeederList		The MonitoredDataFeederList variable contains a list of the data feeder definitions managed by this service adapter instance. Each data feeder definition is identified by its name and its version.
DataFeederDefConfigList		The DataFeederDefConfigList variable contains all the configuration data of Data Feeder Definitions. All the monitored Data Feeder Definitions should be found inside the configuration list.
Fault Tolerance		
heartbeat	%%SCFTGheartbeat (sec)%%	This variable represents the heartbeat period of the Service Adapter to notify the Data Collector that the application is still up and running. This value has to be equal to the DataCollector SA Heartbeat period It must be greater than preparationInterval and lower than activationInterval
preparationInterval	20	Before becoming active, Service Adapters processes might need to do some time-consuming steps, such as connecting to an agent or opening a database connection. Hence, such processes need to prepare before they can activate. This interval represents
activationInterval		When the heartbeat signal from one or more active members has been silent for this interval (in seconds), TIBCO Rendezvous fault tolerance software considers the silent member to be lost.

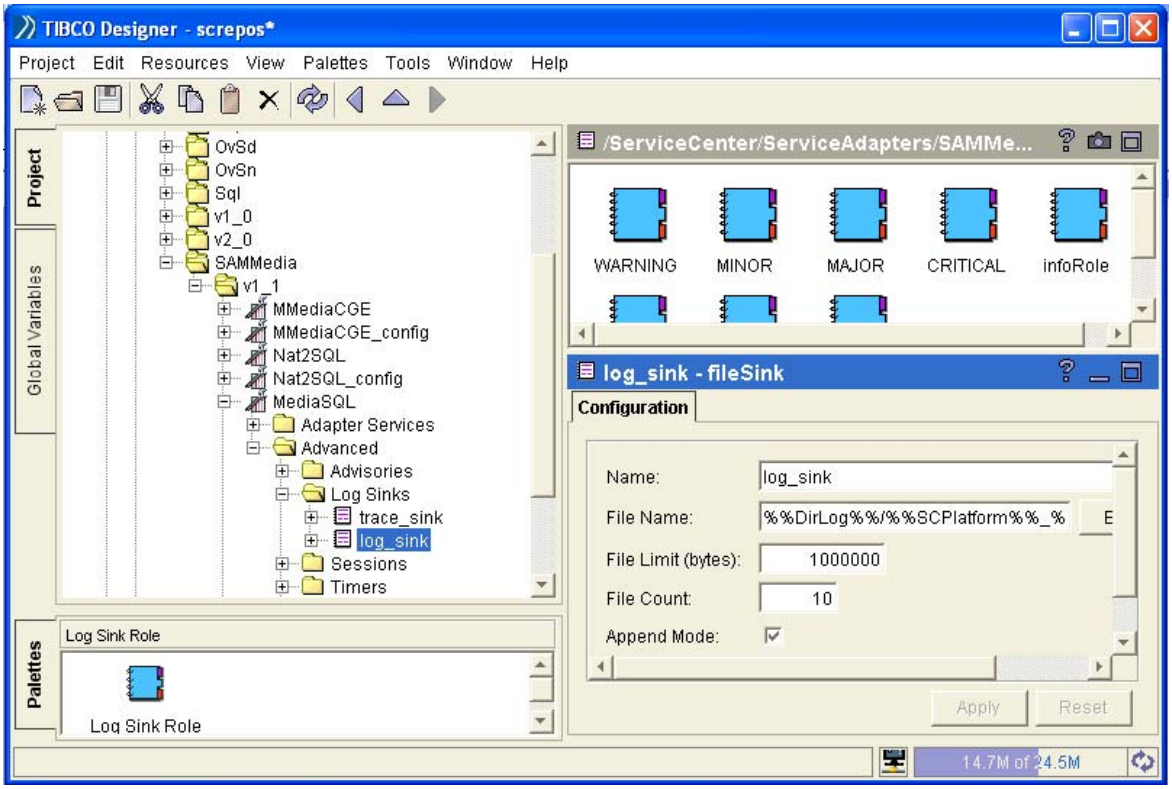
In the repository, it is also possible to manage the Service Adapter application trace and log files (size and count). Editing the application instance:

1- For traces, with the path

/ screpos / ServiceCenter / ServiceAdapters / <ServiceAdapterName>/<ServiceAdapterVersion>/<ApplicationName>/Advanced/Log Sinks/trace_sink

2- For Logs, with the path

/ screpos / ServiceCenter / ServiceAdapters / <ServiceAdapterName>/<ServiceAdapterVersion>/<ApplicationName>/Advanced/Log Sinks/log_sink

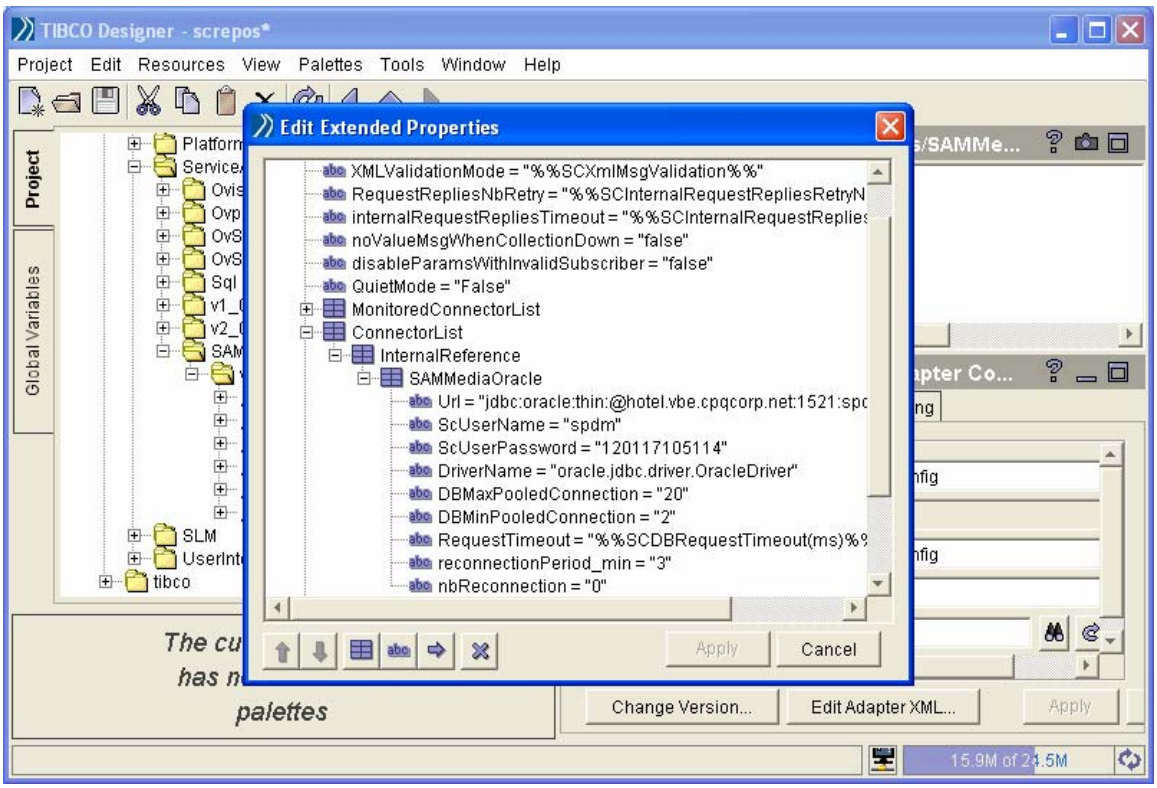


Connector Configuration Data

The Service Adapter connector configuration data are available in the application configuration.

Using the TIBCO designer, you can edit the connector configuration information with the menu “Edit Extended Properties” on the following path:

/ screpos / ServiceCenter / ServiceAdapters /
<ServiceAdapterName>/<ServiceAdapterVersion>/<ApplicationName>_config/extendedProperties/
ConnectorList/InternalReference/<connectorName>



The connector name is the parent cell of the connection parameters tree. This connector name cannot contain a period (.) in its name. The connector name is a logical name (in Figure 8: TIBCO Hawk Display console, it is the name of the host of the database with an underscore character (_) instead of period (.)). The real hostname belongs to the configuration parameters of the connection.

For each service adapter, connection parameters may be completely different. In the example, the connection parameters apply to connecting with the Oracle database.

The reconnection period (ReconnectionPeriod_min) and the number of reconnections (NbReconnection) are configuration parameters common to all Service Adapters. If these parameters are not specified, default values will be used. When the connection is in error, a polling mechanism starts and attempts to reopen the connection. A reconnection value of “0” means that it will attempt reconnecting an infinite number of times. The reconnection polling-period is given in minutes.

Reconnection and polling periods are independent, and have to fit the following basic constraints:

Reconnection period: min = 1 mn, max = 24 hr, default = 10 mn

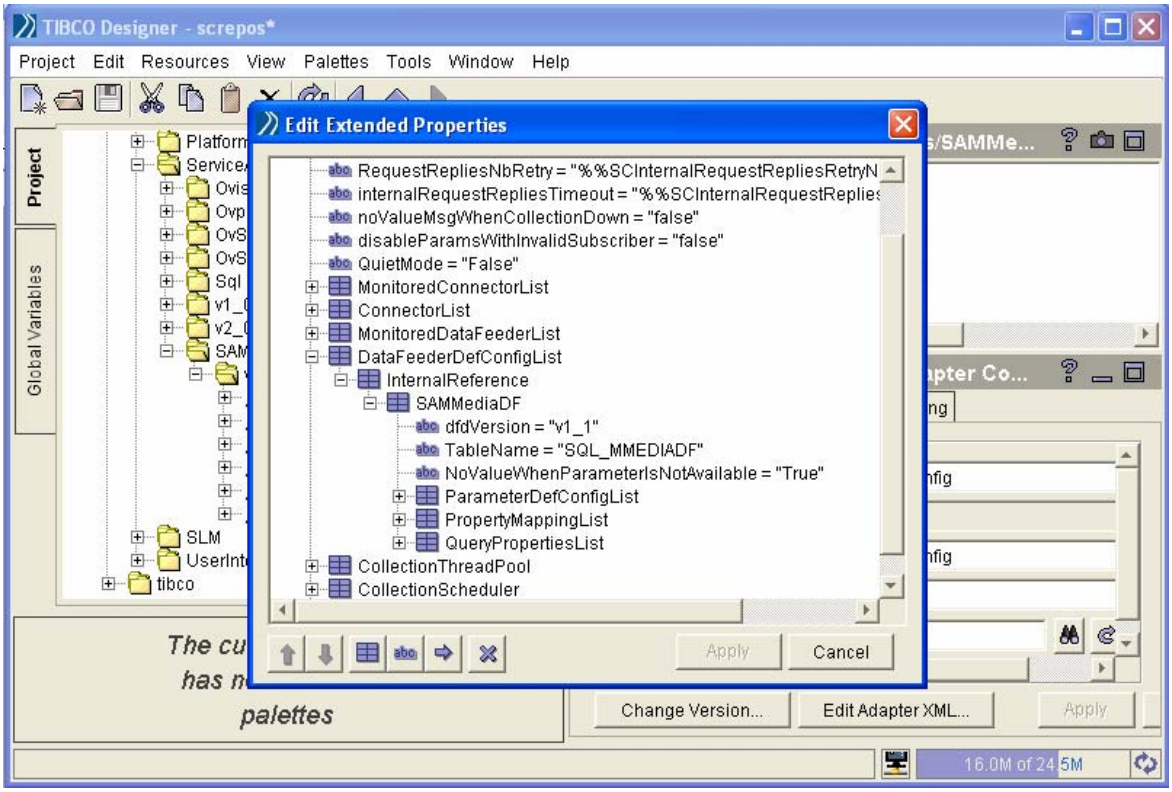
Polling period: min = 1 s, max = 24 hr, default = 5 mn

For Service Adapters working in unsolicited mode (they receive the collection data asynchronously), it is recommended to set the polling period to a big value. Nevertheless, to make sure that everything received has been published, you should set it to a mean value to have the polling mechanism working as a heartbeat mechanism.

Data Feeder Definition Configuration Data

The Service Adapter application configuration data includes the definition of the DataFeeders at the following path:

/ screpos / ServiceCenter / ServiceAdapters /
<ServiceAdapterName>/<ServiceAdapterVersion>/<ApplicationName>_config/extendedProperties/
DataFeederDefConfigList/InternalReference/<DataFeederName>



The DataFeederDefConfigList structure contains configuration data required to map OpenView SQM parameter names to the real parameter name in the third party product.

The root level of the “SAMMediaDF” DFD contains the configuration parameter that maps this DFD with a SQL table name. It also contains the DFD version, which is mandatory. Additional mapping information is available in ParameterDefConfigList, PropertyMappingList, QueryPropertiesList.

The configuration data may differ slightly for each Service Adapter. This example tries to describe the main principles for the information stored in the TiBCO repository for each DFD.

3.3.10 Gateways

Please refer to the appropriate Gateway Installation and Configuration Guide.

3.3.11 SLA Monitoring UI

Some settings of the SLA Monitoring can be customized by the SQM Administrator.

This customization is mainly about

- CORBA communication timeout
- Icons that are in the dialog box “Definition Icon..” and that can be associates to the nodes of the SLA View
- Rendering formats (dates, relative times, floating point numbers)
- GUI Settings (Chart size, Table columns size,…)

- …

3.3.11.1 SLA Monitoring UI Configuration File

The SQM SLA Monitoring UI has a default value for each of these settings. The SQM Administrator can override any of these values by adding them into the SLA Monitoring configuration file.

Default location of the configuration file

The default location is "`%USERPROFILE%\HP OpenView\SQM\UI\SLMClient\config\TeSCUISLMClientConfig.properties`"

Due to the fact that the configuration file is stored into `%USERPROFILE%`, if several operators use the SLA Monitoring UI on the same PC, each operator has his own configuration file.

The location of the file can be changed:

- To centralize a backup operation
- To share the same file by all the users in order to simplify the administration if the customization is important and the same for all the users.

Changing the location of the configuration file (Standard Installation)

If the SLA Monitoring UI is installed locally on the PC (standard installation), edit the file `%TEMIP_SC_HOME/UI/SLMClient/bin/TeSCUISLMClient_extra_var_env.bat`

and add the line

```
set UI_SLM_SYSTEM_PROPERTIES=%UI_SLM_SYSTEM_PROPERTIES% -
Dsqm.config.filename=my_path_to_config_folder\MySLMClientConfig.properties
```

Changing the location of the configuration file (Java Webstart Deployment)

If the SLA Monitoring UI is deployed using Java Web Start, edit the SLAMonitoring Java Web Start deployment descriptor file:
`<SQM_WEB_START_INSTALLATION_DIRECTORY>/webstart/TeSCUISLMClient.jnlp`

Define the system property `sqm.config.filename`

Example:

```
<property name="sqm.config.filename"
value="http://your_website_url/custom/TeSCUISLMClientConfig.properties"/>
```

Notes

- The Configuration file is created at startup if it does not exist yet
- A Copy of the file delivered in the kit is stored into
  `%USERPROFILE%\HP OpenView\SQM\kit\...`
  it can be used as a template to create your own configuration file
- The file location can be
  - either a path
    `drive:\folder\MySLMClientConfig.properties`
  - or an url:
    `http://your_website_url/custom/MySLMClientConfig.properties`
- The configuration file is a java property file. The syntax is
  - Key=value
  - # comments the line

3.3.11.2 Most important customizations

This chapter describes the most important (and commonly modified) keys.

Some others are described in the configuration file. They are commented (not applied). The values set are the default used.

Other settings are not documented and must be considered as non public.

Caution: Any customization done in this file has to be considered as an Administration task. No check is done regarding the validity of the value (example: “FooValue” when an integer is expected). Any wrong value can cause unexpected behavior in the SLA Monitoring UI.

Communication settings

Corba Timeout for a Request (in seconds)

- uis.client.com.timeout_seconds.reply=180

Add Icons to the Definition Icons… dialog box

To add new icons in the Dialog Box "Select Icons for Definition”:

- Copy the icon (16*16 gif image) in a folder (accessible by the Windows user) or in a web server (the one used by web start for example).
- In the configuration file:
    - Add a property line per icon. The line must look like `icon_bank.node_def.icon_<id>.url=<url>`
    - Each <id> must be unique
    - The SLA Monitoring will read the lines with contiguus <id> starting 0.
    - <url> is the URL that locates the icon. It can be
        - an URL that locates the icon file on a Web Server `http://serverwebhostname[:port]/urlpath/myicon.gif`
        - an URL that locates the icon file on the workstation
          example : if the icon myicon.gif is located in the folder c:\myicons
          `file:/c:/myicons/myicon.gif`

To change the size of this dialog box, use the following keys. The values are in pixels

- `common.select_icon_dlg.width`
- `common.select_icon_dlg.height`

Rendering format

SQM manages floating point numbers.

- The configuration key `monitoring.number_formatters.number_of_significant_digits` specifies the number of significant digit for floating point numbers. The default value is 4.
  It identifies the number of digit that are meaning full.
  Examples:
    - 1/3 = 0.333333333333333 will be displayed as 0.3333 if the number of significant digit is 4
    - 123.456789 will be displayed 123.5

Floating point numbers can be very small (example 0.00000012345678) or “big numbers” (example 12345678000000). In order to render correctly any kind of number, three ranges of numbers are identified. A specific rendering is associated to each range:

- “big numbers”. Numbers > 1.10^{b}
    - The configuration key stores the exponent (**b**).
      `monitoring.number_formatters.exponent_of_big_number_limit`.
      Default value is 4
    - The rendering uses the scientific notation
      example 12341.234 will be rendered 1.234 E 5 (=$1.234 * 10^{5}$). In this example, the number of significant digits used is 4.
- “small numbers”. Numbers < 1.10^{-s}
    - The configuration key stores the exponent (**s**).
      `monitoring.number_formatters.exponent_of_small_number_limit`.
      Default value is 4
    - The rendering uses the scientific notation
      example: 0.0000012341234 will be rendered 1.234 E -6 (=$1.234 * 10^{-6}$) In this example, the number of significant digits used is 4.
- “middle numbers” : 1.10^{-s} < Numbers < 1.10^{b}
    - Numbers in this range are displayed using the decimal format.
      Example :
        - 123.4123 will be rendered 123.4
        - 0.01234123 will be rendered 0.01234

SQM Server settings (replication)

In order to have a better chart representation of a data, it is recommended (but not mandatory) to set the property `monitoring.core_server.calculation_period_min` to the same value that the TIBCO Designer global variable "`SCDataAcquisitionPeriod(min)`”.

User Interface settings

The most important graphical user interface settings are:

- Tooltip management (time before display,…)
  `monitoring.gui.tooltip….`
- Width of the columns of the tables
    - Properties (of Service, Service Component, Service Group) Table
      `monitoring.gui.table.col.width.properties…..`
    - Parameters Value table in Detail Tabbed view
      `monitoring.gui.table.col.width.param….`
    - Scope Definition Dialog Box
      `monitoring.gui.table.col.width.scope....`
    - SLA Scope in SLA Bar
      `monitoring.gui.table.col.width.sla_summary....`
    - Parameter Current value Table in dashboards
      `monitoring.gui.table.col.width.cvt….`
    - Chart Properties Dialog box. Configuration of the parameters
      `monitoring.gui.table.col.width.param_conf….`
    - Current Value Table Properties Dialog box. Configuration of the parameters
      `monitoring.gui.table.col.width.cvt.options_table.….`

- Result table for a Find… of a Search … action
  `monitoring.gui.table.col.width.result.fis....`

- Width of the columns of the Charts
  - Line chart
    ```
    monitoring.grt_mgr.grt_0.fixed_height=320
    monitoring.grt_mgr.grt_0.preferred_width=400
    ```
  - Area chart
    ```
    monitoring.grt_mgr.grt_1.fixed_height=320
    monitoring.grt_mgr.grt_1.preferred_width=400
    ```
  - Bar chart
    ```
    monitoring.grt_mgr.grt_2.fixed_height=320
    monitoring.grt_mgr.grt_2.preferred_width=400
    ```
  - Current Value Table
    ```
    monitoring.grt_mgr.grt_3.fixed_height=320
    monitoring.grt_mgr.grt_3.preferred_width=585
    ```

- Add possible color for a parameter in a chart
  Replace X with a number starting with 9 and increasing 1 at every new color. In this example, the color will be dark green
  ```
  #color #X in the combo of available colors
  monitoring.grt.plot_and_bar.color_x.red=21
  monitoring.grt.plot_and_bar.color_x.green=116
  monitoring.grt.plot_and_bar.color_x.blue=21
  ```

3.3.12 SLA Administration UI

Some settings of the SLA Administration can be customized by the SQM Administrator.

This customization is mainly about:

- CORBA communication timeout
- Sizing of the result sets retrieved form the SQM Servers
- Rendering formats
- GUI Settings
- …

3.3.12.1 SLA Administration UI Configuration File

The SQM SLA Administration UI has a default value for each of these settings. The SQM Administrator can override any of these values adding them in the SLA Administration configuration file.

Default location of the configuration file

The default location is "`%USERPROFILE%\HP OpenView\SQM\UI\SLAClient\config\TeSCUISLAClientConfig.properties`"

Due to the fact that the configuration file is stored into `%USERPROFILE%`, if several operators use the SLA Administration UI on the same PC, each operator has his own configuration file.

The location of the file can be changed:

- To centralize a backup operation
- To share the same file by all the users in order to simplify the administration if the customization is important and the same for all the users.

Changing the location of the configuration file (Standard Installation)

If SLA Monitoring UI is installed locally on the PC (standard installation), edit the file `%TEMIP_SC_HOME/UI/SLAClient/bin/temip_sc_sla_start.bat`

And, after line 21: `set JAVA_ARGS=-Xms128M -Xmx340M`, add the line
`set JAVA_ARGS=%JAVA_ARGS% -Dsqm.config.filename=my_path_to_config_folder\MySLAClientConfig.properties`

Notes

- The Configuration file is created at the startup if it does not exist yet.
- A Copy of the file delivered in the kit is stored in
  `%USERPROFILE%\HP OpenView\SQM\kit\...`
  it can be used as a template to create your own configuration file
- The file location can be
  - either a path
    `drive:\folder\MySLAClientConfig.properties`
  - or an url:
    `http://your_website_url/custom/MySLAClientConfig.properties`
- This is a java property file. The syntax is
  - Key=value
  - # comments the line

3.3.12.2 Most important customizations

This chapter describes the most important (and commonly modified) keys.

Some others are described in the configuration file. They are commented (not applied). The values set are the default used.

Other settings are not documented and must be considered as non public.

Caution: Any customization done in this file has to be considered as an Administration task. No check is done regarding the validity of the value (example: “FooValue” when an integer is expected). Any wrong value can cause unexpected behavior in the SLA Administration UI.

Communication settings

Corba Timeout for a Request (in seconds)

- `uis.client.com.timeout_seconds.reply=180`

Renderer for Dates

The rendering of dates can be changed by modifying the value of the key:
`default.absolute.time.format`.

Default value is `yyyy-MM-dd' 'HH:mm:ss.SSS`

The format must respect Java class SimpleDateFormat documentation (see http://java.sun.com/j2se/1.4.2/docs/api/index.html)

GUI settings

Some keys allow setting the initial size of the SLA Administration UI main frame and internal windows
`window…`

Chapter 4

Command Line Utilities

The service management UNIX command line utilities (a.k.a. CLUI) are dedicated to manage the services you have created on the SQM platform.
They can be invoked by any user that belongs to the sqmadm user group. Unlike the SQM administration commands, the sqmadm user id is not required to invoke the Service Management command line utilities.
For each tool described in this chapter, you can use the following parameters:

-h|help : to display the command line syntax

-version : display tools and versions

Optional parameters:

-timeout < value> : set timeout in second for call

-u <Configuration File> : specify a configuration file. By default, the configuration file of the current SQM platform is taken.

4.1 Create utilities

4.1.1 temip_sc_create_cust

The **temip_sc_create_cust** command creates a customer in the SQM Service Repository.

Command line syntax

`temip_sc_create_cust –i | -input <fileList>`

Parameters description

<fileList> : XML file that contains customer information following the Create Customer DTD.

4.1.2 temip_sc_create_expression

The **temip_sc_create_expression** command creates a calculation expression definition in the SQM Service Repository.

Command line syntax

`temip_sc_create_expression –i | -input <fileList>`

Parameters description

<fileList> : XML file that contains calculation expression information following the Create Calculation Expression DTD.

4.1.3 temip_sc_create_sd

The **temip_sc_create_sd** command creates a Service Definition in the SQM Service Repository.

Command line syntax

```
temip_sc_create_sd   –i | -input   <fileList>
```

Parameters description

<*fileList*> : XML file that contains service definition information following the Create SD DTD.

4.1.4 temip_sc_create_si

The **temip_sc_create_si** command creates a service instance in the SQM Service Repository.

Command line syntax

```
temip_sc_create_si   –i | -input   <fileList>
```

Parameters description

<*fileList*> : XML file that contains service instance information following the Create SI DTD.

4.1.5 temip_sc_create_sig

The **temip_sc_create_sig** command creates a service instance group in the SQM Service Repository.

Command line syntax

```
temip_sc_create_sig   –i | -input   <fileList>
```

Parameters description

<*fileList*> : XML file that contains service instance group information following the Create SIG DTD.

4.1.6 temip_sc_create_sla

The **temip_sc_create_sla** command creates a service level agreement in the SQM Service Repository.

Command line syntax

```
temip_sc_create_sla   –i | -input   <fileList>
```

Parameters description

<*fileList*> : XML file that contains service level agreement information following the Create SLA DTD.

4.1.7 temip_sc_create_sl

The **temip_sc_create_sl** command creates a service level in the SQM Service Repository.

Command line syntax

```
temip_sc_create_sl   –i | -input   <fileList>
```

Parameters description

<*fileList*> : XML file that contains service level information following the Create SL DTD.

4.1.8 temip_sc_create_dfd

The **temip_sc_create_dfd** command creates a data feeder definition in the SQM Service Repository.

Command line syntax

```
temip_sc_create_dfd   –i | -input   <fileList>
```

Parameters description

<*fileList*> : XML file that contains data feeder information following the Create DFD DTD.

4.1.9 temip_sc_declare_dfi

The **temip_sc_declare_dfi** command declares a Data Feeder Instance in the SQM Service Repository.

Command line syntax

```
temip_sc_declare_dfi   –i | -input   <fileList>
```

Parameters description

<*fileList*> : XML file that contains Data Feeder Instance information following the Declare DFI DTD.

4.2 Get utilities

4.2.1 temip_sc_get_cust

The **temip_sc_get_cust** command gets the list of customers.

Command line syntaxes

- Get a list of Customers:
  ```
  temip_sc_get_cust
  ```
- Get the full information on a Customer:
  ```
  temip_sc_get_cust -cust <custName>
  ```
- Get Customer information from a given SI:
  ```
  temip_sc_get_cust -sd <sdName> -si <siName>
  ```

Parameters description

<*custName*> : Customer name.

<*sdName*> : Service Definition name.

<*siName*> : Service Instance name.

4.2.2 temip_sc_get_dfd

The **temip_sc_get_dfd** command gets a summary of all Data Feeder Definitions.

Command line syntaxes

- Get a summary list of all Data Feeder Definitions:
  `temip_sc_get_dfd`
- Get a detailed list of all Data Feeder Definitions:
  `temip_sc_get_dfd –verbose`
- Get the definition of a specific Data Feeder:
  `temip_sc_get_dfd -dfd <dfdName> -dfdv <dfdVersion>`
- Get Data Feeder Definition from a Service Definition:
  `temip_sc_get_dfd -sd <sdName>`
- Get Data Feeder Definition from a Service Component:
  `temip_sc_get_dfd -sc <scName>`

Parameters description

<*dfdName*> : Data Feeder name.

<*dfdVersion*> : Data Feeder version.

<*sdName*> : Service Definition name.

<*scName*> : Service Component name.

4.2.3 temip_sc_get_dfdexpbind

The **temip_sc_get_dfdexpbind** command gets the list of expressions defined on a Data Feeder.

Command line syntax

`temip_sc_get_dfdexpbind -dfd <dfdName>`

Parameters description

<*dfdName*> : Data Feeder name.

4.2.4 temip_sc_get_dfi

The **temip_sc_get_dfi** command gets the definition of Data Feeder Instance.

Command line syntaxes

- Get the list of DFI for a given DFD:
  `temip_sc_get_dfi -dfd <dfdName>`
  `temip_sc_get_dfi -dfd <dfdName> -dfdv <dfdVersion>`
- Get the definition of a Data Feeder Instance:
  `temip_sc_get_dfi -dfd <dfdName> -dfdv <dfdVersion>`
  `-dfi <dfiId>`
  `temip_sc_get_dfi -dfd <dfdName> -dfdv <dfdVersion>`
  `-mrp <mrpName>`

Parameters description

<*dfdName*> : Data Feeder name.

<*dfdVersion*> : Data Feeder version.

<*dfiId*> : Data Feeder Instance Identifier.

<*mrpName*> : Measurement Reference Point name.

4.2.5 temip_sc_get_expr

The **temip_sc_get_expr** command gets the specified calculation expression definition for the defined language (Java, PL/SQL...).

Command line syntax

```
temip_sc_get_expr -expr <ExprName> -language <language>
```

Parameters description

<*ExprName*> : expression name.

<*language*> : language of the expression. Value is either **Java** or **PL_SQL**.

4.2.6 temip_sc_get_sc

The **temip_sc_get_sc** command gets Service Component Definition.

Command line syntaxes

- Get the list of Service Components of a given Service Definition:
  ```
  temip_sc_get_sc -sd <sdName>
  ```
- Get the definition of a Service Component of a given Service Definition:
  ```
  temip_sc_get_sc -sd <sdName> -sc <scName>
  ```

Parameters description

<*sdName*> : Service Definition name.

<*scName*> : Service Component name.

4.2.7 temip_sc_get_sci

The **temip_sc_get_sci** command retrieves information about Service Component Instance.

Command line syntaxes

- Get a summary list of the Service Component Instances shared among various definitions:
  ```
  temip_sc_get_sci -sc <scName>
  ```
- Get a summary list of the Service Component Instances shared between instances of the same Service Definition:
  ```
  temip_sc_get_sci -sd <sdName> -sc <scName>
  ```
- Get the full definition of a Service Component Instance:
  ```
  temip_sc_get_sci -sd <sdName> -sc <scName>
                           -sci <sciName>
  ```
- Get the list of Service Component Instances linked to a given DFD:
  ```
  temip_sc_get_sci -dfd <dfdName>
  ```

Parameters description

<*sdName*> : Service Definition name.

<*scName*> : Service Component name.

<*sciName*> : Service Component Instance name.

<*dfdName*> : Data Feeder Definition name.

4.2.8 temip_sc_get_sd

The **temip_sc_get_sd** command retrieves information about Service Definitions registered in the SQM Service Repository.

Command line syntaxes

- Get a summary list of all Service Definitions:
  `temip_sc_get_sd`
- Get the full description of a given Service Definition:
  `temip_sc_get_sd –sd <sdName>`
- Get a summary list of the Service Definitions that use the given **shared** Service Component. If the specified Service Component is not shared between several Service Definitions the returned list is empty:
  `temip_sc_get_sd –sc <scName>`

Parameters description

<*sdName*> : Service Definition name.

<*scName*> : Name of the shared Service Component .

4.2.9 temip_sc_get_si

The **temip_sc_get_si** command retrieves information about Service Instance registered in the SQM Service Repository.

Command line syntaxes

- Get a summary list of the Service Instances of the given Service Definition:
  `temip_sc_get_si –sd <sdName>`
- Get the list of the Service Instance of the given Service Definition, with their Service Component Instances:
  `temip_sc_get_si –sd <sdName> -verbose`
- Get a full description of the given Service Instance of the given Service Definition:
  `temip_sc_get_si –sd <sdName> –si <siName>`
- Get the list of Service Instances that shared the given Service Component Instance. If the specified Service Component Instance is not shared between several Service Instances the returned list is empty:
  `temip_sc_get_si –sc <scName> –sci <sciName>`

Parameters description

<*sdName*> : Service Definition name.

<*scName*> : Service Component name.

<*siName*> : Service Instance name.

<*sciName*> : Name of the shared Service Component Instance.

4.2.10 temip_sc_get_sig

The **temip_sc_get_sig** command retrieves information about Service Instance Group.

Command line syntaxes

- Get the list of Service Instance Groups from the given Service Definition:
  `temip_sc_get_sig –sd <sdName>`
- Get the list of Service Instance Groups from the given Service Instance:

```
temip_sc_get_sig –sd <sdName> –si <siName>
```

- Get the definition of the given Service Instance Group:
```
temip_sc_get_sig –sd <sdName> –sig <sigName>
```

Parameters description

<*sdName*> : Service Definition name.

<*siName*> : Service Instance name.

<*sigName*> : Service Instance Group name.

4.2.11 temip_sc_get_sla

The **temip_sc_get_sla** command retrieves information about defined Service Level Agreement.

Command line syntaxes

- Get the list of Service Level Agreements defined for the given Service Definition:
```
temip_sc_get_sla –sd <sdName>
```
- Get a specific Service Level Agreements defined for the given Service Definition:
```
temip_sc_get_sla –sd <sdName> –sla <slaName>
```
- Get the list of Service Level Agreements defined for the given Service Instance:
```
temip_sc_get_sla –sd <sdName> –si <siName>
```
- Get the list of Service Level Agreements which use the specified Service Component Instance:
```
temip_sc_get_sla –sd <sdName> –sci <sciName>
```

Parameters description

<*sdName*> : Service Definition name.

<*slaName*> : Service Level Agreement name.

<*siName*> : Service Instance name.

<*sciName*> : Service Component Instance name

4.2.12 temip_sc_get_sl

The **temip_sc_get_sl** command retrieves information about defined Service Level.

Command line syntaxes

- Get a summary list of Service Levels defined for a given Service Definition:
```
temip_sc_get_sl –sd <sdName>
```
- Get the list of Service Levels defined for a given Service Definition with their full definition:
```
temip_sc_get_sl –sd <sdName> -verbose
```
- Get the full definition of a given Service Level:
```
temip_sc_get_sl –sd <sdName> –sl <slName>
```

Parameters description

<*sdName*> : Service Definition name.

<*slName*> : Service Level name.

4.2.13 temip_sc_get_model

The **temip_sc_get_model** command retrieves the SQM model stored into the SQM Service Repository Manager and exports this model as XML files.

Command line syntax

```
temip_sc_get_model –outputDir <directory>
```

Parameters description

<*directory*> : Root directory where the resulting XML files are created. The created subdirectories are: Customer, DFD, DeclareDFI, RegisterDFI, ExprDef, ServiceDef, SI, SIG, ServiceLevel, SLA.

4.3 State Management utilities

4.3.1 temip_sc_unlock_sla

The **temip_sc_unlock_sla** command unlocks a given SLA (Administrative state will be *Unlocked*) for the given Service Definition.

Command line syntax

```
temip_sc_unlock_sla –sd <sdName> -sla <slaName>
```

Parameters description

<*sdName*> : Service Definition name

<*slaName*> : Service Level Agreement name

4.3.2 temip_sc_lock_sla

The **temip_sc_lock_sla** command locks a given SLA (Administrative state will be *Locked*) for the given Service Definition.

Command line syntax

```
temip_sc_lock_sla –sd <sdName> -sla <slaName>
```

Parameters description

<*sdName*> : Service Definition name

<*slaName*> : Service Level Agreement name

4.3.3 temip_sc_unlock_si

The **temip_sc_unlock_si** command unlocks a given Service Instance (Administrative state will be *Unlocked*) for the given Service Definition.

Command line syntax

```
temip_sc_unlock_si –sd <sdName> -si <siName>
```

Parameters description

<*sdName*> : Service Definition name

<*siName*> : Service Instance name

4.3.4 temip_sc_lock_si

The **temip_sc_lock_si** command locks a given Service Instance (Administrative state will be *Locked*) for the given Service Definition.

Command line syntax

```
temip_sc_unlock_si –sd <sdName> -si <siName>
```

Parameters description

<*sdName*> : Service Definition name

<*siName*> : Service Instance name

4.3.5 temip_sc_unlock_dfi

The **temip_sc_unlock_dfi** command unlocks a given Data Feeder Instance (Administrative state will be *Unlocked*) for the given Data Feeder Definition.

Command line syntax

```
temip_sc_unlock_dfi –dfd <dfdName> -dfdv <dfdVersion>
                    -dfi <dfiId>
```

Parameters description

<*dfdName*> : Data Feeder name.

<*dfdVersion*> : Data Feeder version.

<*dfiId*> : Data Feeder Instance identifier.

4.3.6 temip_sc_lock_dfi

The **temip_sc_lock_dfi** command locks a given Data Feeder Instance (Administrative state will be *Locked*) for the given Data Feeder Definition.

Command line syntax

```
temip_sc_lock_dfi –dfd <dfdName> -dfdv <dfdVersion>
                  -dfi <dfiId>
```

Parameters description

<*dfdName*> : Data Feeder name.

<*dfdVersion*> : Data Feeder version.

<*dfiId*> : Data Feeder Instance identifier.

4.4 Delete utilities

4.4.1 temip_sc_delete_cust

The **temip_sc_delete_cust** command deletes a customer.

Command line syntaxes

- Delete the given Customer:
  ```
  temip_sc_delete_cust –cust <custName>
  ```
- Delete the given Customer even if dependencies are present:
  ```
  temip_sc_delete_cust –cust <custName> -force
  ```

Parameters description

<*custName*> : Customer name.

4.4.2 temip_sc_delete_dfd

The **temip_sc_delete_dfd** command deletes a Data Feeder Definition.

Command line syntaxes

- Delete the given Data Feeder Definition:
```
temip_sc_delete_dfd –dfd <dfdName> -dfdv <dfdVersion>
```
- Delete the given Data Feeder Definition even if dependencies are present:
```
temip_sc_delete_dfd –dfd <dfdName> -dfdv <dfdVersion>
                    -force
```

Parameters description

<*dfdName*> : Data Feeder name.

<*dfdVersion*> : Data Feeder version.

4.4.3 temip_sc_delete_dfi

The **temip_sc_delete_dfi** command deletes a Data Feeder Instance.

Command line syntaxes

- Delete the given Data Feeder Instance named by its *Id* or by its *MRP*
```
temip_sc_delete_dfi -dfd <dfdName> -dfdv <dfdVersion>
                    -dfi <dfiId>
temip_sc_delete_dfi -dfd <dfdName> -dfdv <dfdVersion>
                    -mrp <mrpName>
```
- Delete the given Data Feeder Instance even if dependencies are present:
```
temip_sc_delete_dfi -dfd <dfdName> -dfdv <dfdVersion>
                    -dfi <dfiId> -force
temip_sc_delete_dfi -dfd <dfdName> -dfdv <dfdVersion>
                    -mrp <mrpName> -force
```

Parameters description

<*dfdName*> : Data Feeder name.

<*dfdVersion*> : Data Feeder version.

<*dfiId*> : Data Feeder Instance identifier.

<*mrpName*> : Measurement Reference Point name.

4.4.4 temip_sc_delete_expr

The **temip_sc_delete_expr** command deletes the specified calculation expression definition for the defined language (Java, PL/SQL...).

Command line syntaxes

- Delete the given Expression:
```
temip_sc_delete_expr –expr <ExprName>
                     -language <language>
```
- Delete the given Expression even if dependencies are present:
```
temip_sc_delete_expr –expr <ExprName>
                     -language <language> -force
```

Parameters description

<*ExprName*> : expression name.

<*language*> : language of the expression. Value is either **Java** or **PL_SQL**.

4.4.5 temip_sc_delete_sd

The **temip_sc_delete_sd** command deletes a Service Definition.

Command line syntaxes

- Delete the given Service Definition:
  `temip_sc_delete_sd –sd <sdName>`
- Delete the given Service Definition even if dependencies are present:
  `temip_sc_delete_sd –sd <sdName> -force`

Parameters description

<*sdName*> : Service Definition name.

4.4.6 temip_sc_delete_si

The **temip_sc_delete_si** command deletes a Service Instance.

Command line syntaxes

- Delete the given Service Instance:
  `temip_sc_delete_si –sd <sdName> -si <siName>`
- Delete the given Service Instance even if dependencies are present:
  `temip_sc_delete_si –sd <sdName> -si <siName> -force`

Parameters description

<*sdName*> : Service Definition name.

<*siName*> : Service Instance name.

4.4.7 temip_sc_delete_sig

The **temip_sc_delete_sig** command deletes a Service Instance Group.

Command line syntaxes

- Delete the given Service Instance Group:
  `temip_sc_delete_sig –sd <sdName> -sig <sigName>`
- Delete the given Service Instance Group even if dependencies are present:
  `temip_sc_delete_sig –sd <sdName> -sig <sigName> -force`

Parameters description

<*sdName*> : Service Definition name.

<*sigName*> : Service Instance Group name.

4.4.8 temip_sc_delete_sl

The **temip_sc_delete_sl** command deletes a Service Level.

Command line syntaxes

- Delete the given Service Level for the given Service Definition:

```
temip_sc_delete_sl –sd <sdName> -sl <slName>
```

- Delete the given Service Level even if dependencies are present:
```
temip_sc_delete_sl –sd <sdName> -sl <slName> -force
```

Parameters description

<*sdName*> : Service Definition name.

<*slName*> : Service Level name.

4.4.9 temip_sc_delete_sla

The **temip_sc_delete_sla** command deletes a Service Level Agreement of a given Service Definition.

Command line syntax

```
temip_sc_delete_sla –sd <sdName> -sla <slaName>
```

Parameters description

<*sdName*> : Service Definition name.

<*slaName*> : Service Level Agreement name.

4.5 Subscriber/Customer mapping utility

The **temip_sc_ns_admin_tool.sh** command is dedicated to the management of the Naming Server (NS) configuration, and particularly its Naming Plan (the Naming Plan allows the mapping of subscriber Identifiers to customers).

A new naming plan for OpenView SQM can be provided using this tool.

Command line syntax

```
temip_sc_ns_admin_tool.sh
      -platform <platform>
      -director <director>
      -application <application>
      -action <action>
      <filename>
      [-reloadLater]
```

Parameters description

<*platform*>*:* the platform to which the application belongs.

<*director*>: the director to which the application belongs.

<*application*>: the Naming Server (NS) application.

<*action*>: action can be either:

- *ExportNamingPlan:*

Allows retrieving the NamingPlan (with full config) <*filename*> will contain the whole NS config.

- *ImportNamingPlan:*

Allows importing the NamingPlan (and full config) <*filename*> contains the NS config to import.

- *ImportCustomer:*

Allows importing a customer into the NamingPlan <*filename*> contains the customer definition.

Note

Use ***ImportCustomer*** only when current loaded NamingPlan contains few customers. We recommend using ***ExportNamingPlan***/***ImportNamingPlan*** when dealing with heavy NamingPlan.

The <*filename*> used for ***ImportCustomer*** option must contain the Naming Plan only.

[-reloadLater]: indicates that the new Naming Plan has NOT to be taken into account now by the Naming Server.
Used only for ***ImportNamingPlan*** and ***ImportCustomer***.
The TIBCO Hawk Display allows, at any time, to ask the NS to reload its config.

Note

ImportCustomer allows importing a single customer from <*filename*>. This should be used only when dealing with a small NamingPlan; otherwise the operation will take too much time.

When dealing with a heavy NamingPlan, use ***ExportNamingPlan***/***ImportNamingPlan*** <*filename*> options.
The Export option will retrieve the full NS config into <*filename*>.
It is then possible to modify the NamingPlan: add new customers, modify existing ones, deleting some.

Once updated, use Import option with updated file in order to update NS NamingPlan.

Note that before importing the modified NamingPlan from <*filename*> a copy of the currently loaded NamingPlan will be generated in */tmp/SavedNsConfig###*.
You can use *-reloadLater* if you do not want to reload the NamingPlan in the NS just after the import operation. This could be done later using the *reloadConfig* TIBCO Hawk AMI on NS micro agent (using the TIBCO Hawk display for example).
Note that an AMI timeout message can be displayed on a reload operation when dealing with a heavy NamingPlan. Be aware that in spite of this message, the reload operation is not canceled. In such a case, wait a while and then use the Export option to check the NamingPlan.

4.6 Basic Messaging utilities

The message utilities are dedicated to send or listen to SQM XML messages directly onto the SQM buses.
There are three tools:

- Listen
- Publish
- Request/Reply

All the SQM messages are described in the *$TEMIP_SC_HOME/etc/scmsg_descr.txt* file under the following format:

Version **;** Msg Id **;** Certified/Not Certified(2/1) **;** MsgId (AlphaNumeric) **;** BUS **;** Fixed part of the Subject **;** unused ; DTD of the xml message **;** list of variable parts

SQM messages can be specified by the –*t* option either using the Msg Id in numeric or alpha numeric format (-*t 52* to listen/send RAW messages for example).

Common parameters

-*h*|*help*	:	to display the command line syntax
-*version*	:	display tools and versions

Optional parameters

-*timeout* < *value*>:	set timeout in second for the call
-*use/u* <*Configuration File*>:	specify a configuration file. By default, the configuration file of the current platform is taken.
-service <service>:	indicates the name of the service used to open a TIBCO connection. This option must not be set if the option <-use> is used.
-*network* <*network*>:	indicates the network string value used to open a TIBCO connection. This option must not be set if the option <-use> is used.
-*daemon* <*daemon*>:	indicates the daemon value used to open a TIBCOconnection. This option must not be set if the option <-use> is used.

4.6.1 temip_sc_listen

The **temip_sc_listen** command listens specific message(s) on a given TIBCO bus. The listened subject is given as arguments of the command. If the option - *type* is present, the listened subject is composed with the fixed part of the message subject and with the part given by the option - *subject*. Messages received by the command are printed in the standard output or in the output file depending on whether the option -*output* is present or not. If the options -*number* and -*timeout* are not present in the command, the listen command will be infinite on the bus until the end-user stops the command using the <CTRL-C> key sequence.

Command line syntax

```
temip_sc_listen
            [-help/-h] [-version/-v]
            -type <message_type>
            [-subject/-s <subject>] [-output/-o <filename>]
            {[-use/-u <filename>] | [-service <service>]
                                   [-network <network>]
                                   [-daemon <daemon>]
            }
            [-count/-c <count>]
            [-timeout <timeout(seconds)>]
            [-cmname <cmname>]
            [-cmledger <cmledger>] [-cmcancel]
            [-xmlbodyonly/-x] [-pretty/p] [-debug]
```

Parameters description

-*type/t* <*message_type*>:	Indicates the type of the message (see the table in the next section). If this option is set, the listened subject is composed with the fixed part of the message subject concatenated with the part given by the option–*subject.*
-*output/-o* <*filename*>:	Indicates the output file where the result of the command will be written. Only the result of the command will be

	written to the file. Errors will be printed on the standard output. The file must be expressed as an absolute path.
-subject/-s <subject>:	Indicates the subject for which messages are listened on the bus. Wildcards characters (*, >) can be used in this field. The character "*" replaces a part of the subject. The character “>” replaces the trailing part of the subject.
-count/-c <count>:	Indicates the number of messages to be listened for. The command waits for the given number of messages. When this number is reached, the tool will be stopped automatically.
-timeout <timeout(seconds)>:	Indicates the number of seconds during which the listen command will be active. When this timeout is reached, the tool will be stopped automatically.
-cmname <cmname>:	Indicates the name of the channel used for certified messaging mode. This option is mandatory when other certified messaging mode options are used (*-cmledger* and *-cmcancel*).
-cmledger <cmledger>:	Indicates localization of the ledger file used to store certified messaging events. This option is mandatory when certified messaging mode is activated (when *-cmname* option is used).
-cmcancel:	Cancels all certified delivery agreements of this listener. Certified senders delete from their ledgers all messages sent to this listener.
-xmlbodyonly/-x:	Receives only the xml part of the message.
-pretty/-p:	Receives only the pretty part of the message not the xml part.
-debug:	Print traces while during execution.

Examples

```
temip_sc_listen -u scplatform_tibrv_config.txt -t SCG

temip_sc_listen -u scplatform_tibrv_config.txt -t SCG \
      -s Video

temip_sc_listen -u scplatform_tibrv_config.txt -t SCG \
      -s Video -timeout 100

temip_sc_listen -daemon 7474 -network ";" -service 9950 \
      -t SDG -s Video -c 50

temip_sc_listen -u scplatform_tibrv_config.txt -t SDG \
      -s Video -c 2 -timeout 100 -o log-SDG.txt

temip_sc_listen -u scplatform_tibrv_config.txt -t SDG \
      -s Video -o log-SCG.txt

temip_sc_listen -t SIU -s Video.VideoParis -u \
      scplatform_tibrv_config.txt -c 1

temip_sc_listen -s SC.SI.Update.Video.VideoParis -daemon \
      7474 -network ";224.2.0.1" -service 9950 -c 1

temip_sc_listen -t SIU -s Video.VideoParis -u \
```

```
      scplatform_tibrv_config.txt -c 10 -cmname namel1 \
      -cmledger ledger/ledgerl1.ldg

temip_sc_listen -t SIU -s Video.VideoParis -u \
      scplatform_tibrv_config.txt -cmname namel1 \
      -cmledger ledger/ledgerl1.ldg -cmcancel
```

The following examples return 0 if the listen is able to receive the number of messages mentioned by *-c* option within the period given by the *-timeout* option.
If the listen does not receive any message within the timeout, then the return code is 1.
If the listen receives one or more messages but less than the number specified by *–count* option, then the return code is 2.

```
temip_sc_listen -t SDG -s Video \
                -c 5 -timeout 30 -o log-SDG.txt -p -x

temip_sc_listen -t SDG -s Video \
                -c 5 -timeout 30 -o log-SDG.txt -p

temip_sc_listen -t SDG -s Video \
                -c 5 -timeout 30 -o log-SDG.txt -x
```

4.6.2 temip_sc_publish

The **temip_sc_publish** command sends a message on a specific TIBCO bus. The message that will be sent on the bus can be generated automatically by the command or can be provided in an XML file. The command has a base of generic messages for all generated messages. Those generic messages can contain parameters that will be substituted when the message will be sent. The *argument* option of the command allows you to pass those parameters.

Command line syntax

```
temip_sc_publish
            [-help/-h] [-version/-v]
            -type/-t <message_type>
            [-input/-i <xml_input_file>]
            [-output/-o <filename>]
            [-subject/-s <subject>]
            {[-use/-u <filename>] | [-service <service>]
                                   [-network <network>]
                                   [-daemon <daemon>]}
            [-cmname <cmname>] [-cmledger <cmledger>]
            [-cmreview <number of subjects>]
            [-count/-c <count>]
            [-delay/-d <delay(in seconds)>]
```

Parameters description

-type/-t <message_type>:	Indicates the type of the message. This option is mandatory.
-input/-i <xml_input_file> :	Indicates the XML file which contains the message to send. The file must be expressed as an absolute path.
-output/-o <filename>:	Indicates the output file where the result of the command will be stored. Only the result of the command will be written on the file. Errors will be printed on the standard output. The file must be expressed as an absolute path.

-subject/-s <subject> :	Indicates the variable part of the subject for the message to send. The variable part of the subject must not begin with a dot character.
-count/-c <count> :	Indicates the number of messages to be listened for. The command waits for a given number of messages. When this number is reached, the tool will be stopped automatically.
-delay/-d <delay> :	Delay in seconds between the successive messages published.
-cmname <cmname> :	Indicates the name of the channel used for certified messaging mode. This option is mandatory when other certified messaging mode options are used (*-cmledger* and *-cmcancel*).
-cmledger <cmledger> :	Indicates localization of the ledger file used to store certified messaging events. This option is mandatory when certified messaging mode is activated (when *-cmname* option is used).
-cmreview <number of subjects>:	Indicates that the publisher will not publish messages but only review the ledger file provided with *-cmledger* option. Argument *<number of subjects>* gives the maximum number of subjects reviewed.

Examples

```
temip_sc_publish  -t SCG -s Video \
            -i SC.ServiceDef.SC.GetReq.Video.xml

temip_sc_publish -daemon 7474 -network ";224.2.0.1" \
            -service 7865 -t SDG -s Video \
            -i SC.ServiceDef.SC.GetReq.Video.xml

temip_sc_publish -t SDG -s Video \
            -i SC.ServiceDef.SC.GetReq.Video.xml \
            -o log-SDG.txt

temip_sc_publish -t SCG -s Video -o log-SCG.txt

temip_sc_publish -t SIU -s Video.VideoParis

temip_sc_publish -t SIU -s Video.VideoParis –i\
      datafile/snd/SC.SI/SC.SI.Update.Video.Paris.xml  \
            -c 5 -d 6

temip_sc_publish -t SIU -s Video.VideoParis –i\
      datafile/snd/SC.SI/SC.SI.Update.Video.Paris.xml \
            -c 10 -d 1 -cmname namep1 \
            -cmledger ledger/ledgerp1.ldg

temip_sc_publish -t SIU -s 'Video.>'  \
            -cmname namep1 -cmledger ledger/ledgerp1.ldg \
            -cmreview 10

temip_sc_publish -t SIU -s Video.VideoParis -i\
      datafile/snd/SC.SI/SC.SI.Update.Video.Paris.xml \
            -c 10 -d 1 -cmname namep1 \
            –cmledgerledger/ledgerp1.ldg
```

4.6.3 temip_sc_request

The **temip_sc_request** command sends a request/reply message. Compare to a publish message, the command waits for the answer. If no timeout is specified, the command waits forever.
The message that will be sent on the bus can be generated automatically by the command or can be provided in an XML file. The command has a base of generic messages for all generated messages. Those generic messages contain parameters that will be substituted when the message is sent. The argument option of the command allows you to pass those parameters.

Command line syntax

```
temip_sc_request
            [-help/-h] [-version/-v]
            -type/-t <message_type>
            [-input <xml_input_file>] [-output <filename>]
            [-subject/-s <subject>]
            [-timeout <timeout(seconds)>]
            [-displayresponseonly]
            [-count/-c <count>]
            {[-use/-u <filename>] | [-service <service>]
                                   [-network <network>]
                                   [-daemon <daemon>]
            }
```

Parameters description

-type/-t <message_type> :	Indicates the type of the message. This option is mandatory.
-input/-i <xml_input_file> :	Indicates the XML file which contains the message to send. The file must be expressed as an absolute path.
-output/-o <filename> :	Indicates the output file where the result of the command will be written. Only the result of the command will be written to the file. Errors will be printed on the standard output. The file must be expressed as an absolute path.
-subject <subject>:	Indicates the variable part of the subject for the message to send. The variable part of the subject must not begin with a dot character.
-displayresponseonly	Displays only the response output.
-count/-c <count> :	Indicates the number of messages to be listened. The command waits for a given number of messages. When this number is reached, the tool will be stopped automatically.
-timeout <timeout(seconds)>:	Indicates the number of seconds during which the command will be active. When this timeout is reached, the tool will be stopped automatically.

Examples

```
temip_sc_request -t SDG -s Video

temip_sc_request -t SCG -s Video \
       -u scplatform_tibrv_config.txt\
       -i SC.ServiceDef.SC.GetReq.Video.xml

temip_sc_request -daemon 9600 -network ";224.2.0.1" \
       -service 9950 \
```

```
       -t SDG -p Video \
       -i SC.ServiceDef.SC.GetReq.Video.xml

temip_sc_request -t SDG -s Video \
       -i SC.ServiceDef.SC.GetReq.Video.xml -o log-SDG.txt

temip_sc_request -t SCG -s Video \
       -o log-SCG.txt

temip_sc_request -t SDG -s Video \
      -i SC.ServiceDef.GetReq.Video.KO.xml -timeout 40

temip_sc_request -t SDG -s Video \
      -i SC.ServiceDef.GetReq.Video.KO.xml -timeout 40 -c 2

temip_sc_request -t SDG -s Video \
      -i SC.ServiceDef.GetReq.Video.KO.xml \
      -displayresponseonly
```

Chapter 5

Troubleshooting Guide

5.1 SQM platform troubleshooting

5.1.1 Alert logging

To view a log, or error information, you can use the **TIBCO Hawk Display** tool.

The **TIBCO Hawk Display** GUI displays in real-time Alerts coming from any SQM components.
The error files are located in `$TEMIP_SC_VAR_HOME/log` directory.

For each application, a microagent monitors the slmv12_*slmonitoring_ApplicationName.log* file, and when a new error is detected in this file, the microagent sends a message to the **TIBCO Hawk Display** (if it is started) including some information about the error.

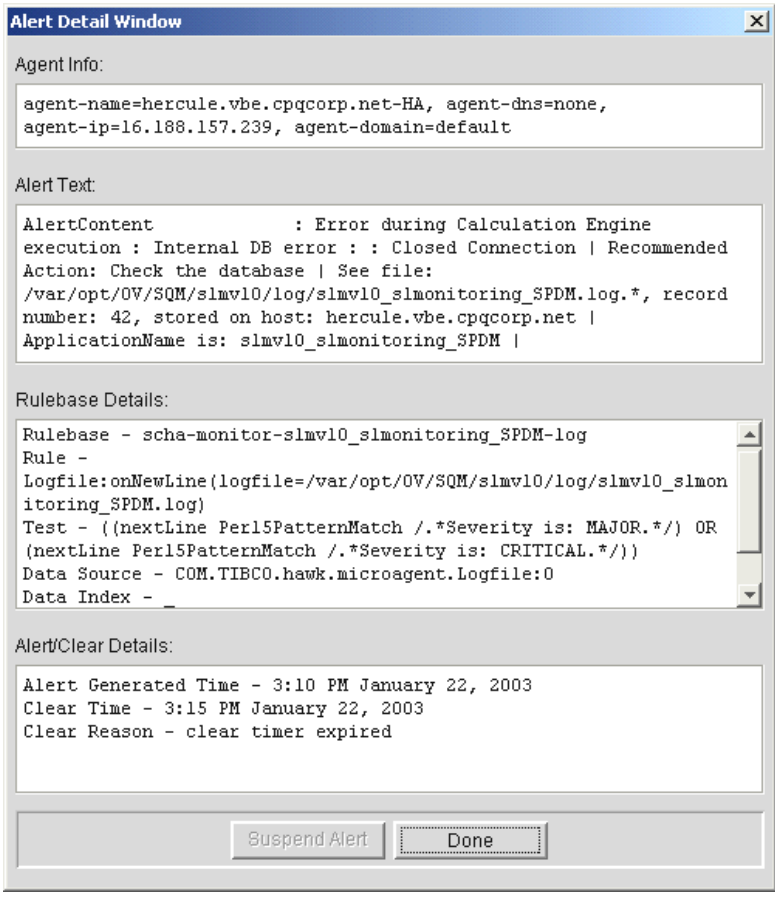
The **TIBCO Hawk Display** does not keep a history of errors. If you want to see the history of errors, you need to use a UNIX console and display the *$TEMIP_SC_VAR_HOME/log/slmv12_slmonitoring_ApplicationName.log.number* file.

Note

To manage the log files, the **fileCount** and the **fileLimit** parameters can be configured. The first log file is named xxx.log.number, where *number* starts at 1. Each time a log file reaches the **fileLimit** size, *number* is incremented and a new log file is created. When *number* exceeds **FileCount**, it is reset to 1. You can configure these parameters through the Service Designer.

The **TIBCO Hawk Display** can be used to view the list of alerts for an agent. Right clicking on the platform agent brings up a submenu containing the **Show Alerts** item. This item displays in real-time the list of alerts and double-clicking on a row in this list will display the details about the selected alert (see Figure 30).

Figure 30: TIBCO Hawk Display Alert Details Window



Information displayed in the **Alert Detail Window** is obtained from the corresponding *$TEMIP_SC_VAR_HOME/log/slmv12_slmonitoring_SPDM.log* error file:

```
>>>>>>>>>>>>>>> <MTLogRecord>
TimeStamp                 : Wednesday, January 22, 2003
2:10:07 PM GMT+00:00
ApplicationName           : slmv10_slmonitoring_SPDM
PackageName               : core.spdm.process
RecordNumber              : 42
ThreadID                  : 4
HostName                  : hercule.vbe.cpqcorp.net
UserName                  : sqmadm
Severity                  : MAJOR
Message                   : Error during Calculation Engine
execution : Internal DB error : : Closed Connection
LogCode                   : Log_ErrorDuringEngineExecution
LogType                   : Calculation Engine execution
ProbableCause             : Database problem
RecommendedAction         : Check the database
AdditionalInfo            :
AlertContent              : Error during Calculation Engine
execution : Internal DB error : : Closed Connection |
Recommended Action: Check the database | See file:
/var/opt/OV/SQM/slmv10/log/slmv10_slmonitoring_SPDM.log.*,
record number: 42, stored on host: hercule.vbe.cpqcorp.net |
ApplicationName is: slmv10_slmonitoring_SPDM | Severity is:
MAJOR
MsgParameters             : Internal DB error : : Closed
Connection
>>>>>>>>>>>>>>> </MTLogRecord>
```

Description of the main fields composing the message

- **Timestamp**: time at which error has occurred.
- **ApplicationName**: application that has logged the error.
- **PackageName**: name of the java package where the error has occurred.
- **RecordNumber**: the error number to retrieve the error in the Log file.
- **ThreadId**: id of the thread that has logged the error.
- **HostName**: name of the host on which the application was running.
- **Severity**: either Critical, Major, Minor, Clear, Indeterminate or Warning.
- **Message**: short description of the error.
- **ProbableCause**: short description of the probable cause of the error.
- **RecommendedAction**: what can be done to avoid the problem.
- **AdditionalInfo**: contains more details about the cause of the problem and the recommended action.

5.1.1.1 Probable Cause

You can find below the list of probable causes that are currently defined on the OpenView SQM platform:

- **Message Reception Error**: bad reception of an incoming message.
- **Message Decoding Error**: Problem decoding an incoming message, validation error (syntax error).
- **Message Incorrect Information**: expected parameter is missing in a message or invalid input data (semantic error).
- **Message publishing Error**: problem when trying to publish a message.
- **Model Inconsistent Information**: inconsistent model despite SRM validation checks.
- **DB Access Failure**: problem when connecting or trying to access Database Information.
- **DB Inconsistent Information**: problem when trying to access information in Database because the data is not present, or inconsistent.
- **DB Processing Error**: problem when trying to perform a SQL query on DB.
- **Application Initialization Error**: problem when starting the application.
- **Application Configuration Error**: problem when trying to read a configuration file, or the configuration registered by the Central Repository.
- **Application Processing Error**: a process thread has stopped, or any application processing error.
- **Application Communication Error**: a contacted application does not respond.
- **Unknown**: should be used only when it is an unexpected or undefined problem.

5.1.2 Traces logging

Trace files are located in the *$TEMIP_SC_VAR_HOME/trace* directory. They are used for debugging purposes. The name and file structure used for trace files is identical to those used for error files. By default, components traces are set to OFF, which means that no traces are generated.

Note

To manage the trace files, the **fileCount** and the **fileLimit** parameters can be configured. The first trace file is named xxx.log.number, where *number* starts at 1. Each time a trace file reaches the **fileLimit** size, *number* is incremented and a new trace file is created. When number exceeds **FileCount**, it is reset to 1. You can configure these parameters through the Service Designer.

Each microagent owns the **setTraceLogLevel** and **getTraceLogLevel** methods, which allow the setting and display of the level of the trace (see below).

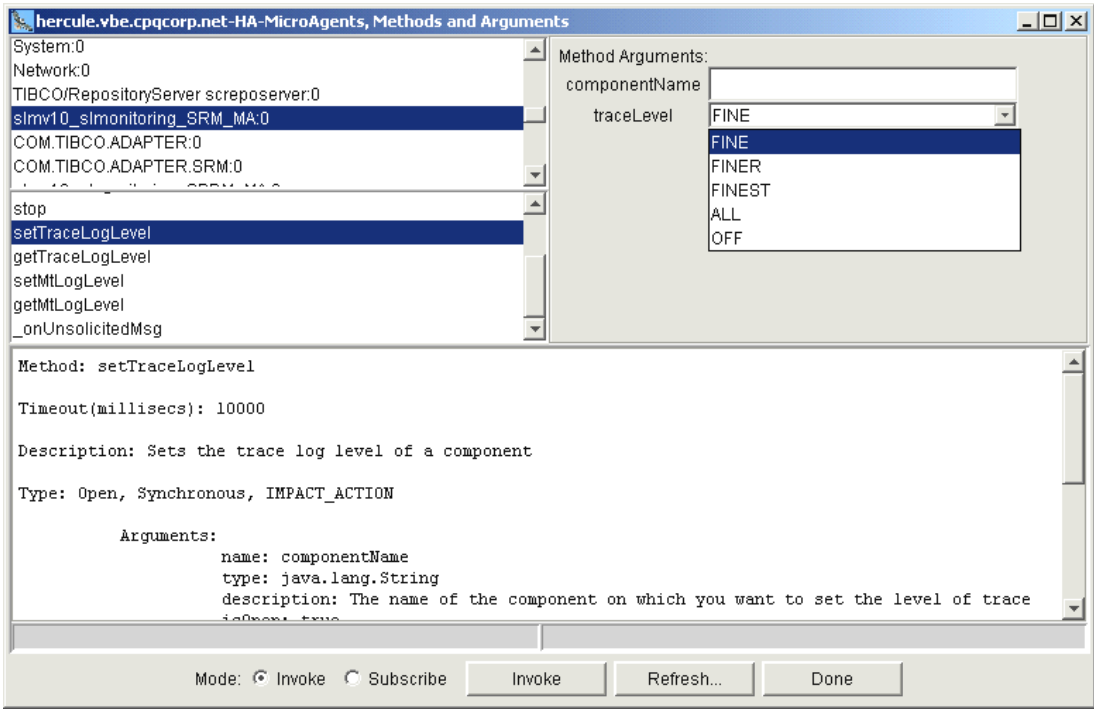
5.1.2.1 How to set traces

The SQM Admin console (see paragraph Graphical OpenView SQM Admin Console) can be used to invoke the AMI **`setTraceLogLevel`**. This method allows setting a specific trace level (see Figure 31):

- OFF
- FINE
- FINER
- FINEST
- ALL

In ascending order of detail

Figure 31: setTraceLogLevel method on a SQM component



The use of the **Invoke** button sets the defined trace level on the selected component. Traces are stored in the *$TEMIP_SC_VAR_HOME/slmv12_slmonitoring_SRM.log* file.

Note

The trace level is taken into account while the application is running. Once it has been stopped, you need to redo the operation, since by default trace level is set to OFF at application start.

You can also set the trace level using the command line either with `temip_sc_selfmgmt` (see section 2.2.6), or with the following command:

`$TEMIP_SC_HOME/support/sqm_set_trace.ksh <ApplicationName> [<TraceLevel> [<Component>]]`

Where:

<ApplicationName>	is the name of the application,
<TraceLevel>	is the level of trace desired (default=ALL)
<Component>	is sub component to be traced (optional).

Note

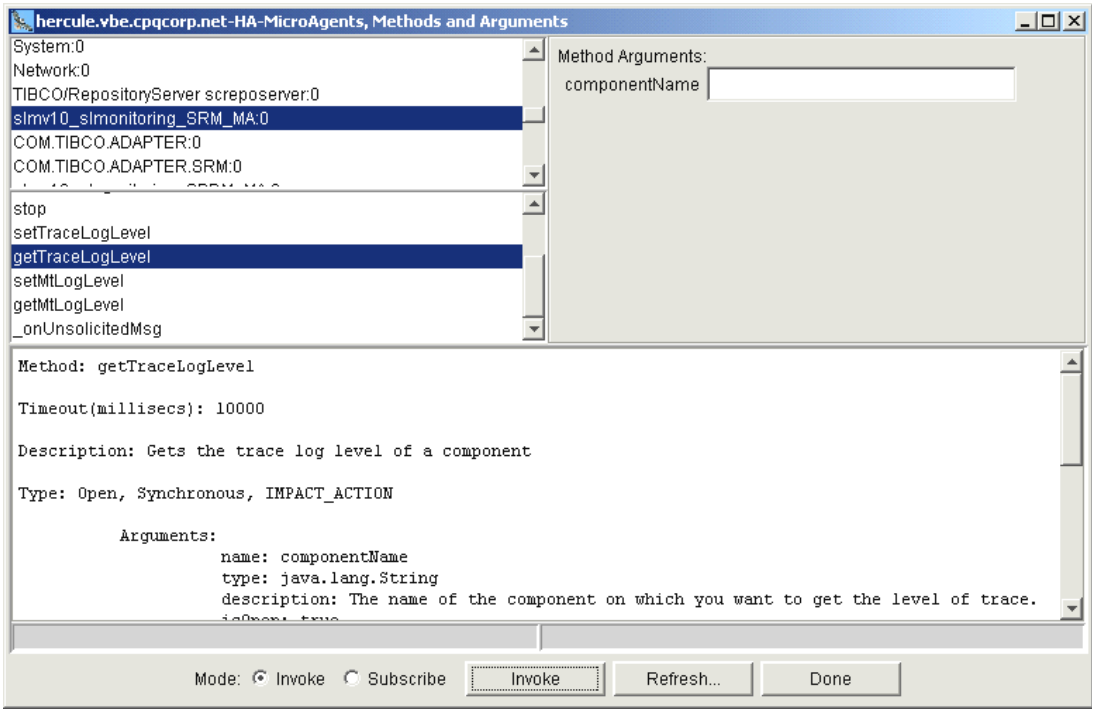
To trace all XML buffers sent and received by a SQM application in the folder $TEMIP_SC_HOME/messages/<platform>_<direction>_<application>/msgdebug, then set the trace level FINEST on the component "com.compaq.temip.servicecenter.messaging" of the application.

This setting activates the tracing of the XML buffers without the traces on the rest of the application.

5.1.2.2 How to get the current trace level

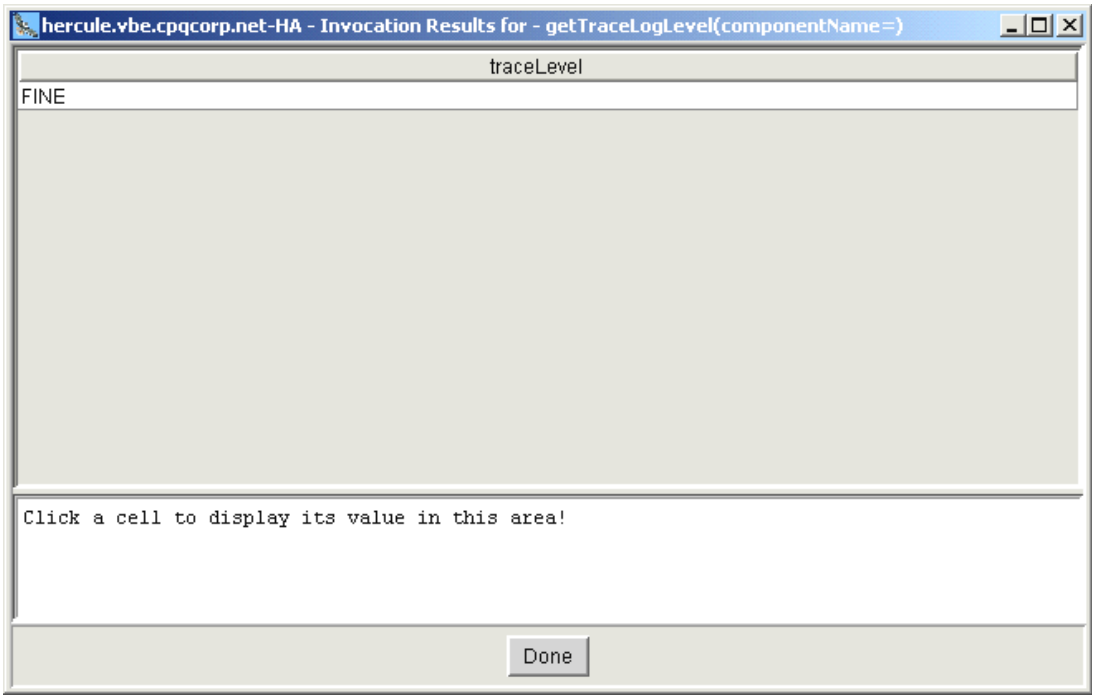
The SQM Admin console (see paragraph Graphical OpenView SQM Admin Console) can be used to invoke the AMI **`getTraceLogLevel`**. This method allows displaying the current trace level (see Figure 32 and Figure 33).

Figure 32: getTraceLogLevel method on a SQM component



By clicking on the **Invoke** button, you get the following dialog box:

Figure 33: getTraceLogLevel output on a SQM component



You can also get the trace level using the command line either with `temip_sc_selfmgmt` (see section 2.2.6), or with the following command:

`$TEMIP_SC_HOME/support/sqm_get_trace.ksh <ApplicationName> [<Component>]`

Where:

<ApplicationName>	is the name of the application,
<Component>	is the sub component to be retrieved (optional).

5.2 SQM components troubleshooting

5.2.1 Traces and logs

Troubleshooting in OpenView SQM is achieved using tracing capabilities: execution tracing and error logging. Debug and error messages are both saved in files. Different levels of trace allow specific troubleshooting. They are configurable at run-time from the **TIBCO Hawk Display**. They are performed at both client and server levels. The format and configuration of the tracing facility is similar in all the environments (client application, UI Server or Core, UNIX or Windows).

5.2.2 Dump facility

In case of problem, the SQM Support may ask you to invoke the dump AMI command on a specific OpenView SQM component (SRM, SPDM, User Interface Server, etc.).

- This facility may be invoked from the SQM Admin console (see paragraph Graphical OpenView SQM Admin Console). The dump is an exhaustive log of internal data. The dump results are saved to a specific file in `$TEMIP_SC_VAR_HOME/trace`

directory even if the traces are set to OFF position. The name of this file is displayed by the SQM Admin console as result of the dump AMI invocation.

- This facility, may also be invoked from the UNIX command line, with command:

  `$TEMIP_SC_HOME/support/sqm_dump.ksh <ApplicationName> [<DumpMode>]`

  Where:

  <*ApplicationName*> is the name of the application to dump,

  <*DumpMode* > is the dump mode (e.g. Memory, Topics...). The default is All. The first letter of the standard dump modes can be used in place of the complete mode.

5.2.3 SQM components specific troubleshooting

This paragraph describes additional specific methods that may be used in a first approach, to check the health of each of the SQM Components.

5.2.3.1 Service Repository Manager

Diagnostic

To check whether the Service Repository Manager is alive and working properly, use the ***temip_sc_get_sd*** command. It will get the summary of all the service definitions.

5.2.3.2 Service Performance Data Manager

Diagnostic

To check whether the Service Performance Data Manager is alive and working properly, you have to create a file named *publish_msg_56.xml* with the following content:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE sc:SCIPrimPerfMeasure SYSTEM
"DTD/tsc_PerformanceDataEvt.dtd">
<sc:SCIPrimPerfMeasure  granularity="300" msg.id="56"
  sd.name="Video"  scd.name="PlatformV"
sci.name="platform-CamWeb"  rootComponent.flag="False"
  suspect.flag="False"
xmlns:sc="http://www.compaq.com/TeMIP/ServiceCenter">
  <sc:Measure timeStamp="2003-02-14T14:04:12.000">
    <sc:ParameterValue datatype="Int"
parameter.name="CPULoad"
noValue="False">52</sc:ParameterValue>
  </sc:Measure>
</sc:SCIPrimPerfMeasure>
```

- Do not forget to update the timestamp. The timestamp must be set to the current time (near future).
- From a shell, use the following command:

  ```
  temip_sc_listen –t 57
  ```

- From another shell, use the following command:

  ```
  temip_sc_publish -t 56 -subject
  Video.NoCust.NoSI.PlatformV.platform-CamWeb -i
  publish_msg_56.xml
  ```

The temip_sc_listen command will provide an output value corresponding to the AutoForwarded value.

5.2.3.3 Data Collector

Diagnostic

To check whether the Data Collector is alive and working properly, you have to listen to messages on the *PrimaryData* bus, by using the following command:

```
temip_sc_listen –t SIPM
```

5.2.3.4 Naming Service

Diagnostic

To check whether the Naming Service is alive and working properly, you have to request a subscriber validation to the Naming Service.

Choose a subscriber and domain that belong to your Naming Plan. In our example we will use domain=*domain2* and subscriber=*sub1_1*

Enter the command:

```
temip_sc_request \
-t NSR \
-a sc:Subscribers="{
        'sc:Subscriber'     =>  {
        'subscriber.domain' => 'Domain2',
        'subscriber.name'   => 'sub1_1'
                                }
                    }"
```

You should receive an answer from the Naming Service component that should validate the subscriber (or not depending of the input you gave).

When it is validated, you receive the following kind of answer:

```
<sc:ResolveSubscriberReply
xmlns:sc="http://www.compaq.com/TeMIP/ServiceCenter"
msg.id="2">
 <sc:SubscriberCustomerAssocs>
   <sc:SubscriberCustomerAssoc subscriber.name="sub1_1"
        subscriber.domain="Domain2"
       customer.name="C1"
      static.flag="True"/>
</sc:SubscriberCustomerAssocs>
</sc:ResolveSubscriberReply>
```

5.2.3.5 Service Level Objective Manager

5.2.3.6 Logger

Diagnostic

To check whether Logger is alive and working properly, you can try to publish a customer update message and try to see if the message is logged into the Logger DB.

First, count the number of messages already stored in the Logger DB:

```
export ORACLE_SID=logger
sqlplus logger/logger
select count(*) from logrecord;
exit
```

Publish a Customer Update (message 80):

```
temip_sc_publish –t CUSTU –s customerTest
```

Wait `DBMaxNbSecondsbetweenTransactions,` then reconnect to the DB and count the number of records again.

It should be incremented of one.

5.2.3.7 User Interface Server

Diagnostic

Launch a SLA Monitoring and try to connect to the User Interface Server.

In case of failure

On the presentation director host, execute the command:

```
ps -edfx | grep orb
```

You result should contains 3 processes: `itlocator`, `itnode_daemon` and `itnaming`.

If the previous command does not list these three processes, this means that the Orbix services are not started.

➔ Log as root, source the SQM environment and execute the command

```
$TEMIP_SC_VAR_HOME/etc/start_orbix_tsc_services
```

5.2.3.8 SLA Monitoring UI

Diagnostic

Launch a SLA Monitoring and try to connect to the User Interface Server.

How to trace in SLA Monitoring UI

It is possible to trace the activity of the SLA Monitoring UI for troubleshooting purposes.

Traces have to be set ON-line by updating a configuration file and any update of this configuration file is taken into account only at next UI restart.

The trace configuration file

The SLA Monitoring UI trace configuration file is located in the client configuration directory: **`%USERPROFILE%/HP OpenView/SQM/UI/SLMClient/config`**.

It is named **`TeSCUISLMClientTrace.properties`**.

Note

`USERPROFILE` is a Windows environment variable. To know its value, type in a DOS Console:

```
set USERPROFILE
```

How to choose the information to trace?

Different subparts of traces can be activated, by un-commenting the corresponding lines in the trace configuration file:

- XML buffers exchanged between the UI server and the Monitoring UI
- Debug (Audit) traces
- …

Note

Do not activate all traces at the same time on the SLA Monitoring. A huge quantity of trace may be generated. Activate only the mandatory components. Please refer to *SQM Support* to know the components to activate regarding your problems.

How to select the level of trace?

For each component, a level can be chosen between:

- OFF: no traces
- FINE: intermediate level 1
- FINER: intermediate level 2
- FINEST (or ALL): most detailed level

Where are the traces files located?

Traces are generated in the directory **`%USERPROFILE%\HP OpenView\SQM\trace`**.

This location can be modified in the trace configuration file. Trace files names are generated for a specific process. A new set of traces files is created every time you restart the SLA Monitoring UI.

The SLA Monitoring UI trace file names start with **`TeSCUISLMonitoringClient_Trace.`**

5.2.3.9 SLA Administration UI

Diagnostic

In case the SLA Administration UI has to work “Online” with the Service Repository Manager, launch a SLA Administration UI and and try to connect to the User Interface Server. Check the connection message in the SLA Administration UI console window. The message should be “Application online and connected to the server”.

How to trace in SLA Administration UI

Traces are set in the same way as for the SLA Monitoring UI (refer to 5.2.3.8) with specific location of files :

- The SLA Administration trace configuration file is located in the client configuration directory: **`%USERPROFILE%/HP OpenView/SQM/UI/SLAClient/config`**. It is named **`TeSCUISLAClientTrace.properties`**.
- Traces are generated in the directory **`%USERPROFILE%\HP OpenView\SQM\trace`**.
- SLA Administration trace files names start with **`TeSCUISLAdminClient_Trace`**

5.2.3.10 Service Adapters

Diagnostic

To check whether a Service Adapter instance is alive and working properly, use the command:

```
temip_sc_listen –t 52
```

5.2.3.11 Gateways

Diagnostic

To check whether a Gateway is alive and working properly, use the:

- The Dump AMI directive and check the output, or set the trace level
- Simulate a parameter ThresholdCrossed to which has been associated an Action Executor and check that the action has been executed correctly (e.g. alarm creation…).

5.3 How to detect if network support multicast or broadcast IP for TIBCO Rendezvous?

If the distributed SQM Components do not actually communicate between them, it could be due to the Network parameter you have chosen during the SQM Setup. Remember that this parameter specifies if the IP transport used by the TIBCO Rendezvous buses is either a broadcast IP (Network is "`;`") or a multicast IP (Network is "`;224.2.0.X`"). If this parameter does not match your physical network capabilities, TIBCO Rendezvous will not work correctly and consequently SQM execution will fail.

You can verify if your physical network supports broadcast IP or multicast IP between two system boxes that host the SQM Platform, or none of these two cases by using a ***`tibrvlisten`*** on the local host and a ***`tibrvsend`*** on the remote host.

1. Source the TIBCO variables in your environment on both system boxes:
   a. If you have a kernel already setup on your machine you have to source
      - On UNIX:

        ```
        $TEMIP_SC_VAR_HOME/temip_sc_env.sh
        ```

      - On WINDOWS:

        ```
        %TEMIP_SC_VAR_HOME%/temip_sc_env.bat
        ```

   b. Or if you have not setup a kernel yet:
      - On UNIX:

        Set your TEMIP_SC_HOME variable (for instance, `TEMIP_SC_HOME=/opt/OV/SQM`)

        ```
        $TEMIP_SC_HOME/tibco/tibco-setup.sh
        ```

      - On WINDOWS:

        Set your TEMIP_SC_HOME variable (for instance by default, `set TEMIP_SC_HOME=C:\Program Files\HP Openview\SQM`

        ```
        call "%TEMIP_SC_HOME%\tibco\tibco-setup.bat"
        ```

2. Check if IP broadcast is supported:
   a. Onto the first system box, do a:

      ```
      tibrvlisten -daemon "tcp:<YourRVDport>" \
      network ";"  -service 22222 FOO
      ```

   b. On the second system box, do a:

      ```
      tibrvsend -daemon "tcp:<YourRVDport>" \
      -network ";" -service 22222 FOO foo
      ```

   If `tibrvlisten` receives the message FOO, broadcast IP is supported between these two machines, otherwise, let try the multicast IP (see 3/)

3. Check if multicast IP is supported:

   c. On the first system box, do a (do not use the same service number as for 2/):

```
tibrvlisten -daemon "tcp:<YourRVDport>" \
-network ";224.2.0.1" -service 33333 FOO
```

   d. On the second system box, do a:

```
tibrvsend -daemon "tcp:<YourRVDport>" \
-network ";224.2.0.1" -service 33333 FOO foo
```

   If `tibrvlisten` receives the message FOO, multicast IP is supported between these two machines.

If you want to change the Network parameter, which have been setup on your SQM platform you have to setup again the platform and chose the right value.

If none of these two alternatives is supported by your network, you will need to configure a **RVRD** between your two machines (see section RVRD), or possibly a direct connection to a remote **RVD**: to connect to a remote RVD directly, the RV Daemon parameters must specify the full hostname of the remote RVD. For instance "*`tcp:myFullHostname:11270`*" rather than the usual RV Daemon parameter "`tcp:11270`".

Refer to the TIBCO documentation for more details about TIBCO Rendezvous session parameters.

5.4 CORBA troubleshooting

How to check if CORBA services are started correctly?

Using the sqmadm user, retrieve the Jacorb Name Service process ID with the temip_sc_show_kernel command:

```
MINE  PID   UserId  Component               Platform  Start     CPU%  Time   Size  State
…
Yes   6242  sqmadm  Jacorb NS               slmv12    11:54:14  0.26  00:13  892   R
```

Using the root user, invoke the lsof command:

```
lsof | grep TCP | grep -E <PID Jacorb NS>
```

Check that:

- `Jacorb Name server` has opened a socket on port 7171 (LISTEN) This process use other ports (for its own usage) dynamically allocated.

How to check if the User Interface Server is started correctly?

Using the sqmadm user, retrieve the User Interface Server process ID with the temip_sc_show command:

```
PID    UserId  Component    Platform  Director      Appli     Start     CPU%  Time   Size  State
…
20815  sqmadm  sqm_uis      slmv12    presentation  UIS       11:56:12  0.26  00:13  892   R
```

Using the root user, invoke the lsof c ommand:

```
lsof | grep TCP | grep -E <PID SQM_UIS>
```

Check that the UI Server (identified by its process name sqm_invoke) opens a socket on the port 7172 (LISTEN).

Note

`lsof` allows to check the ports used by processes. It can be downloaded from the web.

5.5 How to expand memory allocation of Java processes?

It may be necessary to expand the memory allocated to SQM Java processes. This is specifically the case when SQM manage large or very large models.

Symptom: Out of memory messages are logged into the SQM application log files.

The built-in limit for SQM applications is set to 2400 MB of Java heap. It is possible to get up to 3.8GB of Java heap:

- Check that the bundle PHKL_32578 (or PHKL_28428) is installed
- cd $TEMIP_SC_HOME/adapter/bin
- Execute the following command:

  ```
  chart +q3p enable +q4p enable sqm_invoque
  ```

- Set the Xms (initial heap size) and Xmx (maximum heap size) parameters according to your needs:

  ```
  export JAVA_ARG="-Xms256M –Xmx3000M"
  ```

- Restart your application

You can get more information on how to expand memory at the following URL:

http://www.hp.com/products1/unix/java/infolibrary/prog_guide/expanding_memory.html#ExpandingPA-RISC

Index of Commands

`sqm_dump.ksh`, 134
`sqm_get_trace.ksh`, 133
`sqm_set_trace.ksh`, 132
temip_sc_backup, 49
temip_sc_create_cust, 109
temip_sc_create_dfd, 111
temip_sc_create_expression, 109
temip_sc_create_sd, 110
temip_sc_create_si, 110
temip_sc_create_sig, 110
temip_sc_create_sl, 110
temip_sc_create_sla, 110
temip_sc_dbpasswd, 91, 93, 97
temip_sc_declare_dfi, 111
temip_sc_delete_aplication, 80
temip_sc_delete_cust, 117
temip_sc_delete_dfd, 118
temip_sc_delete_dfi, 118
temip_sc_delete_director, 80
temip_sc_delete_expr, 118
temip_sc_delete_platform, 80
temip_sc_delete_sd, 119
temip_sc_delete_si, 119
temip_sc_delete_sig, 119
temip_sc_delete_sl, 120
temip_sc_delete_sla, 120
temip_sc_get_cust, 111
temip_sc_get_dfd, 112
temip_sc_get_dfdexpbind, 112
temip_sc_get_dfi, 112
temip_sc_get_expr, 113
temip_sc_get_model, 116
temip_sc_get_sc, 113
temip_sc_get_sci, 113
temip_sc_get_sd, 114
temip_sc_get_si, 114
temip_sc_get_sig, 114
temip_sc_get_sl, 115
temip_sc_get_sla, 115
`temip_sc_hawk_display_start`, 36
`temip_sc_install_SQMboot`, 26
temip_sc_kernel_start, 27, 32
temip_sc_license, 54
temip_sc_listen, 122
temip_sc_lock_dfi, 117
temip_sc_lock_sla, 116
temip_sc_ns_admin_tool.sh, 120
temip_sc_publish, 124
`temip_sc_purge_data.sh`, 25
temip_sc_request, 126
`temip_sc_selfmgmt`, 73
`temip_sc_setup`, 75
temip_sc_show, 43
temip_sc_show_application, 35
temip_sc_show_director, 34
temip_sc_show_kernel, 44
temip_sc_show_platform, 33
temip_sc_start_application, 28
temip_sc_start_director, 28
temip_sc_start_platform, 27
temip_sc_stop_application, 31
temip_sc_stop_director, 30
temip_sc_stop_platform, 30
temip_sc_unlock_dfi, 117
temip_sc_unlock_si, 116, 117
temip_sc_unlock_sla, 116
`tibrvlisten`, 138
`tibrvsend`, 138