Oracle Service Architecture Leveraging Tuxedo

Administration Guide

12*c* Release 1 (12.1.1)

June 2012

ORACLE®

Contents

Oracle Service Architecture Leveraging Tuxedo Administration Overview

The following sections provide an overview to Oracle Service Architecture Leveraging Tuxedo (SALT) administration topics:

- Administering Oracle Service Architecture Leveraging Tuxedo

- Oracle Service Architecture Leveraging Tuxedo Web Services Administrative Tasks and Tools

- Oracle Service Architecture Leveraging Tuxedo Web Application Server Administrative Tasks and Tools

- Oracle SALT WS-TX Administrative Tasks and Tools

Administering Oracle Service Architecture Leveraging Tuxedo

This section explains the following basic concepts for administering SALT:

- Oracle Tuxedo Service Metadata Repository

- Oracle Service Architecture Leveraging Tuxedo Web Service Deployment Model

Oracle Tuxedo Service Metadata Repository

The Oracle Tuxedo Service Metadata Repository was developed to facilitate saving and retrieving Oracle Tuxedo service metadata. Oracle Tuxedo service metadata is a collection of Oracle Tuxedo service attributes that are especially useful in describing the request/response

details of an Oracle Tuxedo service. The SALT gateway server (GWWS), relies on the Oracle Tuxedo Service Metadata Repository for conversions between the Oracle Tuxedo request/response format (buffer types) and standard SOAP message format.

When exposing Oracle Tuxedo services as Web services using SALT, you must define and load your Oracle Tuxedo service metadata in the Oracle Tuxedo Service Metadata Repository. SALT can then define the corresponding SOAP message format from the Oracle Tuxedo service metadata.

When invoking external Web services from an Oracle Tuxedo application, SALT provides a WSDL file converter, wsdlcvt. This command utility helps you to define Oracle Tuxedo service metadata from each Web service operation. The converted services are called SALT proxy services and can be invoked as normal Oracle Tuxedo services. SALT proxy services also need to be loaded in the Oracle Tuxedo Service Metadata Repository.

To retrieve the Oracle Tuxedo service metadata information, you must configure the Oracle Tuxedo Service Metadata Repository system server (TMMETADATA), to be booted in the Oracle Tuxedo application.

Note: TMMETADATA must be booted prior to using any SALT gateway GWWS server.

For more information, see "Oracle Tuxedo Service Metadata Repository" and Using Oracle Tuxedo Service Metadata Repository for Oracle Service Architecture Leveraging Tuxedo in the SALT Configuration Guide.

Oracle Service Architecture Leveraging Tuxedo Web Service Deployment Model

Deploying SALT requires two configuration file types:

- SALT Web Service Definition File (WSDF)

- SALT Deployment File (SALTDEPLOY)

SALT Web Service Definition File

The SALT Web Service Definition File (WSDF) is an XML-based file used to define SALT Web service components (Web Service Bindings, Web Service Operations, Web Service Policies, and so on). The WSDF is an SALT specific representation of the Web Service Definition Language data model. There are two WSDF types: native and non-native.

- Native WSDF

A native WSDF is created manually. You must define a set of Oracle Tuxedo services and how they are exposed as Web services in the WSDF. It looks similar to the SALT 1.1 configuration file. The native WSDF is the input file for the SALT WSDL generator (tmwsdlgen). For more information, see the SALT Configuration Guide.

● Non-native WSDF

A non-native WSDF is generated from an external WSDL file that has been converted using the SALT WSDL converter (wsdlcvt). Basically, you do not need to change the generated WSDF (except to configure advanced features). For more information, see Configuring Web Services.

Oracle Service Architecture Leveraging Tuxedo Deployment File

The SALT Deployment File (SALTDEPLOY) is an XML-based file used to define SALT GWWS server deployment information on a *per* Oracle Tuxedo machine basis. The SALTDEPLOY file lists all necessary WSDF files. It also specifies how many GWWS servers are deployed on an Oracle Tuxedo machine and associates inbound and outbound Web service endpoints for each GWWS server. The SALTDEPLOY file contains a system section where global resources are configured (including certificates and plug-in load libraries). For more information, see Creating the SALT Deployment File in the SALT Configuration Guide.

Figure 1-1 illustrates the SALT deployment model.

Figure 1-1 SALT Deployment Model
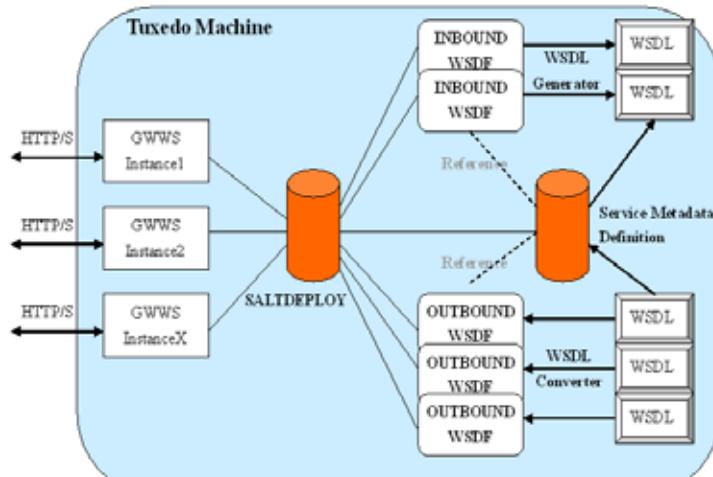
Oracle Service Architecture Leveraging Tuxedo Web Services Administrative Tasks and Tools

SALT provides a set of command utilities for managing different parts of an SALT application built on the Oracle Tuxedo system. These utilities can be used for the following tasks:

- Configuring an Oracle Service Architecture Leveraging Tuxedo Application Using Command-Line Utilities
- Administering an Oracle Service Architecture Leveraging Tuxedo Application Using Command-Line Utilities

Configuring an Oracle Service Architecture Leveraging Tuxedo Application Using Command-Line Utilities

You can configure your SALT application by using command-line utilities. Specifically, you can use an XML editor to create and edit the configuration file (WSDF files and SALTDEPLOY file) for your application, and then use the command-line utility, wsloadcf, to translate the XML files (SALTDEPLOY file and referenced WSDF files) to a binary file (SALTCONFIG). You are then ready to boot the SALT gateway (GWWS) servers.

The following list identifies SALT command-line utilities that you can use to configure your application:

- wsloadcf(1)

  A command that is initiated on each Oracle Tuxedo machine. It allows you to compile your application SALTDEPLOY file and referenced WSDF files into the binary SALTCONFIG file. The wsloadcf command loads the binary file to the location defined by the SALTCONFIG environment variable.

- wsdlcvt(1)

  A command that converts an external Web Service Description Language (WSDL) file into Oracle Tuxedo definition files (WSDF file, Oracle Tuxedo Service Metadata definition file, FML32 field table file and XML Schema file). The generated WSDF file is a non-native WSDF file used for SALT outbound calls specifically.

Since SALT is built on the Oracle Tuxedo framework, you should also use the following Oracle Tuxedo provided command-line utilities to configure SALT specific items in an Oracle Tuxedo application:

- tmloadcf(1)

A command that runs on the master Oracle Tuxedo machine. It is used to compile the Oracle Tuxedo application UBBCONFIG file into the binary TUXCONFIG file. To boot SALT gateway servers, you must define GWWS servers in the UBBCONFIG file.

- tmloadrepos(1)

  A command that runs on the machine where the Oracle Tuxedo Service Metadata Repository System Server (TMMETADATA) is booted. It loads the Oracle Tuxedo service metadata definition text files into the binary Oracle Tuxedo Service Metadata Repository file. You must load all existing Oracle Tuxedo services that are to be exposed as Web service operations in the Oracle Tuxedo Service Metadata Repository. You must also load all wsdlcvt generated SALT proxy services in the Oracle Tuxedo Service Metadata Repository.

- saml

  The wsadmin sub-command saml can be used to manage SAML related metadata for use with the SAML Single Sign-On capability. For more information, see wsadmin in the Oracle Service Architecture Leveraging Tuxedo Command Reference.

Administering an Oracle Service Architecture Leveraging Tuxedo Application Using Command-Line Utilities

You can use the wsadmin(1) command-line utility to perform administrative functions for SALT gateway servers in your Oracle Tuxedo applications. Similar to the tmadmin, dmadmin and qmadmin commands, wsadmin is an interactive meta-command that enables you to run sub-commands.

In an Oracle Tuxedo application, you can run wsadmin(1) on any machine to monitor and manage the SALT gateway servers defined in the Oracle Tuxedo application.

Oracle Service Architecture Leveraging Tuxedo Web Application Server Administrative Tasks and Tools

SALT provides the following Web application server administration tools:

- Web module mod_tuxedo (for Apache or OHS), and tux_nsapi (for iPlanet). The configuration elements are located in the respective configuration files (httpd.conf or magnus.conf).

- The WEBHNDLR system server which serves HTTP requests proxied from mod_tuxedo or tux_nsapi. This system server is configured in the UBBCONFIG file and administered using regular Oracle Tuxedo commands.

For more information, see SALT Installation Guide, Administering SALT at Runtime and Configuring an Oracle SALT Application.

Oracle Service Architecture Leveraging Tuxedo WS-TX Administrative Tasks and Tools

SALT provides a set of command utilities for managing and supporting WS-TX transactions.The names and usages of these utilities are the same as in the SALT Reference Guide.

For more information, see WX-TX Support in Oracle Tuxedo Interoperability and Configuring Oracle Service Architecture Leveraging Tuxedo WX-TX Support in the SALT Configuration Guide.

See Also

- Oracle Service Architecture Leveraging Tuxedo Installation Guide
- Oracle Service Architecture Leveraging Tuxedo Configuration Guide
- Oracle Service Architecture Leveraging Tuxedo Programming Guide
- Oracle Service Architecture Leveraging Tuxedo Reference Guide
- Oracle Service Architecture Leveraging Tuxedo Interoperability Guide

Administering SALT at Runtime

This chapter contains the following topics:

- Administering Oracle Tuxedo Web Services

Administering Oracle Tuxedo Web Services

This section contains the following topics:

- Browsing to the WSDL Document from the GWWS Server

- Tuning the GWWS Server

- Tracing the GWWS Server

- Monitoring the GWWS Server

- Troubleshooting SALT

Browsing to the WSDL Document from the GWWS Server

Each `GWWS` server automatically generates a WSDL document for each deployed inbound native WSDF. The WSDL document can be downloaded from any of the HTTP/S listening endpoints via HTTP GET.

Use the following URL to browse the WSDL document:

```
"http(s)://<host>:<port>/wsdl[? [id=<wsdf_name>]
[&mappolicy=<pack|raw|mtom>] [&toolkit=<wls|axis>]]"
```

Table 2-1 lists all WSDL document download options.

Table 2-1 WSDL Download Options

Option	Value Description
id	Specifies the native WSDF name for the WSDL document. The specified native WSDF must be imported via inbound direction by the GWWS server. If the option is not specified, the first inbound native WSDF is used.
mappolicy	{pack | raw | mtom} Specifies the data mapping policies for certain Oracle Tuxedo Typed buffers for the generated WSDL document. Currently, this option impacts CARRAY typed buffers only. If the option is not specified, pack is used as the default value.
toolkit	{wls | axis} Use this option only if you have previously defined mappolicy=raw. Specify the client toolkit used so that the proper WSDL document description for a CARRAY typed buffer MIME attachment is generated. SALT supports WebLogic Server and Axis for SOAP with Attachments. The default value is wls.

Note: The WSDL download URL supported by SALT 2.0 and later is different from the SALT

1.1 release. In SALT 1.1, one GWWS server adaptively supports both RPC/encoded and

document/literal message style, both SOAP 1.1 and SOAP 1.2 version, from a given

configuration file. In SALT 2.0 and later, each WSDF file associated with the GWWS

server must be pre-combined with a certain SOAP version and a certain SOAP message style. So the following WSDL download options for SALT 1.1 GWWS server are deprecated in this release.

Table 2-2 Deprecated WSDL Download Options

Option	Value Description
SOAPversion	This deprecated option is used to specify the expected SOAP version defined in the generated WSDL document. Now this option is set in the WSDF file.
encstyle	This deprecated option is used to specify the expected SOAP message style defined in the generated WSDL document. Now this option is set in the WSDF file.

Tuning the GWWS Server

The GWWS server is a high performance gateway used between external Web Service application and the Oracle Tuxedo application. It uses a thread-pool working model to improve performance in a multi-processor server environment. The GWWS server also provides options

to control runtime behavior by setting the `<WSGateway>` element property values in the SALT

configuration file. The following topics list deployment considerations based on different

scenarios. For more information, see Configuring the GWWS Servers in the SALT Configuration

Guide.

Thread Pool Size Tuning

Property: `thread_pool_size`

The default thread pool size is 16, but in some cases this may not be enough to handle high volume loads. It is recommended to conduct a typical usage analysis in order to better estimate the proper size requirement. Usually, if the concurrent client number is large (for example, more than 500), it is suggested that you deploy the GWWS gateway on a server with at least a 4-way processor and set the thread pool size to 64.

Network Timeout Control

Property: `timeout`

SALT provides a network timeout tuning parameter in the configuration file. The default timeout value is 300 seconds.The value can be adjusted to reduce timeout errors.

Max Content Length Control

Property: `max_content_length`

SALT administrators may want to limit the buffer size sent from a client. SALT supports this by using a property value that can be set for particular GWWS instances. By default there is no limit.

Backlog Control

Property: `max_backlog`

The default backlog socket listen value is 20. On some systems, such as Windows, 20 may not meet heavy load requirements. The client connection is rejected during TCP handshake.

The recommended value for Windows is based on the max concurrent TCP connections you may encounter. For example, if 80 is the peak point, you may configure the `max_backlog` property value to 60 in the SALT configuration file.

Note: The default backlog value is adequate for most systems. You do not need to tune it unless you experience client connection problems during heavy loads.

WARNING: A large backlog value may increase *syn-blood* attack risk.

Oracle Tuxedo BLOCKTIME

A network receive timeout property is provided in the SALT configuration file. Web service applications are also impacted by the Oracle Tuxedo BLOCKTIME parameter. Blocktime accounting begins when a message is transformed from XML to a typed buffer and delivered to the Oracle Tuxedo framework.

If no reply is received for a particular Web service client within the BLOCKTIME time frame, the GWWS server sends a SOAP fault message to the client and terminates the connection. If the GWWS server receives a delayed reply, it drops this message because the client has been disconnected.

BLOCKTIME is defined in the UBBCONFIG file *RESOURCE section.

Boost Performance Using Multiple GWWS instances

If one GWWS instance is bottlenecks due to network congestion, low CPU resources and so on, multiple GWWS instances can be deployed with the same Web Service binding on distributed Oracle Tuxedo nodes.

Note: Even though multiple GWWS instances can provide the same logic functionality, from a client perspective, they are different Web service endpoints with different HTTP/S listen ports and addresses.

Tracing the GWWS Server

The GWWS server supports Oracle Tuxedo TMTRACE functionality (used to dynamically trace messages). All trace points are logged in the ULOG file. Checking the ULOG file trace information helps to evaluate GWWS server SOAP message problems. GWWS server message tracing behavior is set using the TMTRACE environment variable, or by using the tmadmin chtr sub-command.

The reserved trace category, msg, is used to trace SALT messages. It can be used together with other general trace categories. For example, if trace category "atmi+msg" is specified, both SALT and Oracle Tuxedo ATMI trace messages are logged.

Notes: Message tracing is recommended for diagnostic treatment only.

The following trigger specifications are not recommended for GWWS servers:

abort, system, sleep

In any of these trigger specifications are used, GWWS servers may be unexpectedly terminated.

For more tmtrace and trace specification information, see tmtrace(5) in the *File Formats, Data Descriptions, MIBs, and System Processes Reference*.

TMTRACE specification examples for SALT message tracing are shown below:

- To trace SALT messages only
  ```
  export TMTRACE=msg:ulog:export GWWS_TRACE_LEVEL=100
  ```

- To trace both SALT and Oracle Tuxedo ATMI messages

  ```
  export TMTRACE=atmi+msg:ulog:export GWWS_TRACE_LEVEL=100
  ```

Listing 2-1 shows a ULOG file example containing SALT tracing messages.

Listing 2-1   TMTRACE Messages Logged By GWWS Server

```
183632.BOX1!GWWS.4612.4540.0: TRACE:ms:A HTTP message is received, SCO
index=1023

183632. BOX1!GWWS.4612.4540.0: TRACE:ms:A SOAP message is received, SCO
index=1023

183632. BOX1!GWWS.4612.4540.0: TRACE:ms:Begin data transformation of
request message, buffer type = STRING, SCO index=1023

183632. BOX1!GWWS.4612.4540.0: TRACE:ms:End of data transformation of
request message, buffer type = STRING, SCO index=1023

183632. BOX1!GWWS.4612.840.0: TRACE:ms:Delivering a message to Tuxedo,
service name =TOUPPER,  SCO index=1023

183632. BOX1!GWWS.4612.840.0: TRACE:ms:Got a message from Tuxedo, SCO
index=1023

183632. BOX1!GWWS.4612.4540.0: TRACE:ms:Begin data transformation of reply
message, buffer type = STRING, SCO index=1023

183632. BOX1!GWWS.4612.4540.0: TRACE:ms:End of data transformation of reply
message, buffer type = STRING, SCO index=1023

183632. BOX1!GWWS.4612.4540.0: TRACE:ms:Send a http message to net, SCO
index=1023
```

A more complex log is generated by TMTRACE=msg:ulog, used in WS-ReliableMessaging communication. All the application and infrastructure messages are sent to ULOG. Listing 2-2 shows a ULOG file example containing WS-ReliableMessaging TMTRACE messages.

Listing 2-2 WS-ReliableMessaging TMTRACE Messages

```
184706.BOX1!GWWS.3640.4772.0: TRACE:ms:A HTTP message is received, SCO
index=1023

184706.BOX1!GWWS.3640.4772.0: TRACE:ms:A HTTP Get request is received, SCO
index=1023

184706.BOX1!GWWS.3640.4772.0: TRACE:ms:Send a http message to net, SCO
index=1023

184710.BOX1!GWWS.3640.4772.0: TRACE:ms:A HTTP message is received, SCO
index=1022

184710.BOX1!GWWS.3640.4772.0: TRACE:ms:A SOAP message is received, SCO
index=1022

184710.BOX1!GWWS.3640.4772.0: TRACE:ms:Create a new inbound sequence,
ID=uuid:4F1FEE40-72CB-118C-FFFFFFC0FFFFFFA8FFFFFFEB010000-1811

184710.BOX1!GWWS.3640.4772.0: TRACE:ms:Create a new outbound sequence,
ID=uuid:f7f76200-f612-11da-990d-9f37c3d14ba7

184710.BOX1!GWWS.3640.4772.0: TRACE:ms:Send CreateSequenceResponse message
for sequence uuid:4F1FEE40-72CB-118C-FFFFFFC0FFFFFFA8FFFFFFEB010000-1811

184710.BOX1!GWWS.3640.4772.0: TRACE:ms:Send a http message to net, SCO
index=1022

184712.BOX1!GWWS.3640.3260.0: TRACE:ms:A HTTP message is received, SCO
index=1022

184712.BOX1!GWWS.3640.3260.0: TRACE:ms:A SOAP message is received, SCO
index=1022

184712.BOX1!GWWS.3640.3260.0: TRACE:ms:Begin data transformation of request
message, buffer type = STRING, SCO index=1022
```

```
184712.BOX1!GWWS.3640.3260.0: TRACE:ms:End of data transformation of
request message, buffer type = STRING, SCO index=1022

184712.BOX1!GWWS.3640.3260.0: TRACE:ms:Received a request message in
sequence uuid:4F1FEE40-72CB-118C-FFFFFFC0FFFFFFA8FFFFFFEB010000-1811
```

Checking the ULOG tracing information helps to evaluate GWWS server SOAP message problem
status.

Monitoring the GWWS Server

The GWWS server can be monitored with wsadmin utility, which is a command-line tool. This
tool can show the running status of GWWS.

An example is shown in .

Listing 2-3   Use wsadmin to Monitor GWWS

```
$wsadmin
wsadmin - Copyright (c) 2005-2010 Oracle.
Portions * Copyright 1986-1997 RSA Data Security, Inc.
All Rights Reserved.
Distributed under license by Oracle.
SALT is a registered trademark.

> gwstats -i abcd
GWWS Instance : abcd

Inbound Statistics :
--------------------------------
     Request Response Succ :      74
     Request Response Fail :      32
               Oneway Succ :       0
               Oneway Fail :       0

                Total Succ :      74
```

```
                    Total Fail :     32

          Avg. Processing Time : 210.726 (ms)
Outbound Statistics :
-------------------------------
     Request Response Succ :      0
     Request Response Fail :      0
                Oneway Succ :      0
                Oneway Fail :      0

                 Total Succ :      0
                 Total Fail :      0

          Avg. Processing Time :  0.000 (ms)
-------------------------------
     Total request Pending :      0
  Outbound request Pending :      0
      Active Thread Number :      2


> gws -i out -s getTemp
GWWS Instance : out

Service : getTemp

Outboud Statistics :
-------------------------------
     Request Response Succ :    333
     Request Response Fail :    139
      Avg. Processing Time : 143.064 (ms)


>
```

The gwstats command (abbreviated as gws), displays the WWS server statistics data with a specific instance ID or of a certain GWWS server service of the. The data includes the amount of successful and failed request, etc.

Before wsadmin is executed, both the TUXCONFIG and SALTCONFIG environment variables must be set. wsadmin supports both active mode and in-active mode, which means wsadmin is able to launch with/without booting the Oracle Tuxedo domain.

Table 2-3 lists wsadmin sub-commands.

Table 2-3 wsadmin Sub-Commands

Sub-Command	Description
gwstats(gws)	Displays statistics information of GWWS server
configstats(cstat)	Displays configuration information
default(d)	Specifies the default -i option
echo(e)	Switches echo input on/off
forgettrans	Forgets one or all heuristic log records for the named GWWS instance.
printtrans	Prints transaction information for the named GWWS instance.
paginate(page)	Switches paging output on/off
verbose(v)	Switches verbose output on/off
quit(q)	Terminates the program

Troubleshooting SALT

This section contains the following topics:

- GWWS Start Up Failure

- GWWS Rejects SOAP Request

- WSDL Document Generated Incorrectly or Rejected by SOAP Client Toolkit

GWWS Start Up Failure

If the GWWS server fails to start, check the following:

- Oracle Tuxedo service contract configuration

  – Check if the Oracle Tuxedo service contract definition is correct in the Oracle Tuxedo Service Metadata Repository

  – Check if the Oracle Tuxedo Service Metadata Repository Server - TMMETADATA - is booted successfully.

- GWWS server license

  The GWWS server requires an extra license from Oracle to enable the functionality. Check to make sure it has been installed properly.

- GWWS server HTTP listen port configuration.

  Check the GWWS server listen /WS-Addressing endpoints defined in the SALT configuration files. Avoid port conflicts with other applications.

- GWWS instance ID.

  Check the GWWS instance ID to make sure the two names defined in UBBCONFIG and SALTDEPLOY file are consistent.

- UBBCONFIG file MAXWSCLIENTS definition.

  Make sure that MAXWSCLIENTS is defined in theUBBCONFIG file *MACHINE section on the computer where the GWWS server is deployed.

- RESTART=Y and REPLYQ=Y parameters.

  If the GWWS server is set to RESTART=Y in the UBBCONFIG file, REPLYQ=Y must also be defined.

- SALTCONFIG file.

  Make sure the binary version SALTCONFIG file is compiled successfully and the environment variable SALTCONFIG is set correctly for the GWWS server.

GWWS Rejects SOAP Request

In some cases, the GWWS server may reject SOAP requests. The most common causes are:

- The WSDL document is outdated.

The WSDL document used by SOAP clients is out of date and some services may not be available.

- The GWWS server environment variables are not set correctly.

  When exporting an Oracle Tuxedo service with FML/VIEW buffers to a Web service, make sure the related GWWS environment variables are set with valid values. The GWWS server needs this information for the data mapping conversion.

- Violated Oracle Tuxedo Service Metadata Repository restrictions.

  Check the SOAP client data and make sure Oracle Tuxedo Service Metadata Repository restrictions are not violated.

- Unavailable Oracle Tuxedo service.

  Make sure the Oracle Tuxedo service you want exported as a Web service is available.

WSDL Document Generated Incorrectly or Rejected by SOAP Client Toolkit

If the WSDL document is rejected by the Web Service client toolkit, do the following:

- Try to use the `document/literal` message style and SOAP 1.1 to define native Oracle Tuxedo `WSDF` file. This is also the default behavior.

- Use `tmwsdlgen` to generate the WSDL document manually and compare it with the one downloaded by the GWWS server. If the TMMETADATA server is not started when the GWWS server booted, the GWWS server cannot obtain the correct service contract information. Therefore, the downloaded WSDL document does not contain the correct type definitions.

See Also

- tmadmin
- tmtrace
- Configuring a SALT Application
- SALT Programming
- SALT Command Reference